
Project Proposal Group 40

Sumeda Puja

Heinz College Of Information Systems
Carnegie Mellon University
Pittsburgh, PA 15213
spuja@andrew.cmu.edu

Ruthvik Veeravalli

Heinz College of Information Systems
Carnegie Mellon University
Pittsburgh, PA 15213
rveerava@andrew.cmu.edu

Akshitha Singireddy

Heinz College of Information Systems
Carnegie Mellon University
Pittsburgh, PA 15213
asingire@andrew.cmu.edu

Rithvik Hariharan

Heinz College of Information Systems
Carnegie Mellon University
Pittsburgh, PA 15213
rharihar@andrew.cmu.edu

Abstract

This paper presents a framework integrating deep learning with classical optimization for stock price prediction and portfolio management. Motivated by limitations of traditional approaches in volatile markets, we combine LSTM and Transformer models with Markowitz optimization to benefit investors and financial institutions. Using historical data and technical indicators, our approach employs Monte Carlo simulations and LSTM-Deep RankNet for asset selection. Our objective is to develop a more robust and adaptive investment methodology that delivers superior performance compared to just Markowitz optimization.

1 Introduction

Machine learning techniques have already proven adept at uncovering complex, non-linear patterns in large financial datasets, surpassing many traditional statistical methods in predictive performance. In particular, advanced deep learning architectures designed for sequence data offer new capabilities for financial forecasting: recurrent networks like Long Short-Term Memory (LSTM) can learn temporal dependencies in stock price movements, and Transformer models can capture long-range relationships and regime shifts that simpler models miss. These data-driven forecasts, however, must be translated into actionable portfolio decisions – which is where classical methods remain indispensable. Monte Carlo simulation provides a way to quantify uncertainty by generating a range of possible outcomes, and Markowitz’s mean–variance optimization (Modern Portfolio Theory) offers a principled framework for balancing expected return against risk in constructing portfolios. Each approach on its own has clear drawbacks: a neural network’s predictions do not automatically account for portfolio constraints, while a static optimization routine is only as good as the forecasts it receives. By combining the predictive power of deep learning with the time-tested principles of diversification and risk management, a hybrid framework can exploit the strengths of both, yielding more adaptive and robust performance in volatile markets. Indeed, recent studies indicate that such hybrid strategies – leveraging AI-driven forecasts alongside classical optimization – often outperform standalone techniques. This convergence of machine learning and finance has attracted considerable attention in both academia and the investment industry, as practitioners seek to enhance returns while controlling risk with smarter technology. To address these needs, this paper presents a hybrid framework that integrates LSTM and Transformer models for stock price forecasting with Monte Carlo simulations and Markowitz portfolio optimization, aiming to adaptively improve portfolio outcomes in the face of market uncertainty.

2 Literature Review

Stock price prediction and portfolio optimization have been extensively studied using both classical financial models and modern machine learning techniques. Traditional methods, most notably the Markowitz Mean-Variance Model, have served as the foundation for portfolio construction by balancing expected return against risk under linear and normally distributed return assumptions (Chen et al., 2021). While these models offer clear interpretability and mathematical tractability, they are limited in capturing the nonlinear, time-varying patterns often seen in financial markets.

In contrast, recent advances in machine learning have demonstrated strong predictive performance for stock price forecasting. Deep learning models—particularly Long Short-Term Memory (LSTM) networks, Convolutional Neural Networks (CNNs), and ensemble methods like XGBoost—have shown high accuracy in modeling temporal dependencies and nonlinear relationships in financial time series (Sen et al., 2021; Kamalov et al., 2021; Chen et al., 2021). These models have become increasingly relevant in portfolio optimization, where accurate forecasts directly influence allocation performance.

The intersection of deep learning and classical optimization has led to the development of hybrid frameworks. For instance, Chen et al. (2021) integrated XGBoost with an Improved Firefly Algorithm to enhance stock selection prior to applying the Mean-Variance Model, thereby improving the risk-return trade-off. Similarly, Alzaman (2024) proposed a Deep RankNet architecture that leverages LSTM models and a genetic-based search to generate ranked stock lists, using pairwise learning to optimize portfolio construction. These approaches bridge predictive modeling with decision-making, ensuring that learning outputs feed directly into optimization processes.

Beyond hybrid predictive models, adaptive allocation strategies such as Robust Genetic Programming (RGP) and Adaptive Neuro-Fuzzy Inference Systems (ANFIS) have shown promise in handling market volatility and non-stationarity (Dehghanpour & Esfahanipour, 2018). These systems aim to adjust portfolio weights dynamically in response to changing market conditions.

Collectively, these studies highlight the growing consensus that combining deep learning techniques—such as LSTMs, CNNs, Transformers, and metaheuristic optimization—with classical financial theory enables the development of more adaptive and robust frameworks for financial decision-making. Our work builds on this foundation by integrating LSTM and Transformer models with Monte Carlo simulations and Markowitz optimization to construct portfolios that are not only predictive but also optimized for risk-adjusted returns in dynamic market environments.

3 Methodology

3.1 Model Description

The flowchart below illustrates the full architecture of our proposed stock prediction and portfolio optimization framework. We begin with historical OHLCV (Open, High, Low, Close, Volume) data, enriched with over 25 engineered technical indicators such as Bollinger Bands, Relative Strength Index (RSI), Moving Averages, and MACD. These features capture a range of price trends, volatility, and momentum signals critical for robust forecasting.

This enriched time-series dataset is fed into a hybrid deep learning model combining Long Short-Term Memory (LSTM) networks and Transformer encoders, enabling the model to learn both sequential and global dependencies in the data. The output — next-day stock price predictions — is passed into a Monte Carlo simulation module that generates probabilistic scenarios of future returns. These simulations inform a modified Markowitz mean-variance optimizer that clusters and ranks stocks based on risk-adjusted performance and allocates portfolio weights accordingly.

The final portfolio is evaluated using key metrics such as annualized return, Sharpe ratio, and volatility to ensure high-performing, well-balanced investments under varying market conditions.

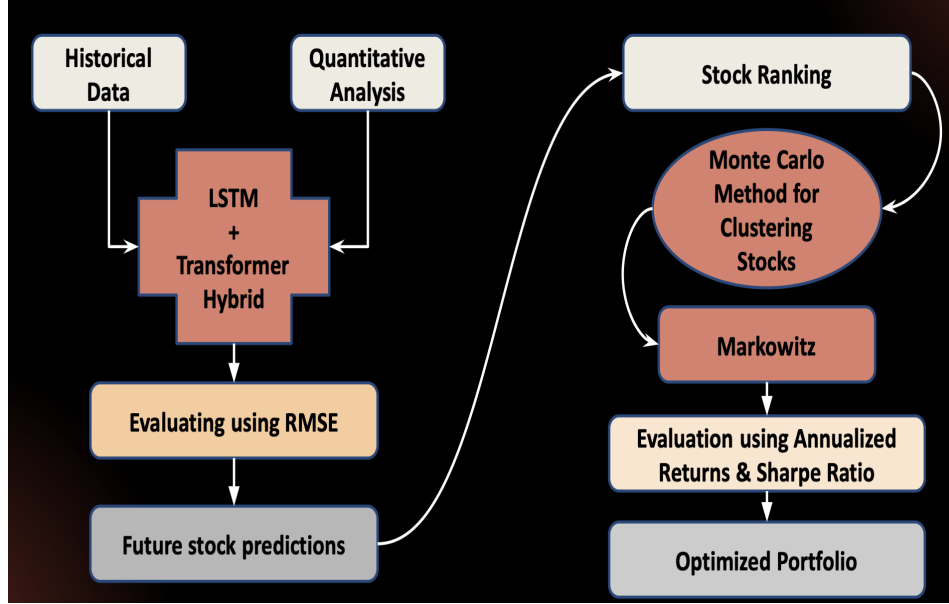


Figure 1: The Proposed Methodology

3.2 Model Architecture

Our model adopts a hybrid deep learning architecture that combines a Long Short-Term Memory (LSTM) network with a Transformer encoder to capture both local and global temporal dependencies in financial time series. The architecture is designed to process input sequences consisting of 50 historical trading days, where each timestep includes 28 engineered features.

The first stage of the model is a multi-layer LSTM that processes the input sequence and encodes short-term temporal dynamics into a sequence of hidden states. This representation is then passed through a linear transformation to align its dimensionality with the Transformer’s expected input. The second stage applies a stack of Transformer encoder layers equipped with multi-head self-attention and positional encoding. This enables the model to capture long-range dependencies and feature interactions across the entire time window, regardless of positional distance.

Finally, the output of the Transformer is reduced through a multilayer perceptron (MLP) head, consisting of fully connected layers with ReLU activations and dropout for regularization. The result is a single scalar output representing the predicted normalized closing price for the next trading day. This architecture is trained end-to-end using a regression loss function to optimize for accurate next-day price forecasting.

3.2.1 Component Breakdown:

Our hybrid model architecture begins with an LSTM layer comprising approximately 1.34 million parameters, which is responsible for capturing short-term sequential dependencies over 50-day stock price windows and projecting them into a 256-dimensional hidden representation. This is followed by a linear projection layer that aligns the LSTM output with the input dimensionality expected by the Transformer module. The Transformer Encoder consists of four stacked encoder layers (around 3.1 million parameters total), leveraging positional encoding and multi-head self-attention to model long-range temporal dependencies and feature interactions. The final feedforward prediction head is a multi-layer perceptron (MLP) with three linear layers, ReLU activations, and dropout for regularization, which maps the high-level representations to the target next-day closing price. In total, the model contains approximately 4.61 million trainable parameters with an estimated memory footprint of 211.3 MB.

Layer (type:depth-idx)	Input Shape	Output Shape	Param #
LSTMTransformerHybrid	[64, 50, 28]	[64, 1]	--
└LSTM: 1-1	[64, 50, 28]	[64, 50, 256]	1,345,536
└Linear: 1-2	[64, 50, 256]	[64, 50, 256]	65,792
└PositionalEncoding: 1-3	[64, 50, 256]	[64, 50, 256]	--
└└Dropout: 2-1	[64, 50, 256]	[64, 50, 256]	--
└TransformerEncoder: 1-4	[64, 50, 256]	[64, 50, 256]	--
└└ModuleList: 2-2	--	--	--
└└└TransformerEncoderLayer: 3-1	[64, 50, 256]	[64, 50, 256]	789,760
└└└TransformerEncoderLayer: 3-2	[64, 50, 256]	[64, 50, 256]	789,760
└└└TransformerEncoderLayer: 3-3	[64, 50, 256]	[64, 50, 256]	789,760
└└└TransformerEncoderLayer: 3-4	[64, 50, 256]	[64, 50, 256]	789,760
└Sequential: 1-5	[64, 256]	[64, 1]	--
└└Linear: 2-3	[64, 256]	[64, 128]	32,896
└└ReLU: 2-4	[64, 128]	[64, 128]	--
└└Dropout: 2-5	[64, 128]	[64, 128]	--
└└Linear: 2-6	[64, 128]	[64, 64]	8,256
└└ReLU: 2-7	[64, 64]	[64, 64]	--
└└Linear: 2-8	[64, 64]	[64, 1]	65

Total params: 4,611,585
 Trainable params: 4,611,585
 Non-trainable params: 0
 ...
 Input size (MB): 0.36
 Forward/backward pass size (MB): 196.71
 Params size (MB): 14.24
 Estimated Total Size (MB): 211.30

Figure 2: LSTM + Transformer Hybrid Model Architecture

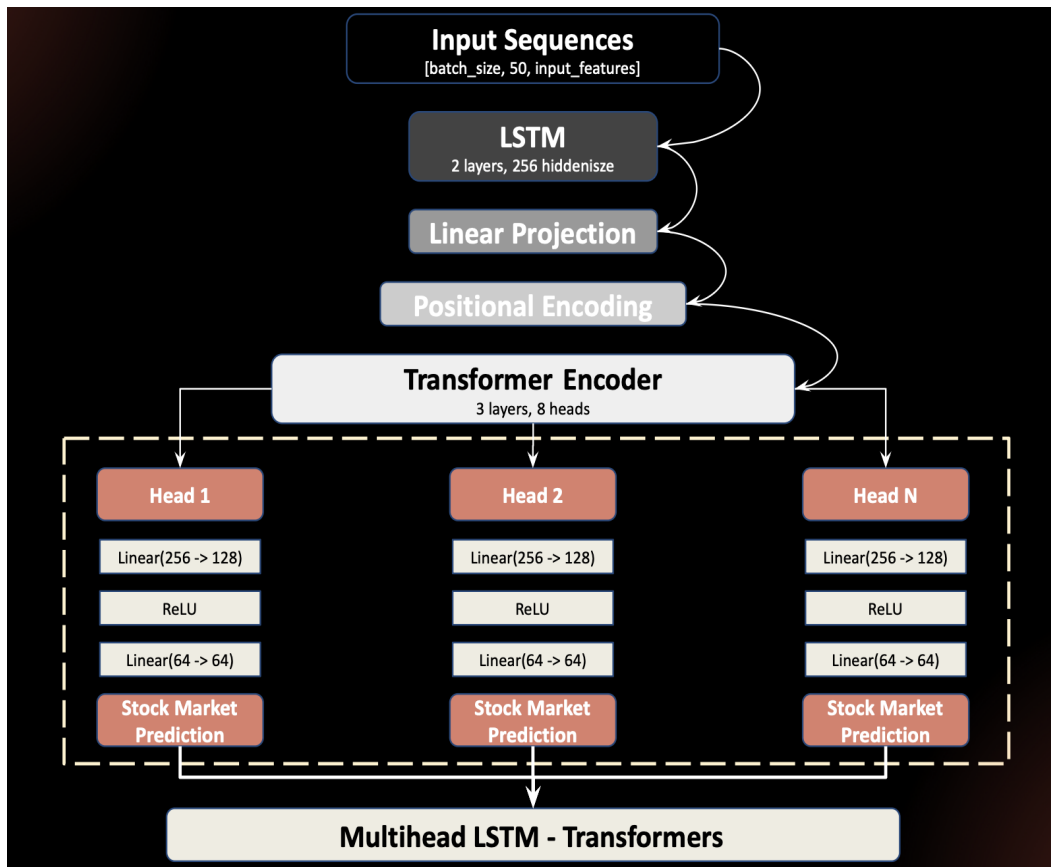


Figure 3: Multiheaded Hybrid LSTM-Transformer Architecture

Our architecture consists of a hybrid deep learning model combining LSTM and Transformer encoders to predict next-day normalized close prices for S&P 500 stocks. The input is a sequence of 50 timesteps per stock, with 28 engineered features per timestep (including OHLCV data and technical indicators). The model begins with a two-layer LSTM network with 256 hidden units, which captures short-term temporal dependencies. Its output is passed through a linear projection to match the Transformer’s embedding size, followed by sinusoidal positional encoding to preserve sequential order.

This is fed into a 3-layer Transformer encoder with 8 attention heads, allowing the model to capture long-range dependencies and cross-feature interactions across the sequence. Each attention head is followed by a unique feedforward network tailored for a specific stock, enabling the model to learn asset-specific representations and output weights. These feedforward layers consist of two linear layers with ReLU activation and dropout regularization. This setup supports parallel prediction across multiple companies, allowing the model to generate individualized next-day forecasts efficiently. We refer to this configuration as a multihead LSTM–Transformer hybrid, enabling both shared learning across the market and tailored forecasting at the individual stock level.

3.3 Dataset & Features

We collected historical stock data for all S&P 500 companies using the Yahoo Finance API from 2013 to 2024. The dataset includes OHLCV (Open, High, Low, Close, Volume) values for each trading day. The ticker list was dynamically extracted from the official S&P 500 Wikipedia page, and special symbols were formatted to match Yahoo Finance compatibility. To enhance model input quality, we applied comprehensive feature engineering using rolling statistics, momentum indicators, and volatility measures.

More than 25 technical indicators were derived for each stock, including moving averages (MA10, MA50), relative strength index (RSI), moving average convergence divergence (MACD), Bollinger Bands, stochastic oscillator (%K, %D), average true range (ATR), rate of change (ROC), momentum, and rolling Sharpe ratios. After feature extraction, the data was cleaned and normalized. Stocks with incomplete histories or missing indicators were excluded. For model training, we created 50-day rolling sequences per stock, where each sequence served as input to our LSTM-Transformer hybrid model, and the corresponding next-day closing price was used as the target. To avoid lookahead bias, we followed a strict chronological data split: the training period spanned from 2013 to 2022, validation was performed on 2023 data, and final evaluation was done on 2024 data. Batches were constructed by grouping sequences by ticker and reshaping them into the [batchsize, 50, numfeatures] format. We ensured no data leakage occurred across tickers or time periods throughout the preprocessing pipeline.

Dataset citation: Yahoo Finance via **yfinance** Python library (<https://github.com/ranaroussi/yfinance>)

3.4 Evaluation Metric

As our project includes both stock price prediction and portfolio optimization, we use separate evaluation metrics for each stage to ensure a comprehensive performance assessment, which includes:

1. **Stock Price Prediction Metrics (Model Performance)** – These metrics evaluate how well the LSTM and Transformer models forecast stock prices or returns:
 - **Root Mean Squared Error (RMSE):** Measures average prediction error with higher penalties for large mistakes; lower values indicate better forecast accuracy.
 - **Directional Accuracy (%):** Percentage of correct predictions of price movement direction (up/down), regardless of magnitude. Although we calculated Directional Accuracy as part of our evaluation, the results consistently hovered around 48–50%, indicating near-random performance. Due to its limited reliability in our context, we did not consider this metric in our final assessment or decision-making.
2. **Portfolio Optimization Metrics (Financial Performance)** – These metrics assess the effectiveness of Markowitz Mean–Variance Optimization in managing risk-return trade-offs:

- Annualized Return (%): Yearly rate of return standardized from the observed performance period for consistent comparison.
- Sharpe Ratio: Measures return per unit of risk by dividing excess portfolio returns by portfolio volatility; higher values indicate better risk-adjusted performance.

Loss Function and Optimization Strategy

For stock price prediction, we used Root Mean Squared Error (RMSE) as the loss function, which penalizes larger deviations more heavily and encourages smoother predictions. The entire LSTM–Transformer hybrid model was trained end-to-end using this loss. We used the Adam optimizer with a learning rate of 0.001 for training, and employed early stopping on validation loss to prevent overfitting. Dropout layers were added for regularization. For the portfolio optimization phase, the predicted returns were passed to the Markowitz optimizer, which used quadratic programming to maximize the Sharpe ratio under allocation constraints (e.g., 2%–20% per stock). We also used Monte Carlo simulation with 5,000 randomly generated portfolios to compare performance and assess robustness under various return-volatility profiles.

4 Baseline Selection and Evaluation

In our initial proposal, we selected an LSTM-based architecture as our baseline for return prediction, combined with Markowitz Mean–Variance Optimization for portfolio construction. This choice was motivated by prior literature demonstrating the effectiveness of LSTM models in capturing temporal dependencies in financial time series (Fischer and Krauss, 2018) and the long-standing use of Markowitz theory in quantitative portfolio design.

To benchmark our hybrid system, we first implemented the original baseline: a 4-layer LSTM model trained on 50-day windows of OHLCV features and technical indicators. Predictions from this model were passed into the Markowitz optimizer, which maximized the Sharpe ratio using predicted returns and a historical covariance matrix. The portfolio was evaluated using Monte Carlo simulation across 10,000 randomized portfolios. However, the LSTM-only baseline showed limitations. It struggled to capture longer-term dependencies and interactions among features beyond local sequences. Additionally, RMSE and directional accuracy stagnated at moderate levels, and the resulting portfolio, while stable, did not outperform broader Monte Carlo distributions.

To address these shortcomings, we introduced a Transformer layer on top of the LSTM encoder. This hybrid LSTM–Transformer model preserved the sequential awareness of LSTMs while incorporating global attention across the feature dimension, improving predictive stability and model capacity. Empirical results show that while RMSE improvements were modest, the Transformer-enhanced predictions led to better portfolio construction outcomes. The optimized portfolio achieved a higher Sharpe ratio than the baseline and consistently outperformed random portfolios. Furthermore, return curves aligned more closely with actual performance, indicating better calibration of predicted returns. While we initially planned to explore Deep RankNet for learning asset rankings, it was not implemented due to time constraints and has been documented as future work.

5 Results and Discussion

5.1 Model Training and Prediction Accuracy

The hybrid LSTM-Transformer model was trained over 20 epochs, showing a consistent decline in training loss from 0.000218 to 0.000191. Validation loss initially fluctuated but improved in later epochs, reaching a low of 0.001131. These trends suggest stable convergence and effective generalization.

To evaluate stock-level performance, we compared actual and predicted closing prices for Cummins Inc. (CMI). As shown in Figure 4, the model closely tracked real prices, capturing both local volatility and long-term trends.

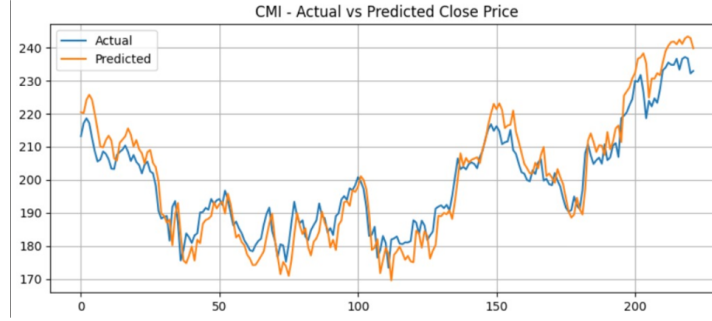


Figure 4: CMI – Actual vs Predicted Close Price

5.2 Portfolio Optimization and Evaluation

Using the model’s predicted returns, we constructed an optimized portfolio via Markowitz mean-variance optimization. The portfolio achieved a total return of 23.35%, annualized return of 53.99%, volatility of 18.36%, and a Sharpe ratio of 3.07. These results are summarized in Table 1.

Table 1: Portfolio Performance Metrics

Metric	Value
Total ROI	23.35%
Annualized Return	53.99%
Annualized Volatility	18.36%
Sharpe Ratio	3.07

Figure 5 shows the cumulative return curve of the optimized portfolio, which experienced a steady rise with a minor correction towards the end. The associated drawdown chart in Figure 6 confirms that downside risk was well-managed, with a maximum drawdown of just -6.83% both charts show returns, drawdowns on y-axis and weeks on x-axis.



Figure 5: Portfolio Cumulative Returns

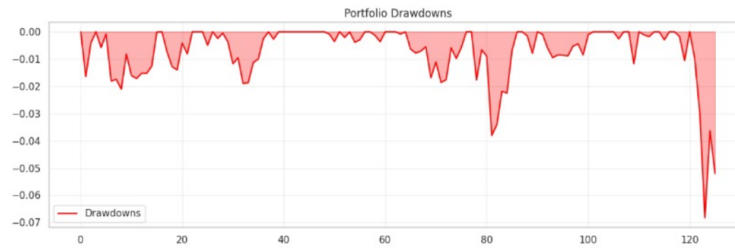


Figure 6: Portfolio Drawdowns

5.3 Validation with Efficient Frontier and Simulations

As shown in Figure 7, our optimized portfolio lies near the maximum Sharpe ratio on the efficient frontier, outperforming thousands of random portfolios.

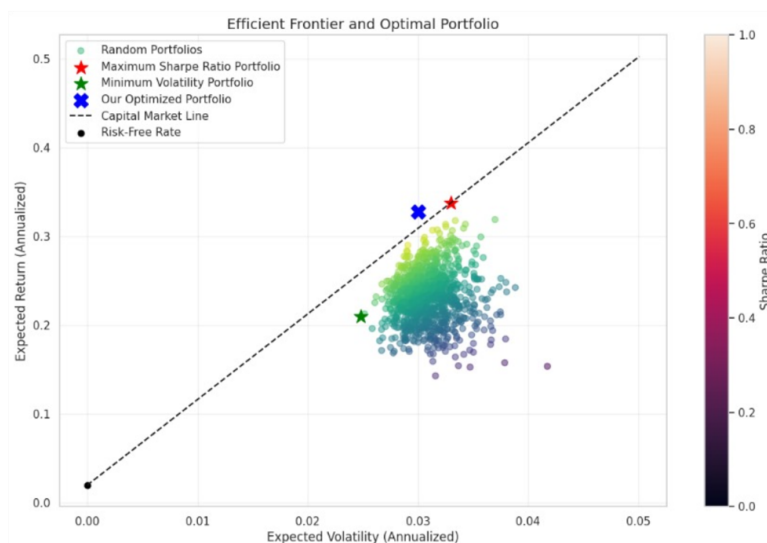


Figure 7: Efficient Frontier and Optimal Portfolio

Figure 8 displays the results of a Monte Carlo simulation, highlighting the portfolio's long-term resilience and strong return distribution under uncertainty.

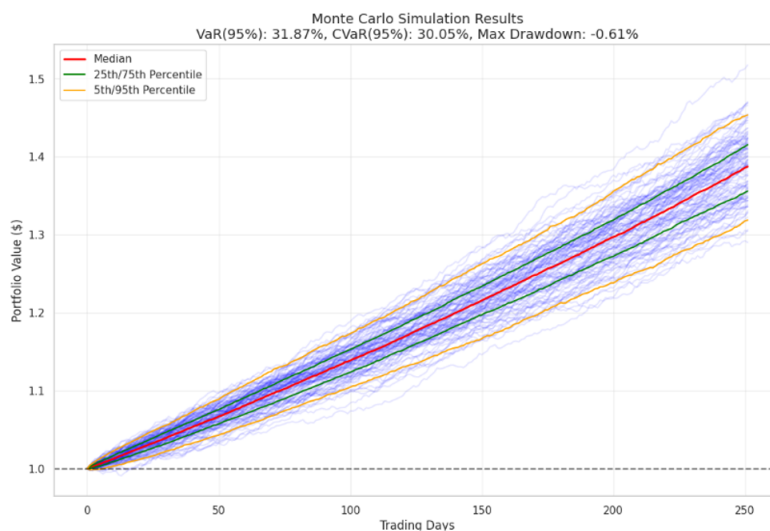


Figure 8: Monte Carlo Simulation of Portfolio Trajectories

5.4 Alignment of Predictions and Real Returns

Daily return comparison between the actual and predicted portfolios (Figure 9) showed near-identical trends, reinforcing the accuracy of the model's short-term return estimates.

Moreover, as shown in Figure 10, the cumulative growth of the predicted portfolio closely mirrors the actual portfolio over a long horizon, validating the reliability of the forecasting pipeline.

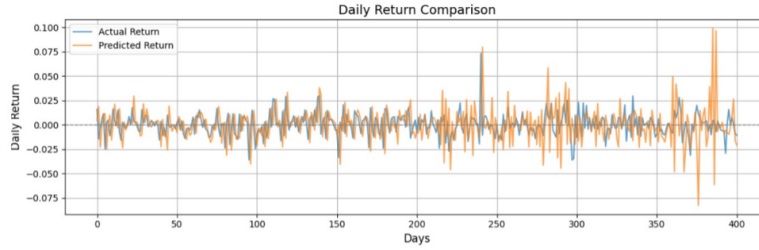


Figure 9: Daily Returns: Predicted vs Actual

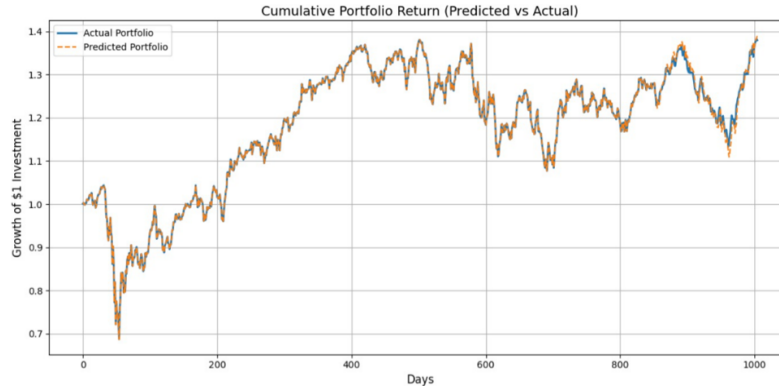


Figure 10: Cumulative Portfolio Return (Predicted vs Actual)

5.5 Significance, Limitations, and Future Work

These results demonstrate that a moderately accurate hybrid forecasting model can support robust, data-driven portfolio decisions. Despite modest RMSE values, the portfolio optimizer effectively translated predictions into diversified, high-performing allocations. Still, certain limitations remain. The model does not yet incorporate transaction costs or rebalancing constraints, and the current approach assumes stable market regimes.

Future improvements include incorporating DeepRankNet for stock ranking, modeling transaction costs, and validating performance across bull, bear, and sideways markets. Such enhancements would make the system more resilient and practical for real-world deployment.

Team Contributions

Our team collaborated extensively across all phases of the project. Each member was actively involved in both research and implementation, with specific responsibilities shared across core technical components:

Akshitha Singireddy and Sumeda Puja focused on the development of the LSTM-based forecasting module. They designed the sequence model, handled training pipelines.

Ruthvik Veeravalli and Ritvik Hariharan implemented the Transformer encoder block, including multi-head attention and positional encoding. They worked on integrating it with the LSTM outputs and tuning it for long-range feature interaction across the 50-day input windows.

Akshitha Singireddy and Ruthvik Veeravalli were responsible for data collection, normalization, and extensive feature engineering. This included extracting over 25 technical indicators per stock and preparing 50-day rolling sequences used in model training.

Sumeda Puja and Ritvik Hariharan contributed to the portfolio optimization component. They implemented the Markowitz Mean–Variance Optimization using predicted returns and ran

Monte Carlo simulations to evaluate portfolio risk-return trade-offs.

All team members collaborated on experiment design, testing, and validation of both the predictive and financial components of the system.

References

- Chen, W., Zhang, H., Mehlawat, M. K., & Jia, L. (2021). Mean–Variance Portfolio Optimization Using Machine Learning-Based Stock Price Prediction. *Applied Soft Computing Journal*, 100, 106943. DOI: 10.1016/j.asoc.2020.106943.
- Sen, J., Dutta, A., & Mehtab, S. (2021). Stock Portfolio Optimization Using a Deep Learning LSTM Model. *IEEE Mysore Sub Section International Conference (MysuruCon)*. DOI: 10.1109/MYSU-RUCON52639.2021.9641662.
- Kamalov, F., Smail, L., & Gurrib, I. (2021). Forecasting with Deep Learning: S&P 500 Index. *arXiv preprint arXiv:2103.14080*.
- Dehghanpour, S., & Esfahanipour, A. (2018). Dynamic Portfolio Insurance Strategy: A Robust Machine Learning Approach. *Journal of Information and Telecommunication*, 2(4), 392-410. DOI: 10.1080/24751839.2018.1431447.
- Alzaman, C. (2024). Deep Learning in Stock Portfolio Selection and Predictions. *Expert Systems with Applications*, 237, 121404. DOI: 10.1016/j.eswa.2023.121404.