Predicting Stock Market Trends Using Deep Neural Architectures

Molly Jenkins Jovany Jimenez Charlie Jia Sophia Contreras
Katia Germanenko

June 2025

Abstract

Predicting stock market trends is challenging due to the nonlinear nature of financial time series data. In this project, we evaluate the forecasting performance of a a few different types of deep learning models: Convolutional Neural Networks (CNN), Variational Autoencoders (VAE), and Radial Basis Function Extreme Learning Machines (RBF-ELM), both individually and in combination with one another. Using a sliding window approach on AAPL stock prices, we compare models based on Mean Absolute Error (MAE) and Fluctuation Around the Trend (FAT). Among all, the RBF-ELM on its own achieved the best results, with the lowest MAE of 1.53 and one of the lowest FAT scores of 14.79. It also performed well on out-of-sample predictions with an MAE of 3.70. On the other hand, the combined models did not offer improvements and often produced less stable forecasts, though they still captured the general form of predicted trends.

1 Introduction

Stock market prediction is a notoriously tough problem, and has been a focal point of research in finance for decades. A big reason is that it is nearly impossible to guess how political events or policy changes will affect the stock market. The data moves fast, reacts suddenly, and rarely follows a clear pattern. Advancements in predictive models within the past decade has seen tremendous growth due to the emergence of powerful AI and machine learning methods. Just a little over 10 years ago, Bormetti et al. (2013) created and utilized a dynamical model for predicting long run stock market returns, completely unrelated to the field of machine learning. Within years, the transition began, with Azari (2019) using an ARIMA approach to analyze and predict bitcoin prices. ARIMA and similar, simple regression models can handle steady trends or linear relationships, but they usually miss the sharp turns and unpredictability of the stock market. More recently, Hemant et al. (2023) and Sarkar et al. (2025) combined various methods into a seamless pipeline that gave promising results. These models can learn from past market behavior better and pick up on patterns that the more basic methods just don't catch.

In this study, we examine the predictive capabilities of three different deep learning models: Convolutional Neural Networks (CNN), Variational Autoencoders (VAE), and Radial Basis Function Extreme Learning Machines (RBF-ELM). We test each model individually and in combination (CNN+VAE, VAE+ELM, and CNN+VAE+ELM) to evaluate how model architecture influences the accuracy of stock price forecasts.

Our work builds on ideas from a recent study by Hemant et al. (2023), which introduced a stock prediction model combining a Convolutional Neural Network (CNN), a Variational Autoencoder (VAE), and a Kernel-based Extreme Learning Machine (KELM). Their approach used technical indicators and historical data to forecast prices on Yes Bank. While they emphasized feature extraction and prediction accuracy using KELM, our study focuses on AAPL stock and relies solely on historical price data as input. We also explore additional model combinations and introduce a new metric, Fluctuation Around the Trend (FAT), which helps evaluate how well predictions reflect the overall shape of market movements rather than just point-by-point accuracy. The goal is not just to find the most accurate model, but to understand how different architectures trade off between performance and consistency.

2 Methodology

2.1 Data and Parameters

We used historical data for AAPL stock from January 1, 2019 through May 5, 2025 and S&P 500, but with the dates January 1, 1928 to June 4, 2025. The data was obtained using the `yfinance` Python library. For all models, input windows were constructed using a sliding window approach: each input sample consists of the previous 15 trading days of stock data, with five numerical features per day (Open, High, Low, Close, and Volume). The prediction target for each sample is the closing price of the next trading day.

- **Window size:** 15
- **Number of filters:** 64
- **Kernel size:** 3
- **Number of training epochs:** 15
- **Number of centers:** 50

2.2 Evaluation Metrics

To evaluate model performance, we use two metrics: Mean Absolute Error (MAE) and Fluctuation Around the Trend (FAT). MAE measures the average prediction error in absolute terms, while FAT captures how much the model's predictions deviate from the general shape or trend of the actual series.

The first metric is the Mean Absolute Error (MAE):

$$MAE = \frac{1}{N} \sum_{i=1}^{N} |t_i - o_i| \tag{1}$$

t_i = target (actual) value
o_i = predicted value

MAE measures the average absolute difference between predicted and true values. The second metric is Fluctuation Around the Trend (FAT):

$$FAT = \frac{1}{N} \sum_{i=1}^{N} |y_i - \bar{y}_i| \tag{2}$$

y_i = observed (true) value
$\bar{y}_i$ = smoothed or trend-estimated value

FAT quantifies how much a model's predictions deviate from the smoothed trend line, providing a shape-sensitive evaluation metric that captures pattern stability rather than just pointwise error. These two metrics will help us evaluate our model and compare them with one another.

2.3 Convolutional Neural Network (CNN)

For the first model, we have a 1D Convolutional Neural Network to forecast stock price movements based on short-term patterns in historical data. The model is trained using a sliding window approach, where each input consists of a 15-day sequence of five financial indicators: Open, High, Low, Close, and Volume. A consistent input-output structure is employed across the CNN, VAE, and RBF-ELM models, with each framework applying a distinct computational strategy to the same $[15, 5]$ input window.

The input tensor has shape $[N, 15, 5]$, where N is the number of sequences in a batch, 15 is the window size (days), and 5 is the number of daily features. The CNN architecture consists of a 1D convolutional layer with 32 filters and a kernel size of 5, followed by a max pooling layer with pool size 1. The output is then flattened and passed through a fully connected layer with 128 neurons (ReLU activation), and finally through a dense layer with one neuron to produce the predicted (normalized) closing price.

Each input sequence maps to a scalar output, giving the model an output shape of $[N, 1]$. During training, the model uses 1479 sequences of shape $[1479, 15, 5]$ and corresponding targets of shape $[1479, 1]$. The model is trained using the Adam optimizer and Mean Absolute Error (MAE) as the loss function. For evaluation, the model is tested on the final 100 days of the dataset, which were held out during training. These testing samples have shape $[100, 15, 5]$, and the model outputs predictions of shape $[100, 1]$, which are then rescaled back to dollar values.

2.4 Variational Autoencoder (VAE)

Now, for the Variational Autoencoder, it will learn a compressed latent representation of stock price sequences. Each input sample is a sequence of shape 15×5, representing 15 consecutive trading days with five standardized features: Open, High, Low, Close, and Volume. These features are normalized using z-score scaling across the dataset.

The encoder uses an LSTM layer to capture temporal dependencies in the input sequence. The final hidden state of the LSTM, with shape 128, is passed through two parallel fully connected layers to produce the mean vector $\boldsymbol{\mu} \in \mathbb{R}^{32}$ and log-variance vector $\log \boldsymbol{\sigma}^2 \in \mathbb{R}^{32}$. A latent vector $z \in \mathbb{R}^{32}$ is sampled from the resulting Gaussian distribution using reparameterization :

$$z = \boldsymbol{\mu} + \boldsymbol{\sigma} \odot \epsilon, \quad \epsilon \sim \mathcal{N}(0, I)$$

The decoder transforms this latent vector through two fully connected layers and outputs a scalar forecast for the closing price of the next day. The output is a single value per sequence, later rescaled back to its original unit. The VAE is trained by minimizing a composite loss function consisting of the reconstruction error and the Kullback-Leibler divergence:

$$\mathcal{L}_{\text{VAE}} = \text{MSE}(x, \hat{x}) + \beta \cdot \text{KL}\left(q(z|x) \| \mathcal{N}(0, I)\right)$$

$$\text{KL} = -\frac{1}{2} \sum_j \left(1 + \log(\sigma_j^2) - \mu_j^2 - \sigma_j^2\right)$$

To prevent premature regularization, β is progressively scaled from 0 to 1 as training proceeds. The model is optimized using the Adam optimizer on a dataset of Apple stock prices. Once trained, the model provides point forecasts for future closing prices using the learned latent representations.

2.5 Radial Basis Function with an Extreme Learning Machine (RBF-ELM)

Lastly, we have the Radial Basis Function Extreme Learning Machine. The RBF-ELM model consists of a single hidden layer where the non-linear activation is computed using Gaussian radial basis functions. The model begins by flattening each 15-day input sequence of shape 15×5 into a single feature vector of dimension 75. A random subset of training samples is selected to serve as kernel centers $C \in \mathbb{R}^{k \times 75}$, where k is the number of radial basis functions (typically $k = 100$ in our implementation). The RBF kernel between training samples X and centers C is computed using:

$$K(x, c) = \exp(-\gamma \|x - c\|^2)$$

where γ is a hyperparameter that controls the spread of the kernel. We perform a grid search over a range of γ values to select the one minimizing validation error. Once the kernel matrix Z is computed, the model solves a regularized least-squares problem:

$$\boldsymbol{\beta} = Z^{\dagger} y$$

where $Z^{\dagger}$ is the Moore-Penrose pseudoinverse of the kernel matrix and y is the target vector of normalized stock prices.

3 Results

3.1 Individual Methods

We first evaluate the three core models: CNN, VAE, and the RBF-ELM to understand how each performs on their own. Figure 1 plots each model's predictions against the actual closing prices from December 2024 to May 2025, the last 100 days of our dataset.

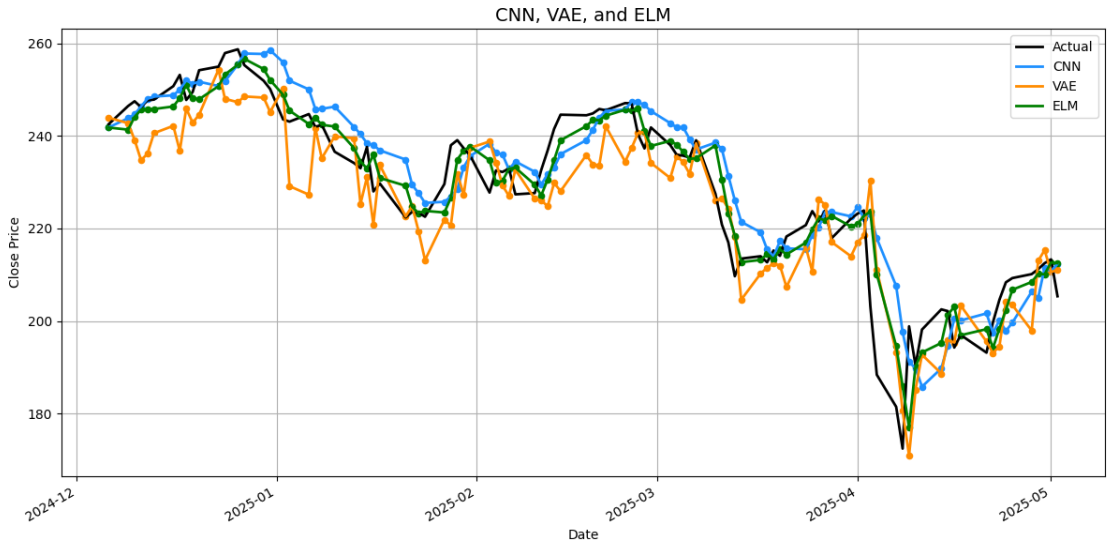

Figure 1: Predicted closing prices using standalone CNN, VAE, and RBF-ELM models.

Among the three, RBF-ELM consistently produced the most accurate forecasts. Its predictions track the direction and curvature of the actual market more closely, especially during rapid drops or rebounds. It achieved the lowest Mean Absolute Error of 1.53 and a Fluctuation Around the Trend score of 14.79, making it the strongest overall performer. The CNN model performed moderately well, identifying general trends but struggling during sharp transitions. While its curve follows the overall path of the actual series, it frequently lags or flattens sudden movements, leading to an MAE of 4.49. The VAE model was the least stable among the three. Although it captured some turning points, it showed frequent jagged predictions. This resulted in more pronounced deviation from actual prices, especially during periods of market instability, with an MAE of 6.29 and the highest FAT of 14.84. Though they are all performing differently, they are all three following the general structure of the actual close prices well.

3.2 Combination Methods

Now, to explore whether combining models could improve performance, we tested three combination's of architectures: CNN+VAE, VAE+ELM, and CNN+ELM in Figure 2.

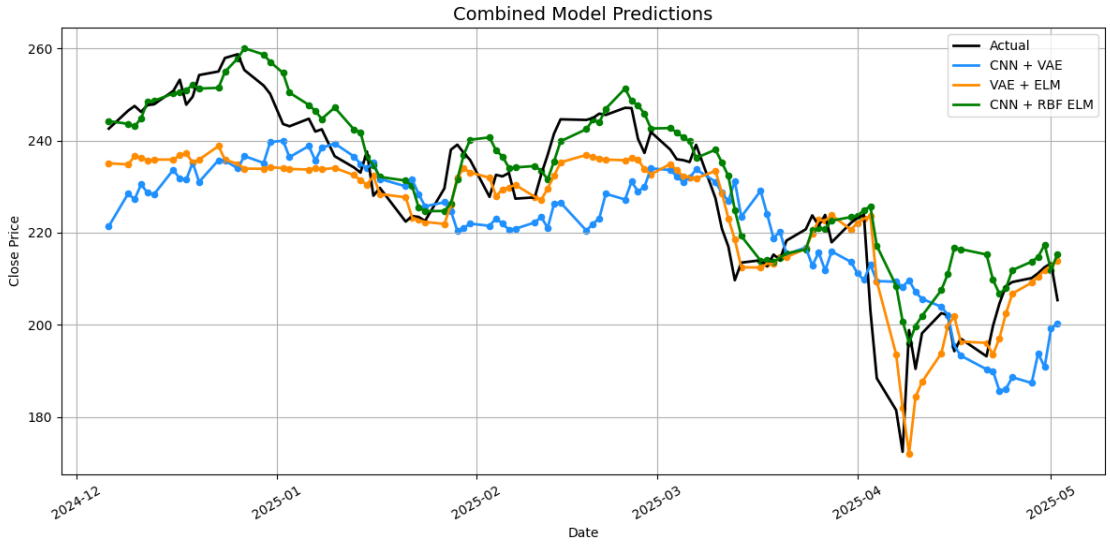

Figure 2: Predicted closing prices using combined models compared with standalone RBF-ELM.

Among the combined models, VAE+ELM and CNN+ELM both showed some improvements in capturing the trend shape compared to VAE or CNN alone. However, neither surpassed the performance of RBF-ELM. The VAE+ELM had a MAE of 5.74 and a relatively low FAT of 10.94, while CNN+ELM slightly outperformed it with a MAE of 5.54 and FAT of 11.39. CNN+VAE, on the other hand, underperformed both other hybrid models and its own components. It failed to adapt to trend reversals and had an MAE of 13.89, nearly three times higher than VAE+ELM. In contrast, the CNN+RBF-ELM performed the best in this group which we think has to do with the RBF-ELM learning well from the CNN's feature selection.

3.3 All Three Combined

We also tested a full pipeline combining all three model types: CNN for feature extraction, VAE for latent encoding, and RBF-ELM for final prediction. Figure 3 shows the model's predictions against actual stock prices.

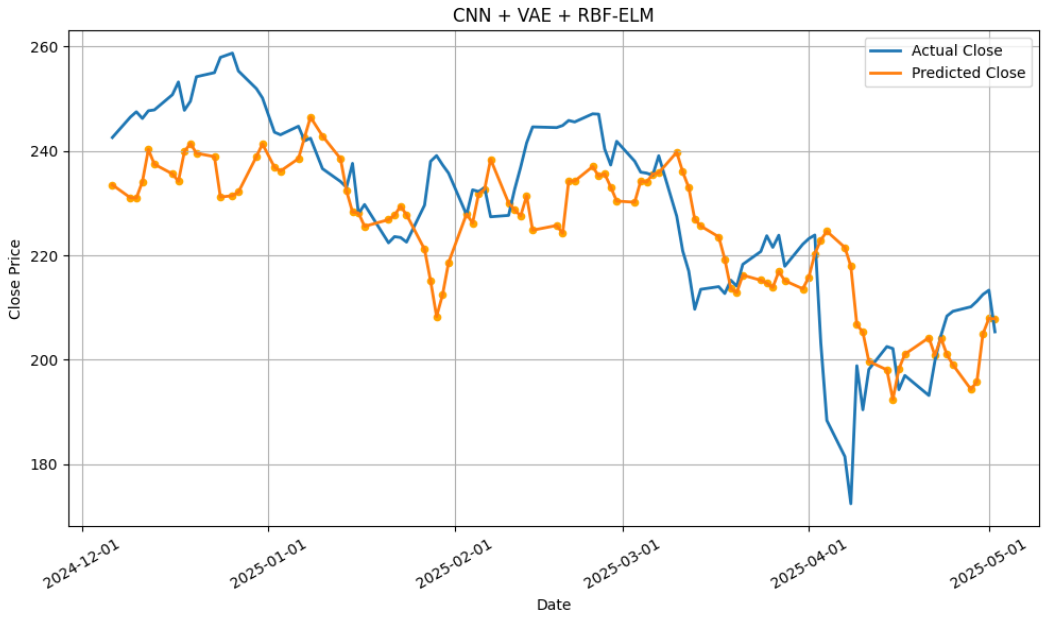

Figure 3: Predicted versus actual prices using the CNN + VAE + RBF-ELM.

Although this architecture was the most complex, it produced the least accurate results overall. The predicted line consistently lagged behind the true prices, especially during sharp reversals and high-volatility periods. The model failed to capture turning points and underestimated the magnitude of upward and downward swings. This model achieved a Mean Absolute Error (MAE) of 10.60 and a Fluctuation Around the Trend (FAT) of 11.50, both significantly worse than simpler combinations such as CNN+ELM or VAE+ELM. We think that this is because of the interaction between deep feature extraction (CNN), probabilistic encoding (VAE), and a shallow learner (ELM) may introduce misalignment, but it is unclear as of right now. While each component has its strengths independently, the combined architecture struggled to coordinate learning effectively across stages. It is important to note that this model could be improved to produce more accurate results, with enhancements to parameters such as increasing epoch, number of centers, or playing around with the gamma value.

3.4 Autoregressive Future Prediction

To evaluate how well the models generalize to unseen data, we tested the RBF-ELM in an autoregressive setting. This model was chosen because of how well it was performing above. Instead of predicting just one day ahead, the model used its own previous forecasts as input to recursively predict the next 30 days. This type of forecasting better replicates real-world applications where ground truth data is not always available and when we do not know how the stock market is going to move. Figure 4 shows the model's performance from May 5, 2025 to June 4, 2025.

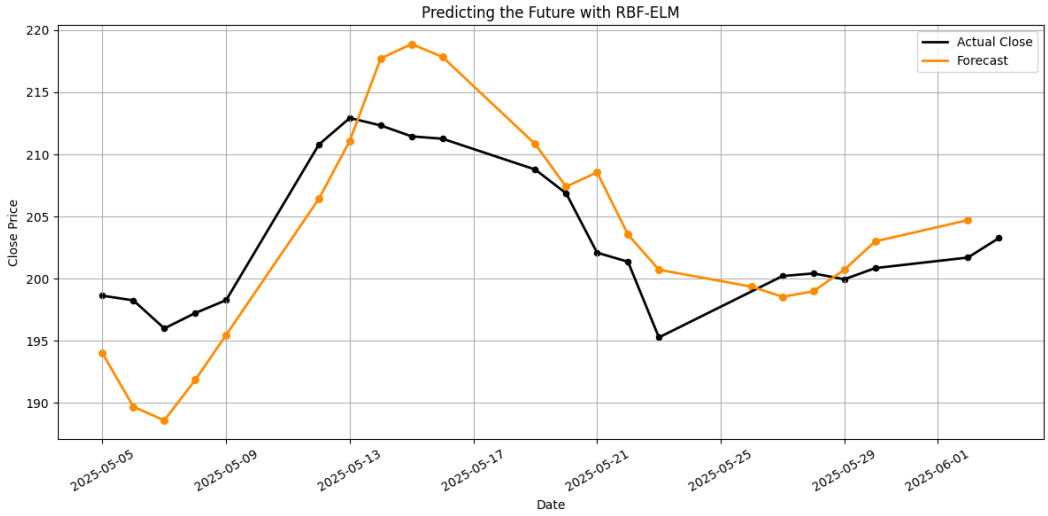

Figure 4: Autoregressive forecast with RBF-ELM for May 5 – June 4, 2025.

While the forecast deviated from the actual values, it was able to closely follow the general shape of the trend. This ability to preserve trajectory is captured by a relatively low Fluctuation Around the Trend (FAT) score of 7.11. The Mean Absolute Error (MAE) increased to 3.70, which is expected given the uncertainty introduced by recursive predictions. However, this still is a strong result considering that the model had no access to ground truth during this window.

3.5 Extended Application: S&P 500

To test how our model architecture generalizes beyond the AAPL dataset, we applied the CNN + RBF-ELM pipeline to a much longer time series: the historical S&P 500 index, spanning from January 1, 1928 to June 4, 2025, predicting the last 1000 days of the dataset. This dataset included nearly a century of market cycles, from the Great Depression through the COVID-19 crash, so it will be interesting to observe.

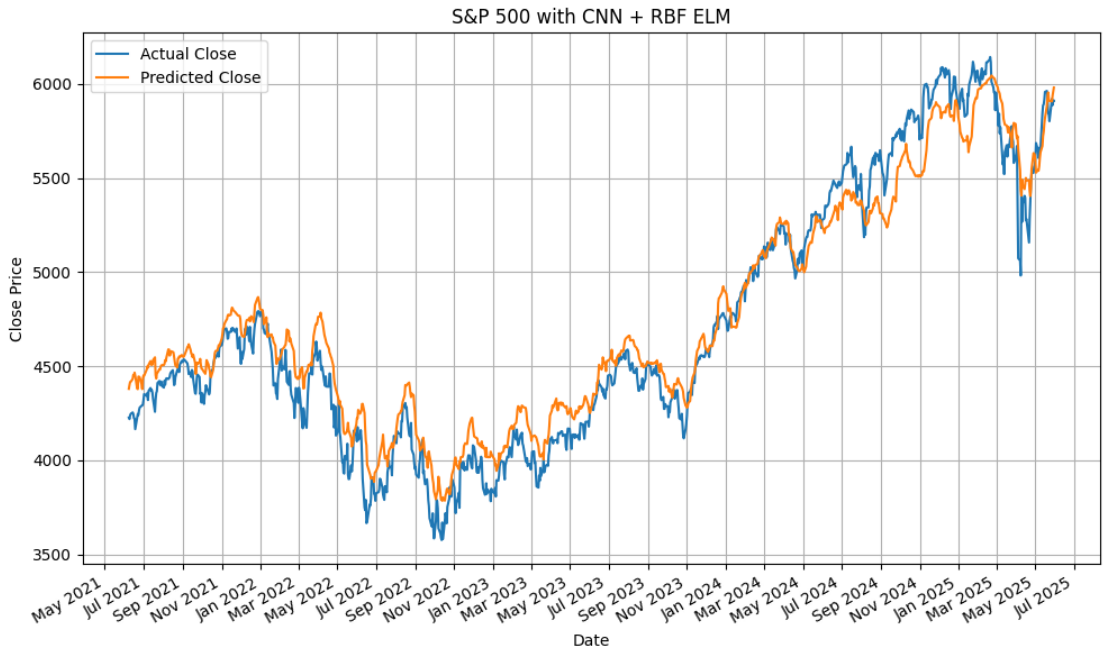

Figure 5: S&P 500 closing prices with CNN + RBF-ELM model predictions (1928–2025).

In Figure 5, the predicted close, learning from data since 1928, follows the actual close very well. Something that is different about this model is that the parameters are very different from our other models. We used 150 centers and our gamma was set to 1.0 instead of 0.1. These adjustments were made because they were better suited for the larger data set and larger output window. It seems to capture the jumps and falls accurately. With more data, the model works even better, which is not surprising. The CNN + RBF-ELM was chosen due to it being the best performing model, making it the best candidate to observe the S&P 500 index.

3.6 Comparison of All Models

To summarize the performance of all tested models, Table 1 lists the Mean Absolute Error (MAE) and Fluctuation Around the Trend (FAT) for each architecture.

Model	MAE	FAT
RBF ELM	**1.53**	14.79
RBF ELM (future prediction)	3.70	**7.11**
CNN	4.49	13.79
CNN + ELM	5.54	11.39
VAE + ELM	5.74	10.94
VAE	6.29	14.84
CNN + VAE + ELM	10.60	11.50
CNN + VAE	13.89	12.41
S&P 500	324.89	403.83

Table 1: Comparison of model performance across Mean Absolute Error (MAE) and Fluctuation Around the Trend (FAT). Lower values indicate better accuracy and stability.

It is important to note that the S&P 500 model operates on a different scale than the AAPL model, which explains the large differences in MAE and FAT values. In our experiments, the RBF-ELM performed best overall, with the standalone CNN model coming in a close second, both benefiting from their ability to directly learn predictive patterns. In contrast, the CNN+VAE model underperformed, whis is most likely due to the added complexity of compressing already informative CNN features through a generative VAE bottleneck. Saying this, it does has room to perform better with further improvement of code.

4 Conclusion, Limitations, and Future Work

In this project, we explored the how the deep learning architectures, CNN, VAE, RBF-ELM, and their combinations, for forecasting stock prices. Using a 15-day input windows of historical financial data, we evaluated each model's performance on both AAPL and S&P 500 datasets. Our results show that the RBF-ELM model is the most accurate in predicting stock market trends. The CNN model also performed well on its own, surprisingly capturing short-term temporal patterns very well for time series data, as CNN's are most commonly used for image processing. On the other hand, the CNN+VAE combination gave us the weakest results, which is surprising. It is possible that the CNN+VAE and the CNN+VAE+RBF-ELM model can be improved with further edits to the code.

While our models showed good results in terms of accuracy and trend tracking, there are several important limitations to note. First, our models are not made to handle unexpected market shocks, such as sudden policy shifts or geopolitical events. These are inherently unpredictable and fall outside the scope of time series patterns derived from historical price data. Secondly, our approach relied exclusively on stock price and volume data, without incorporating external signals such as macroeconomic indicators. This means the models are largely learning on past daya and cannot gain new information outside of the data. In addition to this, most of these models have plenty of room for improvement in parameters and methodology, such as the CNN+VAE model.

Looking ahead, there are multiple ways this work could be extended. Future work could explore the integration of external datasets, such as news sentiment, earnings reports, or

global market indicators. It could be interesting to explore these results with a longer time-series to see if more data affects the output of prediction. In addition to this, more future research could be applied to autoregressive techniques, specifically applying all of our models and seeing how well they can predict without seeing the data. Finally, while our focus was on hybrid deep learning models, comparing our results against traditional forecasting methods like ARIMA, LSTM, or Prophet would offer useful benchmarks and insight into what would perform better for the data that we have.

5 GitHub

The python code for this paper is available on GitHub here.

6 References

Hemant, H., Parida, A. K., Kumari, R., Singh, A. R., Bandyopadhyay, A., & Swain, S. (2023). *Stock market prediction under a deep learning approach using Variational Autoencoder and kernel extreme learning machine.* In 2023 21st OITS International Conference on Information Technology (OCIT) (pp. 156–161). IEEE. https://ieeexplore.ieee.org/document/10430718

Angelini, N., Bormetti, G., Marmi, S., & Nardini, F. (2013) *Value matters: Predictability of Stock Index Returns.* https://doi.org/10.48550/arXiv.1204.5055

Azari, A. (2019). *Bitcoin Price Prediction: An ARIMA Approach.* https://doi.org/10.48550/arXiv.1904.05315

Sarkar, A. & Vadivu, G. (2025). *An Advanced Ensemble Deep Learning Framework for Stock Price Prediction Using VAE, Transformer, and LSTM Model.* https://doi.org/10.48550/arXiv.2503.22192