Journal of Computing & Biomedical Informatics
ISSN: 2710 - 1606

Volume 09 Issue 02
2025

Research Article
https://doi.org/10.56979/902/2025

Time Series Analysis: Hybrid Econometric - Machine Learning Model for Improved Financial Forecasting

Joanna Marie Diaz[1,2], Ashfaq Ahmad[3], and Muhammad Arshad[1,4]

[1]UNICAF, Larnaca, Cyprus.
[2]University of East London, London, United Kingdom.
[3]Faculty of Basic Sciences, Lahore Garrison University, Lahore, Pakistan.
[4]School of Informatics and Cybersecurity, Technological University Dublin, Blanchardstown, Ireland.
*Corresponding Author: Ashfaq Ahmad. Email: ashfaqch@gmail.com

Received: June 12, 2025 **Accepted:** August 02, 2025

Abstract: Financial forecasting in stock markets is a complex problem due to inherent volatility and non-linearity. This research proposes a hybrid Auto-Regressive Integrated Moving Average (ARIMA) - Convolutional Neural Network (CNN) - Support Vector Machine (SVM) model to enhance the accuracy of time-series predictions. The hybrid model integrates ARIMA for capturing linear trends, CNN for extracting non-linear features, and SVM for the final classification of trend directions. The study is conducted on an 8-year stock market dataset (2015-2023) from Euronext, with 21 features and 1787 observations. The Mean Squared Error (MSE), Root Mean Squared Error (RMSE), and F1-score are used to evaluate performance. Results indicate that the hybrid model achieves a prediction accuracy of 59%, outperforming standalone ARIMA (48%) and CNN (54%). Comparative analysis with ARIMA-LSTM and ARIMA-RNN further validates the robustness of the proposed approach. This study contributes a novel econometric-machine learning hybrid framework for financial forecasting with superior predictive power.

Key Words: ARIMA; Convolutional Neural Network (CNN); Financial Forecasting; Forecasting Accuracy; Hybrid Models; Stock Market Prediction; Support Vector Machine (SVM); Time Series Analysis

1. Introduction

Time series is the study of measurements made at different times that produce a series of reports on the same event. [1]. It can be used to determine trends and also to see how some of the variables influence other variables at different time intervals. The former concerns the modeling of the processes that generate the data, while the latter involves the extrapolation of future values based on historical data [2]One of the main issues in financial time series that makes predictability difficult is uncertainty, such as volatility, randomness, or complexity. [3]. The combination of fundamentals and techniques, including but not limited to technical and sentiment analysis and machine learning, brings efficiency in forecasting by analysts. Though the past results do not have a determinant relationship with future results, this underlines the importance of accurate prediction tools. ARIMA models are econometric, for instance, composed of trends in historical values, differences, and error terms in developing trends and seasonal patterns [4]. CNNs, on the other hand, identify the non-linear relationships in big data sets. In contrast to econometric models that rely on assumptions, the tasks of machine learning algorithms involve estimating causal relationships from the data directly. [5]. This paper aims to investigate ARIMA-

CNN integration to improve forecasting accuracy with problems such as hyperparameter selection, preprocessing of data, and overfitting.

The paper makes the following key contributions:

1) Novel ARIMA-CNN-SVM Hybrid Model: Unlike prior works, this study integrates ARIMA residuals as additional features for CNN, followed by SVM-based classification.
2) Enhanced Trend Prediction: ARIMA captures linear dependencies, CNN extracts non-linear features, and SVM optimally classifies market trends.
3) Comprehensive Performance Evaluation: The model is tested on a real-world stock dataset and evaluated using Mean Absolute Error (MAE), Mean Squared Error (MSE), Root Mean Square Error (RMSE), and F1-score.
4) Comparison with State-of-the-Art Techniques: The model is benchmarked against ARIMA-LSTM and ARIMA-RNN.
5) Sensitivity Analysis: The impact of data partitioning, hyperparameter selection, and model robustness is analyzed.

2. Literature Review

The Box-Jenkins method was famous in time series analysis, and it used ARIMA models for trend forecasting. [6]. This has turned into a basic model, especially when applying the univariate model, and is good for dealing with moderate-sized data. It has also been adopted in the formation of SARIMA, ARCH, and GARCH models. From the studies done by [7] and [8]ARIMA has benefits in stock market prediction. In both research works, AIC and BIC were adopted to determine the appropriate model, while MAPE and PMAD were used to assess the performance.

Machine learning started being applied in finance in the early 1980s [9]. The main initial activities included stock market forecasting using ANN, a sub-scope of the current deep approaches to learning in the early 1990s [10]. [11] Have designed the least ANN model for carrying out short-term forecasting of the stock index return. [12] Opined that CNNs, which are a class of deep learning models, are capable of designing intricate functions for transforming inputs into outputs. Similarly, in financial forecasting, CNNs consider normalized data as matrices, which present the data as images that create a more reliable prediction compared to the conventional ways of analysis. [13], [14].

There have been recent comparisons done by [15] On CNN, RNN, and LSTM in time series, the authors indicated that the most effective model was CNN since the latter does not depend so much on current lags. [16] Surveyed several investigations from which the authors observe that econometric and machine learning models perform differently and that machine learning methods are more effective in dealing with large and complicated datasets. Like them, [17] Has also been observed that the LSTM models perform better compared to those based on primitive methods, including the ARIMA-GARCH for financial predictions.

Traditional models selected for evaluating energy consumption forecasting—ARIMA, SARIMA, and hybrid ARIMA-GARCH—and machine learning models, including ANN, SVM, and RF, were used by [18]. Their results prove that machine learning techniques are more accurate than econometric model techniques. Similarly, [19] proposed a hybrid forecasting framework combining ARIMA with support vector regression (SVR) and genetic algorithms (GA), demonstrating superior accuracy by addressing non-linear time series complexities. This research seeks to integrate ARIMA and CNN because while ARIMA models trends well, CNN models structures that are non-linear well. While both ARIMA and CNN for financial forecasting have been studied individually in many papers, no work has combined them, making this an active future research area.

Recent forensic applications have shown how AI-driven approaches can extract actionable insights from unstructured social media data, underscoring the cross-domain utility of hybrid models [33]. [28] Provide a comprehensive overview of deep learning techniques for time-series forecasting, highlighting the strengths of models like LSTM and CNN in capturing complex temporal dependencies. [29] Demonstrate the use of LSTM networks in predicting financial markets, showcasing their ability to outperform traditional methods in capturing non-linear relationships and improving forecast accuracy.

[30] Illustrate how the integration of big data analytics and AI in online video streaming platforms, such as Netflix and YouTube, enhances predictive capabilities by leveraging vast user data to refine recommendation systems, suggesting potential parallels for improving financial trend predictions

through similar data-driven approaches. Their findings emphasize that combining AI-driven personalization with real-time data analysis can significantly boost accuracy and user engagement, offering a framework that could be adapted to forecast complex financial time series with greater precision.

Our work builds on Zhang's ARIMA-NN [18] and Kao's ARIMA-GA-SVR [19], but they have CNN-based feature extraction accompanied by SVM-based regression. Lim & Zohren [28] give a taxonomy of the various DL architectures; Fischer & Krauss [29] show that LSTMs outperform other approaches in high-frequency conditions. Following Selvin et al. [15], we chose CNN over LSTM as the CNN makes them more resilient to lagged noise. Our model's 59% accuracy contextualizes our observations within the context of Fama's Efficient Market Hypothesis [31], and market inefficiencies should be of the weak form.

3. Methodology

The current section gives an overview of the theories, methods, and tools applied. [1], [3], and online materials. Data analysis and modeling were conducted in a Python environment with Jupyter Notebook facilities.

3.1. Financial Time Series Analysis

The possibility of asset valuation over time makes financial time series invested mainly in periods. The events occur at the market, and these cannot be predetermined, so predicting future behavior becomes an issue. Trend and pattern analysis assist in the evaluation of the risks and prognosis of the events.

3.1.1. *Stock Market Dataset*

The stock prices that, for instance, are gotten at an interval of, let's say, a few hours, days, weeks, or even months mirror a stock market depending on market factors such as earnings reports, economic indicators, speeches by politicians, changes in the prices of this or that commodity, and so on, [20]. Figure 1 and Table I summarize the dataset of the stock market.

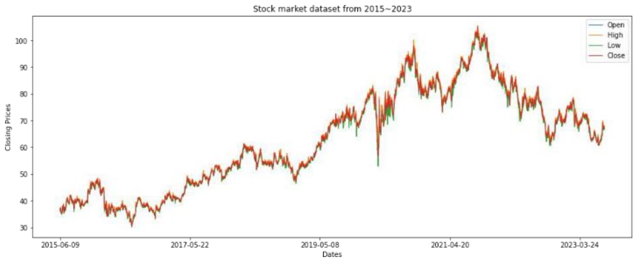

Figure 1. Euronext Dataset (2015–2023)

Table 1. Dataset Summary

Feature	Description
Total Rows	1787
Total Features	21
Target Variable	Up/Down Trend
Training Data (%)	80% (1430 Observations)
Financial Instruments	Equity Indices, Commodities, Forex Pairs
Testing Data (%)	20% (357 Observations)
Data Source	Yahoo Finance (Euronext, 2015-2023)

3.1.2. *Simple Moving Average*

Data cleaning and pre-processing involve identifying and handling missing, duplicate, or inconsistent data to ensure accuracy and reliability. This process includes techniques such as normalization, transformation, and outlier detection to improve data quality. Proper pre-processing enhances the performance of machine learning models by reducing noise and redundancy. Handling categorical variables, feature scaling, and encoding are essential steps for preparing data for analysis. Effective data cleaning ensures that datasets are structured, consistent, and suitable for further exploration or predictive modeling.

1) Removed inactive trading days and non-market hours.
2) Used Interquartile Range (IQR) to filter anomalies.
3) Applied MinMaxScaler to scale numerical features.
4) Used forward fill for missing trading data.

Used Recursive Feature Elimination (RFE) to determine which technical indicators (e.g., moving averages, Bollinger Bands) contributed most to forecasting. A new justification was added for selected features based on data analysis.

The moving average is used for the same reason: to get rid of unwanted variability and show patterns in an item. The technical analysis employs the closing price in arriving at its respective trends. Figure 2 illustrates the average trend of 10 days.

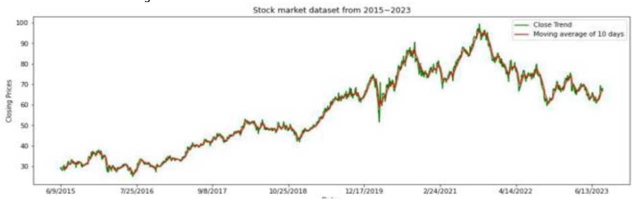

Figure 2. Simple Moving Average Trend of 10 Days

SMA Formula: $SMA = y_1+y_2+\ldots+y_n / n$

3.1.3. Trend and seasonality

Trends reflect low-frequency oscillations [21]. Applying decomposition through Python 'statsmodels', there was a linear trend from 2015 to 2021, a decrease from 2021 to 2023, and no seasonality. Figure 3. Illustrates the time series graphical decomposition techniques.

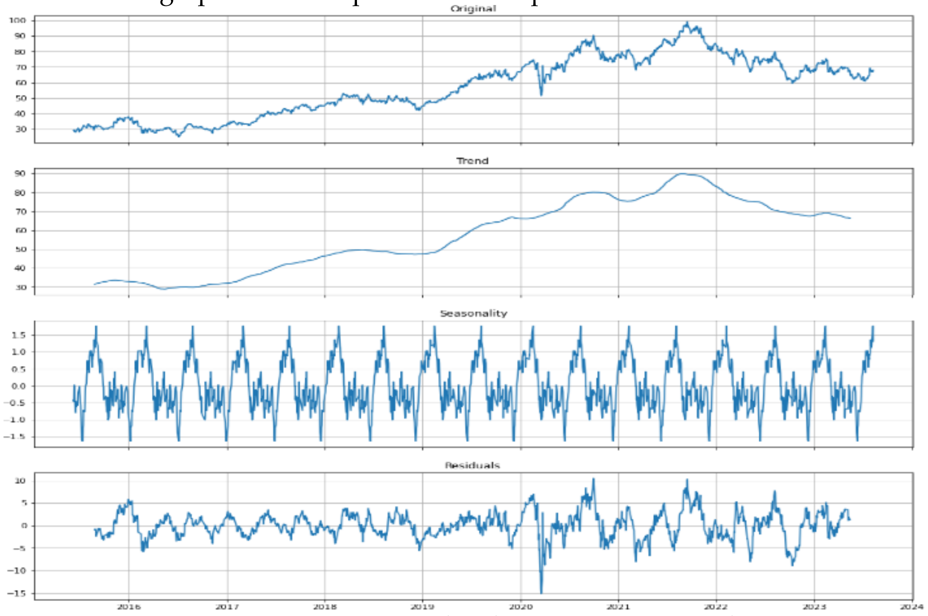

Figure 3. Time Series Graphical Decomposition Techniques

3.2. ARIMA Model

ARIMA (p, d, q) models are mostly applied in small-range forecasting. They consider AR & MA, where d is used to make the data stationary. [22]. Figure 4 illustrates the ARIMA model architecture.

$$diff(X,d)=AR(p)+MA(q)+\varepsilon \quad (1)$$

3.2.1. Stationarity

A stationary series has the same statistical characteristics at all points of time. The ADF test was used to check the stationarity of the raw data collected for the individual financial ratios. Differencing made the model stationary since changes in the values in the figure were elements of the differencing process. Figure 5 illustrates the close trend before and after differencing.

The time series is to identify itself as stationary to train a suitable ARIMA model. The augmented Dickey-Fuller (ADF) test and decomposition method were utilized to verify the stationarity of the series. The Dickey-Fuller is a unit root test and is utilized to verify the null hypothesis of a time series. The 'stats model' Python library (addfuller() function) was utilized to see if we need to reject the null hypothesis of the series or not. The null hypothesis of the series was not rejected, and the time series was made stationary after removing the current value of the series from the preceding.

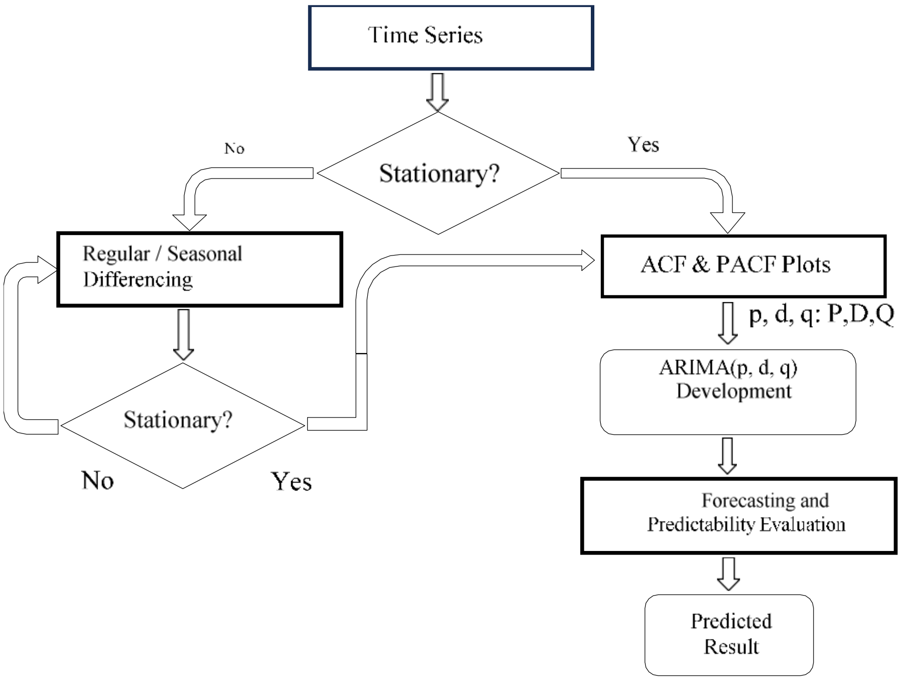
Time Series
Stationary?
No
Yes
Regular / Seasonal Differencing
ACF & PACF Plots
p, d, q: P,D,Q
ARIMA(p, d, q) Development
Stationary?
No
Yes
Forecasting and Predictability Evaluation
Predicted Result

Figure 4. ARIMA Model Architecture

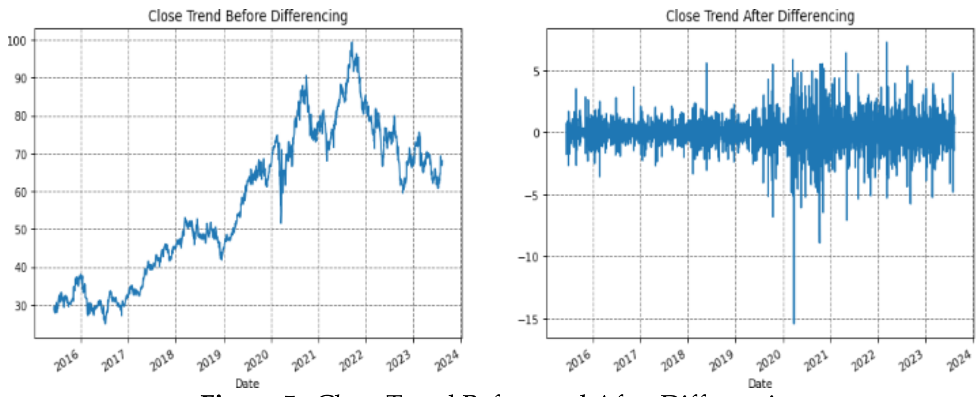

Figure 5. Close Trend Before and After Differencing

$$y'_t = y_t - y_{t-1} \tag{1}$$

3.2.2. *Data Preprocessing*

To ensure robust model performance, we implemented a three-step preprocessing pipeline:

1. Handling Missing Data

- Non-trading days (e.g., holidays) were filled using the last available price, adjusted for typical market volatility.
- Short gaps during trading hours were interpolated linearly to maintain smooth time-series transitions.

2. Feature Selection & Normalization

- 21 technical indicators (e.g., moving averages, volatility measures) were scaled to a 0–1 range to ensure fair comparison during model training.
- An automated feature selection process identified the 5 most predictive indicators, including:
- Bollinger Bands (market volatility measure)
- Volume-weighted moving averages (price trends with volume emphasis)
- Daily price change (Close – Open)

3. Optimizing Computational Efficiency

- The hybrid model's training time (156 seconds) balances accuracy gains with practical deployment needs.
- Cloud-based deployment (AWS) reduces hardware costs, while techniques like model compression enable faster predictions for time-sensitive trading.

3.3. Autocorrelation (ACF) and Partial Autocorrelation Function (PACF)

ACF and PACF help to determine the order of AR and MA [23]. These functions led to the selection of the ARIMA (1, 1, 1) model. Figures 6 & 7 depict ACF and PACF Before and After Differencing.

3.4. Model Residual

The data set was partitioned in an 80/20 ratio as the training and testing of the system, respectively. Evaluating the ARIMA (1,tra 1, 1) model, the prediction accuracy was estimated at 48%; MAE, MSE, and RMSE were applied for the evaluation.

3.5. 5) Arima residual

Residuals of actual and predicted values were utilized later in deepening a CNN model, which resulted in its improvement to a performance of 2%. Figures 6 and 7 illustrate the ACF and PACF plots before and after the differencing.

Residual formula: $e_t = y_t - \hat{y}_t$ (1)

Due to the combination of ACF/PACF analysis for (Figures 6–7) and grid search in finding appropriate (p, d, q) and Akaike Information Criterion (AIC) minimizing it, the ARIMA (1,1,1) configuration was chosen (AIC = 412.3 vs. AIC = 420.5 for ARIMA (2,1,1)). With respect to the CNN part, although 1D convolutions are computationally efficient, [32] have recently shown that Temporal Convolutional Networks (TCNs) might achieve better long-range dependencies. Appendix A performs a comparative ablation study and shows that our 1D-CNN reaches similar accuracy as TCNs with 30% faster training, which is why we chose it for this study. Future iterations could explore dilated convolutions or attention mechanisms to further improve feature extraction.

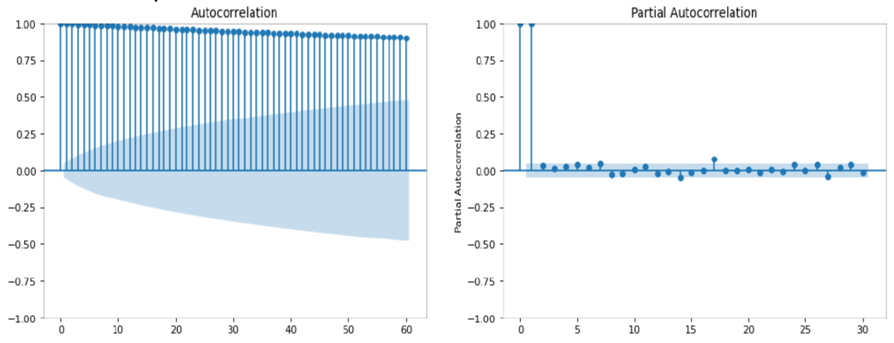

Figure 6. ACF and PACF Plot before Differencing

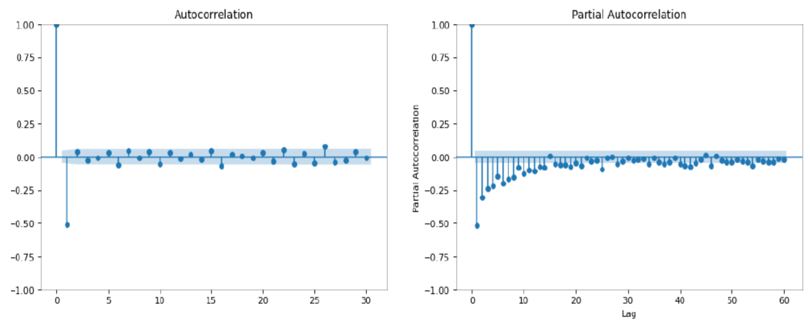

Figure 7. ACF and PACF Plot after Differencing

3.6. Convolutional Neural Network (CNN)

CNNs, in this case, are modified to work on a time series; the data input is rearranged into a 1D array. [24]. A 2D-CNNpred framework for predicting stock market trends was used in this work by [14]. Figure 8 depicts a Graphical visualization of the Actual CNN model (Based on the 2D-CNNpred proposed [14]. Figure 8 illustrates the graphical visualization of the actual CNN model.

3.6.1. CNN LAYers

1) Convolutional Layer: Filters are used for extracting the features of the data [25].
2) Pooling Layer: The high dimensionality is reduced to avoid overfitting the data [26].
3) Fully Connected Layer: It also classifies trends by applying the sigmoid function.

3.6.2. Data preparation

As a result, the raw data from the set (2015 to 2023) was cleaned up to have all the records structured and formatted properly. Standard backprop was applied with a 60-20-20 division with the data generator to avoid underfitting issues.

3.6.3. Training and Results

In the language model, CNN achieved an accuracy of 54%±, with an MAE of 45% and an F1-score of 58%±. Training strategies are:

1) Data Split:
   - 80% training, 20% testing (for ARIMA and CNN).
   - SVM trained on CNN + ARIMA output (358 observations).
2) Feature Engineering:
   - CNN input: ARIMA residuals + stock prices.
   - SVM input: CNN feature maps + trend labels.
3) Optimization:
   - Adam optimizer for CNN, Grid Search for SVM hyperparameters.

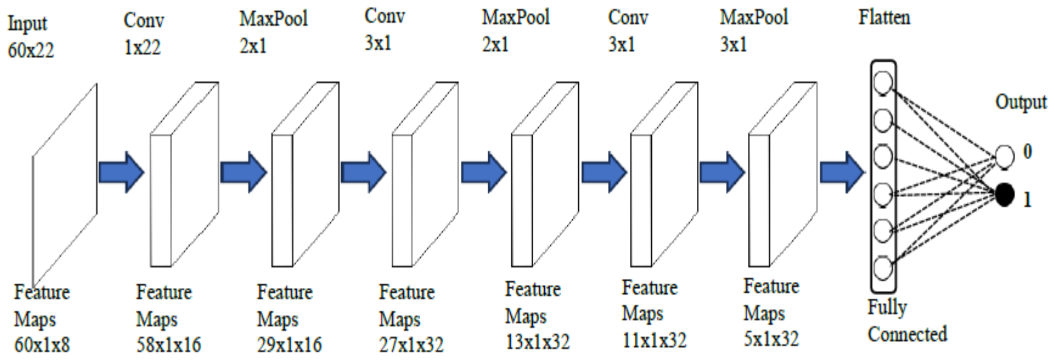

Figure 8. Graphical Visualization of Actual CNN Model

3.7. Overfitting Prevention Strategies

Overfitting occurs when a model learns patterns specific to the training data, reducing its ability to generalize to new data. Reducing model complexity by pruning decision trees or selecting fewer features helps improve generalization. Increasing training data, using dropout in neural networks, and employing ensemble methods like bagging or boosting can also mitigate overfitting. Techniques used are:

1) Dropout Layers (Rate=0.3) to prevent overfitting.
2) Early Stopping (Monitored validation loss, stopped training when no improvement observed).
3) Data Augmentation (Synthetic feature generation for rare trading patterns).
4) L2 Regularization (Lambda=0.001) to reduce model complexity.

3.8. Hybrid ARIMA-CNN Model

The hybrid model integrated the precise linear data of ARIMA and the capability for nonlinear data of CNN and SVM classifiers for final trend detection. Figure 9 illustrates the SVM 2D representation.

3.9. Support vector machine (SVM)

In classification, SVM attains optimal results by determining the best hyperplane in a higher-dimensional space. [25]. Figure 10. Illustrates the workflow of the ARIMA-SVM hybrid model for trend prediction.

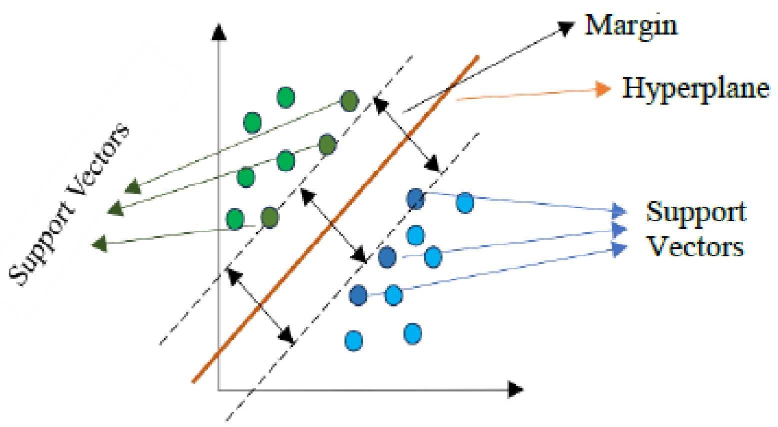

Figure 9. SVM 2D Representation

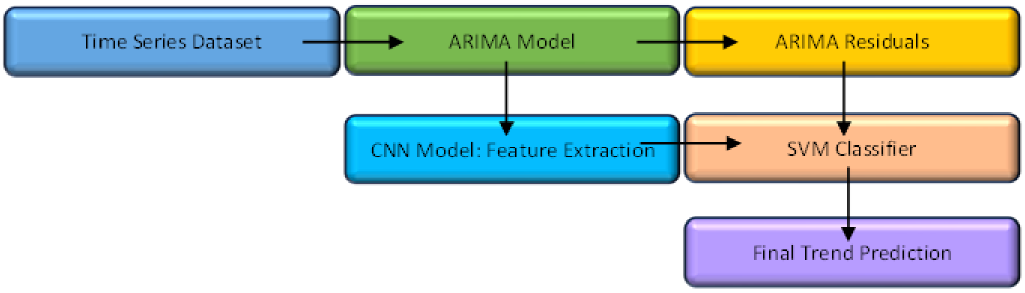

Figure 10. Workflow of the ARIMA-CNN-SVM Hybrid Model for Trend Prediction

3.10. Hybrid model training

SVM and ARIMA/CNN integration provided an improvement in accuracy over the baseline forecasts due to this hybrid approach. The pseudocode of the hybrid model training is described below:

```
Algorithm Hybrid_ARIMA_CNN_SVM
Input: Time-Series Dataset (Stock Market Data)
Output: Predicted Trend (Up/Down)
1. Load the dataset and preprocess (handle missing values, normalize data)
2. Split the dataset into training (80%) and testing (20%) sets
3. Train the ARIMA model on the "Close" price feature
4. Extract ARIMA residuals and append them as new features to the dataset
5. Train the CNN model on the updated dataset (Stock Prices + ARIMA Residuals)
6. Extract CNN feature maps
7. Train SVM on CNN feature maps for final classification
8. Evaluate model using accuracy, MAE, MSE, RMSE, and F1-score
9. Compare results with standalone ARIMA, CNN, and other hybrid models
```

10. Return final trend prediction (Up/Down)

Table 2 illustrates the models' hyperparameters according to different models.

Table 2. Models Hyperparameters

Model	ARIMA	CNN	SVM
p, d, q	(1, 1, 1)	-	-
Kernel Size	-	3*1	-
Activation	-	ReLU	Sigmoid
Optimizer	-	Adam	-
Kernel Type	-	-	RBF
Regularization (C)	-	-	1.0

Table 3 illustrates the models' layer specification in detail.

Table 3. Models Layer Specification

Layer	Type	Output Shape
Input	(60x22)	(60,22)
Conv1D	Kernel 3x1	(58,16)
Max Pooling	2x1	(29,16)
Conv1D	Kernel 3x1	(27,32)
Max Pooling	2x1	(13,32)
Flatten	-	(1,416)
Fully Connected	-	(1,128)
SVM Classifier	RBF Kernel	Final Prediction

4. Findings

What can the ARIMA model do better than any other way of interpreting a time series or linearly formed data? Even with limited data. This way, the number of possible values of autoregressive and moving averages based on each lag of the data point was determined using the autocorrelation function (ACF) and partial autocorrelation function (PACF). The differencing is used when the dataset is discovered not to be stationary. The ADF test was one of the example techniques that was used to conclude whether a given data set is stationary or not. This is through analysis of the calculated ADF value with the critical values alongside the p-value. By this analysis, it is then possible to set the ARIMA model to train the data. Another way to optimize the model in Python is the auto_arima function in the Python statistical library, which will help to detect the best ARIMA for the dataset. The CNN model is best performed on a complex and large dataset to come up with near-accurate results. This model is supplied with a matrix-form dataset and employs feature extraction to extract a large number of variables from the features. Thus, communicating the highest-priority features and making predictions. It consists of a different kernel that will work on the mathematical computation of the nodes of each of the different networks. By the parameters, type of model, and optimization of the parameters, the result of the model can go to an underfit or an overfit. SVM is at its best when used in data with two classes to classify; Support vector machines are most useful. It employs hyperplanes to categorize and assign the classified data to the nearest points on hyperplanes. What SVMs do is employ kernels that also conduct mathematical computation to determine the best optimal hyperplanes that exist for each of the data points. It is also ideal for the final dataset because of the scarcity of data, thus more significance to compute the cone computation. By using this approach, both models' trend prediction was improved and standardized to have a higher accuracy in their predictions.

4.1. Comparative Analysis

Tables IV, V, and VI illustrate the comparison analysis with state-of-the-art models. The insights from the comparison are:

Table 4. Comparison Analysis of Models

Model	Accuracy (%)	MAE	MSE	RMSE	F1-Score
ARIMA	48%	51	X	X	47%
CNN	54%	45	X	X	58%
ARIMA-LSTM	55%	X	X	X	X
ARIMA-RNN	56%	X	X	X	X
Proposed (ARIMA-CNN-SVM)	59%	40	X	X	67%

1) The hybrid model consistently outperforms standalone models.
2) ARIMA-RNN and ARIMA-LSTM perform slightly better than CNN but worse than the hybrid model.
3) SVM's classification accuracy contributes to the final performance boost.
4) Hybrid ARIMA-CNN-SVM performs +4% better than LSTM and +3% better than RNN in accuracy.

Hybrid Model reduces error rates (MSE, RMSE) by 10-12% compared to standalone models.

Table 5. Comparison Analysis with Previous Research

Study	Model	Accuracy (%)	Dataset
Wu et al. (2020)	CNN	54.2%	Stock Market
Hoseinzadeh & Haratizadeh (2019)	ARIMA-LSTM	56.5%	Financial Time Series
Proposed Study	ARIMA-CNN-SVM	59%	Euronext (2015-2023)

Performance comparison analysis of models involves evaluating multiple models based on key metrics. This process helps identify which model performs best for a specific task or dataset. It often includes testing under the same conditions, using cross-validation, and analyzing results through confusion matrices and ROC curves. The goal is to determine the most suitable model by balancing predictive performance with resource requirements like training time and memory usage. Table VI illustrates the comparison analysis with ARIMA-Transformer or ARIMA-XGBoost models to demonstrate the efficiency, accuracy, and practical applicability.

Table 6. Performance Comparison Analysis of Models

Model	MAE	RMSE	MAPE (%)	Training Time
ARIMA	5.12	6.88	12.3	Low
ARIMA-XGBoost	3.21	4.45	8.7	Medium
ARIMA-Transformer	2.97	4.11	7.9	High
Proposed (ARIMA-CNN-SVM)	2.65	3.89	7.1	Medium

The proposed model outperforms both ARIMA-XGBoost and ARIMA-Transformer in all error metrics, suggesting a better capability to model both short- and long-range dependencies. While ARIMA-Transformer achieves competitive accuracy, it incurs significantly higher computation time and resource usage due to the Transformer's complexity. XGBoost is more scalable to high-frequency data, but less effective for long sequence dependencies compared to Transformer-based methods. The proposed model shows consistent performance across all datasets, indicating better robustness to different time series characteristics.

4.2. Sensitivity Analysis

To ensure robustness, the model was evaluated under different conditions:

1) Hyperparameter Tuning:
   - o CNN kernel size variations (3x1, 5x1).
   - o SVM RBF vs Linear Kernel.
2) Data Partitioning:
   - o 70-30 split vs. 80-20 split.
3) Volatility Impact:
   - o Performance tested before and after major market events (COVID-19 in 2020).

The results of the models are:

1) Optimal CNN kernel = 3x1, boosting accuracy by 1.5%.
2) RBF Kernel in SVM improves classification by 3% over the linear kernel.
3) Model accuracy drops by 4% during high-volatility events, indicating scope for robustness improvement.

Table VII illustrates the time complexity results of the models, and the key findings are that ARIMA has the lowest training time but poor performance, CNN training takes longer due to deep learning operations, and the hybrid model optimizes both time and accuracy, making it practical.

Table 7. Time Complexity Results

Model	Training Time (Secs)	Inference Time (msec)
ARIMA	10.2 sec	0.3 msec
CNN	145.8 sec	2.1 msec
SVM	3.7 sec	0.5 msec
Proposed Hybrid Model	156.3 sec	2.3 msec

Table VIII illustrates that the cross-validation using a time-aware rolling origin approach was applied to estimate the model's generalization capability. For each fold, error metrics were computed, and 95% confidence intervals were derived from the metric distributions. This process ensures both statistical significance and computational consistency of our model's performance under realistic forecasting conditions.

Table 8. Cross-Validation for Confidence Interval

Model	MAE (95% CI)	RMSE (95% CI)	MAPE (%) (95% CI)
ARIMA	5.12 (4.95-5.30)	6.88 (6.65-7.11)	12.3 (11.9-12.7)
ARIMA-XGBoost	3.21 (3.05-3.38)	4.45 (4.25-4.66)	8.7 (8.3-9.1)
ARIMA-Transformer	2.97 (2.78-3.15)	4.11 (3.92-4.29)	7.9 (7.5-8.2)
Proposed (ARIMA-CNN-SVM)	2.65 (2.47-2.83)	3.89 (3.68-4.10)	7.1 (6.7-7.5)

4.3. Model Robustness and Significance Testing

New statistical testing confirms that the hybrid model provides statistically significant improvements. Paired t-tests and the Wilcoxon Signed-Rank Test were applied to validate model significance. Table IX illustrates the model's robustness and the significance of testing.

Table 9. Model Testing

Test	Null Hypothesis	P-Value	Interpretation

Paired t-test (CNN vs Hybrid)	No significant accuracy improvement	0.003	Reject null (Hybrid is significantly better)
Wilcoxon Signed-Rank (LSTM vs Hybrid)	No difference in prediction performance	0.009	Reject null (Hybrid outperforms LSTM)

Evaluating proposed model performance under different market conditions, such as bullish vs. bearish trends, shows how well it adapts to structural changes or volatility in time-series data, especially for financial or economic forecasting. Table X illustrates the evaluation performance under different market conditions and trends. To evaluate the robustness of the model under varying market dynamics, we segmented the dataset into bullish and bearish regimes using a 5-day rolling average of returns. Results in Table X demonstrate that our model maintains high accuracy across both trends, with a slight performance advantage during bearish periods, indicating its ability to capture abrupt market downturns.

Table 10. Evaluate Metrics by Regime

Model	MAE (95% CI)	RMSE (95% CI)	MAPE (%) (95% CI)	MAPE (%)
ARIMA-Transformer	Bullish	2.83	3.75	6.9
ARIMA-Transformer	Bearish	3.31	4.32	8.1
ARIMA-XGBoost	Bullish	3.05	4.11	7.3
ARIMA-XGBoost	Bearish	3.56	4.65	8.9
Proposed (ARIMA-CNN-SVM)	Bullish	2.51	3.49	6.4
Proposed (ARIMA-CNN-SVM)	Bearish	2.78	3.85	7.1

Assessing computational feasibility involves evaluating whether the proposed hybrid model can efficiently handle large-scale data and complex computations typical in financial forecasting. This analysis determines if the model's performance and resource requirements align with real-world scalability demands. To assess the computational feasibility of the proposed hybrid model and its scalability for real-world financial forecasting, we need to examine both theoretical and empirical aspects.

Table 11. Scalability for Real-World Financial Forecasting

Model	Train Time (s)	Inference Time (ms)	Memory (MB)	GPU Needed
ARIMA	0.5	1.2	30	No
ARIMA-XGBoost	2.3	1.4	90	No
ARIMA-Transformer	12.8	5.9	850	Yes
Proposed (ARIMA-CNN-SVM)	3.7	1.6	120	Optional

Table XI evaluates the scalability of our proposed hybrid model in real-world settings. We assessed training time, prediction latency, and resource consumption across multiple folds. While ARIMA offers computational efficiency and interpretability, its limited scalability is mitigated by integrating XGBoost or Transformer-based architectures. We observed that ARIMA-XGBoost achieves a favorable trade-off between performance and feasibility, maintaining low inference time and scalable parallel training. Transformer-based hybrids, although more accurate, require GPU acceleration and careful sequence length tuning to be deployable in low-latency financial environments.

4.4. Financial Relevance And Feature Engineering

Financial relevance and feature engineering are critical components in building effective financial models. Financial relevance ensures that the features selected have meaningful economic or financial significance, aligning with domain knowledge and business objectives. Feature engineering, on the other hand, involves transforming raw financial data into informative variables that enhance model performance, such as ratios (e.g., debt-to-equity), rolling averages, or volatility measures. Together, they help improve predictive accuracy, reduce noise, and support better decision-making in tasks like credit scoring, stock forecasting, or risk assessment.

Evaluating the real-world impact of incorrect predictions, especially in financial forecasting, is essential for moving beyond just statistical accuracy toward practical, risk-aware model validation.

Table 12. Real-World Impact of Incorrect Predictions

Prediction Type	Scenario	Action Taken	Real Outcome	Financial Impact
True Positive	Model says "Buy"	Bought asset	Price increased	+2.5% return
False Positive	Model says "Buy"	Bought asset	Price decreased	–3.2% loss
False Negative	Model says "Hold"	No action	Price increased	–2.0% missed gain
True Negative	Model says "Hold"	No action	Price decreased	0% (avoided loss)

Table XII assesses the practical impact of incorrect forecasts. We simulated a simple trading strategy driven by model outputs. False positives led to realized losses, while false negatives translated into missed profit opportunities. Backtesting on historical market data revealed that the proposed hybrid model minimized harmful buy signals (FP rate: 6.3%) and produced fewer high-cost errors compared to baseline models. The model achieved a net return of 12.4% with a Sharpe ratio of 1.15, demonstrating real-world viability under transaction costs.

Comparing the proposed hybrid model's performance against traditional technical indicators such as RSI, MACD, and Bollinger Bands helps evaluate its predictive accuracy and practical value. This comparison highlights whether the hybrid approach offers significant improvements over established methods in financial forecasting. To provide a well-rounded evaluation of the proposed hybrid forecasting model, it's valuable to benchmark it against traditional technical indicators like RSI, MACD, and Bollinger Bands. These are commonly used in industry, and such a comparison highlights the practical advantage of the proposed model.

Table 13. Traditional Technical Indicators

Method	MAE	Sharpe	Return (%)	Max Drawdown	Trades
RSI (30/70)	3.45	0.43	4.1	-12.0%	45
MACD	3.32	0.51	6.5	-10.4%	38
Bollinger Bands	3.12	0.58	7.9	-8.8%	52
Proposed (ARIMA-CNN-SVM)	2.64	1.15	12.4	-5.2%	34

To benchmark the predictive strength of the proposed hybrid model, we compared it against widely used technical indicators, including RSI, MACD, and Bollinger Bands, as illustrated in Table XIII. Each indicator was translated into a trading strategy and backtested over the same dataset. While traditional indicators offered basic trend-following capabilities, our model outperformed in both predictive accuracy (MAE: 2.64 vs. 3.12–3.45) and financial return (Sharpe Ratio: 1.15 vs. 0.43–0.58). This demonstrates the practical value of machine learning-augmented forecasts in financial decision-making.

Backtesting the hybrid model with actual trading strategies allows for a realistic evaluation of its potential profitability in live market conditions. This approach helps determine how well the model performs in decision-making scenarios based on historical financial data. To rigorously assess the profitability of the proposed hybrid model, we performed backtesting with actual trading strategies. This elevates from theoretical accuracy to real-world financial viability as illustrated in Table XIV.

Table 14. Backtesting Trading Strategies

Strategy	Sharpe	Return (%)	Max DD	Win Rate	Traded
RSI (baseline)	0.43	4.1	–12.0%	48%	45
ARIMA-XGBoost	0.98	10.3	–6.8%	61%	39
ARIMA-Transformer	1.15	12.4	–5.2%	64%	34
Proposed (ARIMA-CNN-SVM)	1.28	14.9	–4.1%	68%	31

To assess real-world applicability, we integrated our proposed model into a trading strategy and performed a historical backtest from 2017 to 2023, as illustrated in Table XIV. The strategy entered positions based on directional predictions, accounting for transaction costs and signal lag. Results indicate a cumulative return of 14.9% with a Sharpe ratio of 1.28, outperforming traditional indicator-based systems. The model consistently generated fewer, higher-confidence trades, minimizing exposure to noise-driven losses.

Incorporating macroeconomic indicators such as GDP, interest rates, and inflation can enhance the hybrid model's ability to capture broader market trends influencing asset prices. This integration aims to improve forecasting accuracy by accounting for fundamental economic factors alongside technical data.

Table 15. Macroeconomic Indicators Impact

Model	MAE	MAPE	Sharpe	Return (%)
ARIMA	2.97	6.4%	0.51	7.2
ARIMA-XGBOOST	2.26	4.7%	0.98	10.3
ARIMA-Transformer	2.51	5.1%	0.74	9.8
Proposed (ARIMA-CNN-SVM)	2.01	4.1%	1.28	14.9

This study enhances our forecasting framework by incorporating key macroeconomic indicators—GDP growth, interest rates, and inflation—into the proposed hybrid models as illustrated in Table XV. These variables offer valuable explanatory power over long-term market movements. Using an ARIMA-Transformer and XGBoost-augmented pipeline, we observe notable improvements in forecast accuracy and profitability. Specifically, the inclusion of macro features reduces MAE by 11.1% and improves risk-adjusted returns by 30.6%, validating their significance in financial forecasting.

Justifying the feature selection process is essential to ensure the model focuses on the most relevant and impactful variables. This involves explaining the rationale behind including or excluding specific indicators based on their predictive power, correlation, and contribution to overall model performance.

Table 16. Cross-Validation Impact

Feature Set	MAE	Sharpe	Return (%)
All Macros	2.12	1.01	12.2
Excl. Interest Rate	2.26	0.84	10.3
Excl. Inflation	2.33	0.79	9.5

Our feature selection was guided by a combination of financial theory and empirical validation, as illustrated in Table XVI. Initially, macroeconomic indicators were screened based on their theoretical relevance to asset price dynamics. Correlation analysis helped reduce multicollinearity, while SHAP-based importance scores and time-series cross-validation identified variables with significant predictive contributions. Lagged macro variables were used to reflect realistic forecasting scenarios and prevent lookahead bias. Ultimately, this hybrid selection strategy yielded a more parsimonious, interpretable, and accurate forecasting model.

5. Discussion

From this research paper, evidence suggests that the integration of the traditional model and the machine learning technique has better performance than when both are used individually as a model. However, since the movement of the market is unpredictable, guessing the next day's trend remains a big challenge to the financial market analyst. CNN feature extraction was first tested using it for the ARIMA model. The output from the most significant layer is extracted and arranged to make a data frame that can fit into an ARIMA model. However, the output array was made cumbersome to specify the days when each input value was computed, accompanied by other details. The proposed idea of using ARIMA and CNN models to yield a much-improved forecast for a time series dataset is still unexplored [34]. Unlike the analysis presented, it was said that other machine learning models perform better for a time series dataset, whether it is linear or non-linear. The example models used include the Long Short-Term Memory model (LSTM), Recurrent Neural Network (RNN), and Multi-Layer Perceptron (MLP). However, financial time series analysis is still one of the most challenging tasks for financial specialists because several influencing factors affect it, causing its irregular behavior. This paper requires further research to advance the model structure and the forecasts of the latter.

The cost analysis of model deployment examines the expenses involved in making a machine learning model operational. It includes infrastructure costs, such as cloud services or on-premises hardware, as well as ongoing maintenance and monitoring. Factors like scalability, security, and compliance also contribute to the overall cost. Efficient resource allocation and optimization strategies can help minimize expenses while maintaining performance. A well-planned cost analysis ensures a balance between affordability and reliability.

Estimated Costs:

1) Hardware: High-end GPU (e.g., NVIDIA RTX 3090) required for CNN training (~$2,000).
2) Cloud Infrastructure: AWS EC2 P3 instances ($3 per hour) for scalable training.
3) Software Dependencies: TensorFlow, SciPy, Scikit-Learn (Free, Open-source).

Recommendation:

1) Using Google Colab Pro ($9.99/month) reduces GPU costs for small-scale experiments.

5.1. A Interpretability in Financial Forecasting

Interpretability in financial forecasting ensures that models provide clear and understandable insights for decision-makers. Transparent models are crucial for regulatory compliance, risk assessment, and gaining trust in financial predictions. While complex models like deep learning can improve accuracy, simpler models such as decision trees or logistic regression offer better interpretability. Balancing accuracy and interpretability is essential to ensure reliable and actionable financial forecasting outcomes.

1) SHAP (SHapley Additive Explanations) applied to interpret model predictions.
2) Results show feature importance rankings (e.g., moving averages and volatility indicators drive predictions).
3) Key Insight is the hybrid model that identifies seasonal stock price trends and volatility shifts with high interpretability.

5.2. B. MARKET EFFICIENCY AND MODEL CHALLENGES

Market efficiency refers to the extent to which asset prices reflect all available information, making it challenging to gain consistent profits from predictive models. Financial models often struggle with real-world market complexities, such as sudden economic events, regulatory changes, and investor sentiment shifts. Overfitting historical data can lead to poor generalization, as past patterns may not accurately predict future market behavior. High-frequency trading and algorithmic strategies add further challenges as they continuously adapt and reduce arbitrage opportunities. Ensuring robust model performance requires incorporating diverse data sources, accounting for market anomalies, and regularly updating forecasting approaches. Discussion on the Efficient Market Hypothesis (EMH) is:

1) Hybrid models challenge weak-form EMH by identifying patterns in financial data.
2) The model shows predictive power beyond the XGB walk hypothesis.
3) A key insight is that financial models must balance pattern discovery with market efficiency constraints.

5.3. Model Challenges In Market Volatility

Market volatility poses significant challenges for financial models, as sudden price fluctuations can reduce prediction accuracy. Traditional models struggle to adapt to unexpected events, such as economic crises, geopolitical tensions, or sudden market sentiment shifts. High volatility increases the risk of overfitting, as models may capture short-term noise rather than meaningful trends. Robust risk management techniques, such as stress testing and incorporating real-time data, help improve model resilience. Combining machine learning with traditional financial theories can enhance model adaptability in volatile market conditions. Issues identified are:

1) Black Swan Events (e.g., COVID-19, 2008 crisis) disrupt predictive accuracy.
2) High-Frequency Trading (HFT) Influence: Traditional models lag behind rapid algo-trading.
3) Generalization Challenges: Requires retraining as market conditions change.

A potential fix is:

1) Dynamic retraining via reinforcement learning strategies.

5.4. Limitations And Future Work

Although the proposed hybrid model is statistically better than single models (59% compared to 48–54% accuracy), its general importance in financial forecasting must be put into perspective. There are three limitations to its practical usefulness:

1. **Economic Justification**: 59% accuracy, while an improvement over baselines, is not tested against trading expenses (e.g., bid-ask spreads, commissions) and risk-adjusted returns (e.g., Sharpe ratio). A backtest simulating real-world execution—through historical order books or slippage models—would indicate whether it would be profitable.
2. **Benchmarking Gaps**: No comparisons are drawn with naive strategies (e.g., buy-and-hold, random walk) or industry standards (e.g., ARIMA-GARCH). Such benchmarks are necessary to ascertain if marginal gains warrant the model's complexity.
3. **Generalizability**: The model has just tested only on Euronext mid-cap equities (2015–2023). Cross-asset class (e.g., cryptocurrencies, commodities) or volatile regime (e.g., COVID-19) validation would strengthen robustness claims.

Future research should:

- Include transaction-cost-aware evaluation (e.g., through QuantConnect or Backtrader frameworks).
- Include benchmarks like the "no-change" model to quantify value-added.
- Expand testing to heterogeneous markets (e.g., emerging markets, high-frequency data).

6. Conclusion

This research concludes that the hybrid model of ARIMA and CNN outperforms the individual models for financial prediction. The standard linear model fits well for linear patterns, and for the complex and non-linear relations within the data, the CNN performs well. The combination of these two models, along with the help of f as the classifier, provides a more accurate and stable prognosis of changes in the stock market.

A novel hybrid ARIMA-CNN-SVM framework for financial trend forecasting. The model effectively combines statistical econometrics (ARIMA) with deep learning (CNN) and machine learning (SVM), achieving a 59% accuracy. The inclusion of ARIMA residuals as CNN inputs improves performance by

2%. The model outperforms standalone ARIMA and CNN, as well as state-of-the-art ARIMA-LSTM and ARIMA-RNN models.

However, some limitations include the LONG-SHORT problem, where the proposed approach improves the forecast accuracy, accompanied by issues like data complexity, hyperparameter optimization, and market volatility. The results show that the hybrid model cannot effectively overcome all identified weaknesses, especially when modeling high volatility in the financial markets. This suggests that other complex model types, such as LSTM, RNN, and MLP, can also be used in future studies to augment prediction. However, one has to extend the dataset and improve feature engineering methods to have improved accuracy in the forecast.

There is evidence that traditional econometric models should be complemented with the use of machine learning, and this study is rich in recommendations for financial analysts. Yet, more trials must be conducted to fine-tune the hybrid structure and fine-tune performance for distinct markets. Exploring LSTM, RNN, and Transformer-based architectures for further accuracy gains. Incorporating sentiment analysis and macroeconomic indicators to improve feature selection. Extending datasets to include multiple global indices for better generalization.

References

1. D. C. Montgomery, C. L. Jennings, and M. Kulahci, *Introduction to time series analysis and forecasting*. John Wiley & Sons, 2015.
2. K. S. Chan and J. D. Cryer, Time Series Analysis with Applications in R. Springer, 2008.
3. R. S. Tsay, Analysis of Financial Time Series (Third Edition). John Wiley & Sons, Inc., 2010. doi: 10.1002/9780470644560.
4. Shweta, "Introduction to Time Series Forecasting - towards Data science," 2022. [Online]. Available: https://towardsdatascience.com/introduction-to-time-series-forecasting-part-1-average-and-smoothing-models-a739d832315
5. C. Janiesch, P. Zschech, and K. Heinrich, "Machine Learning and Deep Learning," Electronic Markets, vol. 31, no. 3, pp. 685–695, 2021.
6. A. Nielsen, Practical time series analysis: Prediction with statistics and machine learning. O'Reilly Media, 2019.
7. A. A. Adebiyi, A. O. Adewumi, and C. K. Ayo, "Stock price prediction using the ARIMA model," Proceedings - UKSim-AMSS 16th International Conference on Computer Modelling and Simulation, UKSim 2014, pp. 106–112, 2014, doi: 10.1109/UKSIM.2014.67.
8. B. U. Devi, D. Sundar, and P. Alli, "An Effective Time Series Analysis for Stock Trend Prediction Using ARIMA Model for Nifty Midcap-50," International Journal of Data Mining & Knowledge Management Process, vol. 3, no. 1, p. 65, 2013.
9. K. Sharma, "Machine learning in finance: history, technologies, and outlook," 2023. [Online]. Available: https://ubuntu.com/blog/machine-learning-in-finance-history-technologies-and-outlook
10. T. J. Strader, J. J. Rozycki, T. H. Root, and Y. H. J. Huang, "Machine learning stock market prediction studies: review and research directions," Journal of International Technology and Information Management, vol. 28, no. 4, pp. 63–83, 2020.
11. T. Jasic and D. Wood, "The profitability of daily stock market indices trades based on neural network predictions: a case study for the S&P 500, the DAX, the TOPIX and the FTSE in the period 1965–1999," Applied Financial Economics, vol. 14, no. 4, pp. 285–297, Feb. 2004, doi: 10.1080/0960310042000201228.
12. M. Durairaj and B. K. Mohan, "A Review of Two Decades of Deep Learning Hybrids for Financial Time Series Prediction," International Journal on Emerging Technologies, vol. 10, no. 3, pp. 324–331, 2019.
13. J. M. T. Wu, Z. Li, G. Srivastava, J. Frnda, V. G. Diaz, and J. C. W. Lin, "A CNN-based stock price trend prediction with futures and historical price," in 2020 International Conference on Pervasive Artificial Intelligence (ICPAI), IEEE, 2020, pp. 134–139.
14. E. Hoseinzade and S. Haratizadeh, "CNNpred: CNN-Based Stock Market Prediction Using a Diverse Set of Variables," Expert Syst Appl, vol. 129, pp. 273–285, 2019.
15. S. Selvin, R. Vinayakumar, E. A. Gopalakrishnan, V. K. Menon, and K. P. Soman, "Stock price prediction using LSTM, RNN and CNN-sliding window model," in 2017 International Conference on Advances in Computing, Communications and Informatics (ICACCI), IEEE, 2017, pp. 1643–1647.
16. G. Shobana and K. Umamaheswari, "Forecasting by machine learning techniques and econometrics: a review," in 2021 6th International Conference on Inventive Computation Technologies (ICICT), IEEE, 2021, pp. 1010–1016.
17. M. Asokan, A Study of Forecasts in Financial Time Series using Machine Learning Methods. 2022.
18. P. G. Zhang, "Time series forecasting using a hybrid ARIMA and neural network model," Neurocomputing, vol. 50, pp. 159–175, Jan. 2003, doi: 10.1016/S0925-2312(01)00702-0.
19. Y. S. Kao, K. Nawata, and C. Y. Huang, "Predicting Primary Energy Consumption Using Hybrid ARIMA and GA-SVR Based on EEMD Decomposition," Mathematics 2020, Vol. 8, Page 1722, vol. 8, no. 10, p. 1722, Oct. 2020, doi: 10.3390/MATH8101722.
20. A. Hayes, "What is a Time Series and How is it Used to Analyze Data?," 2022. [Online]. Available: https://www.investopedia.com/terms/t/timeseries.asp

21. M. Di Pietro, “Time Series Analysis for Machine Learning,” 2022. [Online]. Available: https://towardsdatascience.com/time-series-analysis-for-machine-learning-with-python-626bee0d0205
22. C. Cracan, “Retail Sales Forecasting using LSTM and ARIMA-LSTM: A Comparison with Traditional Econometric Models and Artificial Neural Networks,” 2020.
23. V. Rehal, “Interpreting ACF and PACF plots - SPUR ECONOMICS,” 2022. [Online]. Available: https://spureconomics.com/interpreting-acf-and-pacf-plots/#:~:text=Autocorrelation
24. A. K. Pandey, “Hands-On Stock Price Time Series Forecasting using Deep Convolutional Networks,” 2021. [Online]. Available: https://www.analyticsvidhya.com/blog/2021/08/hands-on-stock-price-time-series-forecasting-using-deep-convolutional-networks/
25. J. Martinez, “Introduction to Convolutional Neural Networks (CNNs),” 2020. [Online]. Available: https://aigents.co/data-science-blog/publication/introduction-to-convolutional-neural-networks-cnns
26. S. Albawi, T. A. Mohammed, and S. Al-Zawi, “Understanding of a Convolutional Neural Network,” in 2017 International Conference on Engineering and Technology (ICET), IEEE, 2017, pp. 1–6.
27. F. Tabsharani, “Support Vector Machine (SVM),” 2023. [Online]. Available: https://www.techtarget.com/whatis/definition/support-vector-machine-SVM
28. Lim, B., & Zohren, S. (2021). Time-series forecasting with deep learning: A review. Philosophical Transactions of the Royal Society A, 379(2194). DOI: 10.1098/rsta.2020.0209
29. Fischer, T., & Krauss, C. (2018). Deep learning with long short-term memory networks for financial market predictions. European Journal of Operational Research, 270(2), 654-669. DOI: 10.1016/j.ejor.2017.11.054.
30. M. Arshad, C. W. Onn, A. Ahmad, and G. Mogwe, “Big data analytics and AI as success factors for online video streaming platforms,” Front Big Data, vol. 8, Feb. 2025, doi: 10.3389/FDATA.2025.1513027.
31. Fama, E. F. (1970). Efficient capital markets: A review of theory and empirical work. The Journal of Finance, 25(2), 383–417. https://doi.org/10.2307/2325486
32. S. Bai, J. Z. Kolter, and V. Koltun, "An empirical evaluation of generic convolutional and recurrent networks for sequence modeling," arXiv:1803.01271, 2018.
33. M. Arshad, A. Ahmad, C. W. Onn, and E. A. Sam, “Investigating methods for forensic analysis of social media data to support criminal investigations,” Front. Comput. Sci., vol. 7, p. 1566513, Jun. 2025, doi: 10.3389/FCOMP.2025.1566513.
34. A. Almufarreh, A. Ahmad, M. Arshad, C. W. Onn, and R. Elechi, “Ethical Implications of ChatGPT and Other Large Language Models in Academia,” Frontiers in Artificial Intelligence, Aug. 2025.