Predicting Stock Markets

BSP3 (Academic Year 2025/26) University of Luxembourg

Erikas Kadiša
erikas.kadisa.001@student.uni.lu
University of Luxembourg
Belval, Esch-sur-Alzette, Luxembourg

Prof. Giacomo Di Tollo
giditollo@unisannio.it
Primary BSP Supervisor

Abstract

This paper presents the development and evaluation of a hybrid Bi-LSTM-CNN model for gold market (GLD) price prediction. By integrating ten years of historical market data with sentiment analysis derived from news articles via FinBERT, eleven features were engineered to capture both technical indicators and market psychology. The model was trained and validated using an expanding window cross-validation approach. Empirical results demonstrate that the model achieves a directional accuracy between 0.48 and 0.55, with simulated annual returns ranging from -21.1% to +19.9%. While the model shows strong predictive potential during periods of high volatility (2021–2022), its performance degraded during the consistent price surges of 2023–2025, likely due to a lack of similar trends in the historical training distribution. To facilitate practical use, a polyglot application featuring a Java-based GUI and a Python backend was developed to automate real-time data fetching, normalization, and inference.

1 Introduction

One of the most prominent challenges in 21st-century computational finance is the accurate prediction of market price fluctuations. Driven by the availability of data and computational resources, research has shifted toward sophisticated Machine Learning and Deep Learning architectures. Among these, the prediction of gold prices – represented by the GLD ETF – remains a critical area of focus due to its role as a hedge against inflation and its historical performance as a long-term investment.

Recent studies suggest that the tendencies in gold markets can be captured using Long Short-Term Memory (LSTM) networks [2, 17]. This Bachelor Semester Project (BSP) expands upon these findings by implementing a hybrid Bi-directional LSTM and Convolutional Neural Network (CNN) architecture. The approach aims to extract both spatial features from technical indicators and temporal patterns from historical sequences. In addition, this project incorporates market sentiment among the features by processing financial news articles with the FinBERT model to capture the psychological aspects of the market.

The project utilizes a dataset spanning ten years of GLD market prices and news archives, providing a robust foundation for evaluating the model's predictive performance.

2 Dataset

Data quality is a primary determinant of model performance; this section details the methodology used to derive and refine the dataset.

2.1 Feature selection

One first needs to determine which features are the most promising when inferring the predictions. For this purpose, an analysis of 10 scientific papers with similar goals was conducted to deduce the most frequent features used in the academic world. The overview is provided in Table 1.

#	Paper	Input Features & Indicators	Prediction Target
1	FB-GAN [13]	OHLCV, Sentiment	Next-day Close
2	HCLA Model [8]	OHLCV, RSI(14), MACD(12,26,9), BB(20)	Next-day Close
3	Advanced LSTM [9]	OHLCV, Sentiment, MA10, MA20, MA50	Next-day Close
4	DRL with NLP [5]	Prices, Sentiment, RSI, Momentum, SMA(9)	Next-day Open
5	GRU-Attention [22]	OHLCV	Next-day Close
6	Sentiment Eval. [19]	Close, Sentiment	5-day future prices
7	CNN-BiLSTM-AM [23]	OHLCV, Sentiment	Next-day Close
8	TabLSTM [21]	Micro/Macro, P/E, P/CF, Momentum	Next-day Trend
9	Feature Select. [24]	OHLCV, RSI, MACD, KDJ, William%R	Next-day Close
10	Real-time Pred. [15]	OHLCV, Sentiment, EMA5	Next 15min Price

Table 1: Comparison of Stock Price Prediction Papers

Among the papers, the most frequently used features were the daily market Open, High, Low, Close prices, and Volume (OHLCV), appearing in 7 out of 10 papers. The following popular features are sentiment (used by 6 papers) and moving averages (MA, SMA, EMAs, difference between EMAs (MACD), included in 5 papers). Three papers included the Relative Strength Indicator, and one paper included the volatility indicator – Bollinger Bands. For this reason, the following indicators were included in the project: Open, High, Low, Close prices, Volume, RSI of 14 days, MACD(12, 26, 9), BB low, BB High, and Sentiment with Article Count for the day. In addition, moving average variants were considered: two SMA Ratios (with SMA for 20 and 50 days) with the formula

$$\boxed{Ratio = \frac{Close}{SMA}},$$

and a MACDS(12, 26, 9). The SMA ratio this way is almost normalized and scattered around 1, while MACDS is a special signal, capturing the 9 day MACD tendencies and used in [1] (referred to as MACD line). Finally, the project also includes an additional volatility indicator as used in [6] – Average True Range.

The most popular prediction target choice among the papers was the next-day closing price, chosen by 7 papers. Following this practice, this project also aimed to predict the next-day closing price.

Table 2 summarizes the lookback window used in each of the 10 papers and the total data interval considered.

#	Paper	Lookback window	Total Data Interval
1	FB-GAN[13]	20 days daily	5 years
2	HCLA Model[8]	30 daily	1 year
3	Advanced LSTM[9]	60 daily	1 year
4	DRL with NLP[5]	60 past daily bars	3-5 years
5	GRU-Attention[22]	48 days daily	20 years
6	Sentiment Evaluation[19]	5-day, 10-day, and 30-day (compared)	20 years
7	CNN-BiLSTM-AM[23]	30 days daily	8-10 years
8	TabLSTM[21]	20 days	12 years
9	Feature Selection[24]	20 days daily	8-10 years
10	Real-time forecast[15]	30x past 15-minute intervals	a few months to years

Table 2: Time-wise comparison of stock price and index prediction papers

The window varies from paper to paper, ranging from 5 to 60 samples, with a mean of 30.25 samples. In the meantime, the total data interval, excluding paper number 10 (since its prediction target is the next 15 minutes), has periods ranging from 1 year up to 20 years, with the mean being 9 years. For this project, a window size of 30 was chosen, which provides sufficient temporal context for the Bi-LSTM to capture monthly trends while maintaining a computationally efficient input size for the CNN layers. As for the dataset, ten years (from 30.11.2015 to 30.11.2025) of market and news data were used in the project for training and inference.

2.2 Sources

The dataset used for model training and evaluation comprises 2, 514 samples of open-market trading days from November 30, 2015, to November 30, 2025. Daily market data for the `GLD` ticker was extracted via the `yfinance` Python module [4] and saved in a `CSV` file for easier management.

Later on, the corresponding technical indicators were calculated as described in Subsection 2.1.

The sentiment data was obtained using the GDELT large knowledge database[10] through BigQuery[11]. First, the database was queried for Yahoo Finance news links containing the `GLD` ticker. Article content was then extracted using the Trafilatura library [7], with the Internet Archive's Wayback Machine [12] serving as a fallback for unavailable URLs.

The text was then processed using the FinBERT model, which has been shown to be effective in determining market trends based on the sentiments ([14]). To accommodate the model's 512-token limitation, a "head-and-tail" strategy (effectiveness shown in [20]) was used: the first 128 tokens of the article (introduction) and the final 382 tokens (conclusion) were preserved. The remaining 2 tokens are reserved for the [CLS] and [SEP] special tokens, as required by the model. Finally, daily sentiment scores were computed as the arithmetic mean of all processed articles scores for a given day. This pipeline successfully processed 7, 170 articles, resulting in 2, 133 trading days enriched with a sentiment score. The remaining days without a score were assigned a default 0 value.

The entire workflow of how the data is transformed and used for inferring predictions is described in Figure 1.

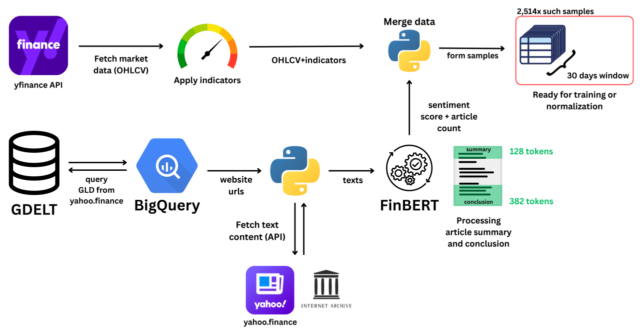


Figure 1: Dataset creation workflow

2.3 Normalization techniques

Despite the popularity of Min-Max normalization ($x_{new} = \frac{x - x_{min}}{x_{max} - x_{min}}$) usage, in practice, it is often unsuitable for financial time series, as it can introduce data leakage by incorporating global extrema from the future into the training set. Furthermore, because market prices generally exhibit a non-stationary upward trend, future values often exceed the [0, 1] training range, leading to poor generalization [16]. To mitigate this, this project utilizes rolling z-scores (as proposed in [18]) and logarithmic returns to ensure stationarity and prevent exploding gradients. Described below are the feature normalization techniques used:

(1) OHLC prices are normalized using logarithmic returns with the formula

$$\boxed{x_n^{new} = ln(\frac{x_n}{Close_{n-1}}) \times 100}$$

This makes the data stationary. Notably, since our target closing price is also normalized, the model will predict the change in the closing price from the previous day's closing price.

(2) Volume is normalized with the formula

$$\boxed{x_n^{new} = ln(\frac{x_n}{x_{n-1}}) \times 100}$$

(3) RSI, MACD, and MACDS features are normalized using rolling z-scores with the formula

$$\boxed{z_{score} = \frac{x - \bar{x}}{s}}$$

, where mean($\bar{x}$) and standard deviation(s) are considered for the last 90 days. This window captures quarterly volatility trends while remaining sensitive to recent market shifts.

(4) SMA20, SMA50, BB Low, and BB High were normalized with the formula

$$\boxed{x_n^{new} = (x_n - 1) \times 10}$$

(5) News Count was normalized with an expanding rolling z-score using the same formula as explained before. The only difference is that the number of days considered is $min(90, n)$, because there is no easily accessible sentiment data for days before 30.11.2015.

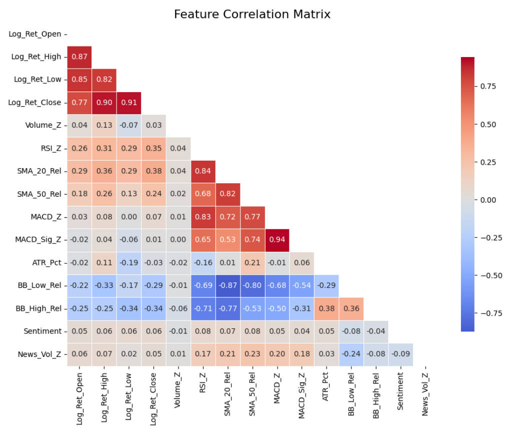
(6) Sentiment scores and ATR values did not require normalization, as they are relatively balanced.

The basic statistics of the features before and after normalization are displayed in Table 3. All features after normalization are in the range of $(-6.7, 8.3)$.

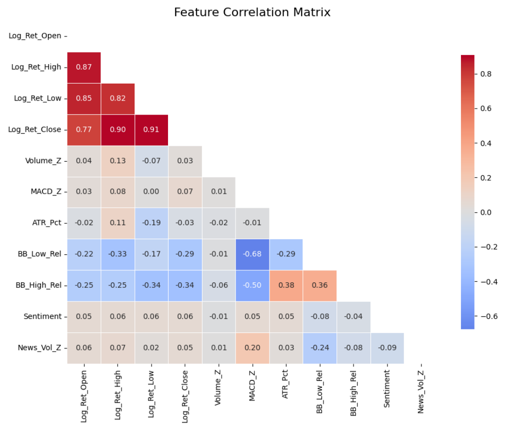
Feature	Statistic	MIN	MAX	MEAN	MEDIAN	STDDEV
Open	Original	89.53	790.62	273.90	198.77	170.50
	Normalized	-4.21	5.28	0.05	0.06	0.70
Close	Original	88.37	789.47	273.93	198.53	170.28
	Normalized	-6.64	4.79	0.05	0.06	0.93
Low	Original	87.55	780.29	270.38	196.41	168.17
	Normalized	-7.07	3.99	-0.39	-0.29	0.84
High	Original	89.91	795.71	277.35	201.33	172.36
	Normalized	-3.75	6.20	0.46	0.42	0.81
Volume	Original	2.515×10^3	2.323×10^8	2.177×10^7	1.812×10^7	1.473×10^7
	Normalized	-1.38	1.72	0.00	-0.02	0.42
RSI	Original	20.04	88.52	53.16	52.67	12.30
	Normalized	-3.20	3.65	0.03	-0.11	1.17
SMA20	Original	0.92	1.10	1.01	1.00	0.02
	Normalized	-0.84	1.00	0.05	0.03	0.22
SMA50	Original	0.92	1.19	1.01	1.01	0.04
	Normalized	-0.84	1.86	0.13	0.09	0.36
MACD	Original	-3.55	15.63	0.74	0.39	2.02
	Normalized	-3.72	3.33	0.08	-0.08	1.27
MACDS	Original	-2.98	13.20	0.73	0.36	1.92
	Normalized	-3.15	3.16	0.09	-0.09	1.31
ATR	Original	0.71	8.24	1.91	1.72	1.08
	Normalized	0.71	8.24	1.91	1.72	1.08
BB Low	Original	0.82	1.03	0.97	0.97	0.03
	Normalized	-1.79	0.25	-0.35	-0.27	0.29
BB High	Original	0.98	1.18	1.03	1.02	0.02
	Normalized	-0.18	1.81	0.26	0.20	0.24
Sentiment	Original	-0.97	0.94	-0.12	0.00	0.37
	Normalized	-0.97	0.94	-0.12	0.00	0.37
Article Count	Original	0	18	2.33	2.00	2.46
	Normalized	-3.51	3.98	0.03	0.02	1.08

Table 3: GLD stock features statistics from 2015-11-30 until 2025-11-30 (10 years period)

Initially selected features may be excessive if they convey the same information. By finding the correlated features and filtering them, this redundancy can be reduced. Figure 2a displays the initial correlation matrix, while Figure 2b depicts the matrix after excluding some features with a similarity coefficient of ≥ 0.7 or ≤ -0.7. Note that since the majority of papers kept OHLCV data, this project also retains it.



(a)



(b)

Figure 2: Feature correlation matrices with all features (a) and with similar (coefficient ≥ 0.7 or ≤ -0.7) features filtered (b). Note that filtering does not apply to OHLC.

The correlation matrix revealed that the normalized MACD is highly correlated with the RSI, SMA20, SMA50, and MACDS. Additionally, BB Low and BB High show high correlations with RSI and SMAs. After filtering out RSI, both SMAs, and MACD, all high correlations are effectively removed, leaving the following 11 meaningful features:

(1) Open
(2) High
(3) Low
(4) Close
(5) Volume
(6) MACD(12, 26, 9)
(7) ATR
(8) BB low
(9) BB High
(10) Sentiment
(11) Article Count

3 Model

Financial time-series prediction has seen a surge in Neural Network-based solutions, ranging from Graph Neural Networks (GNN) to Recurrent Neural Networks (RNN). This project utilizes a hybrid Bi-directional LSTM-CNN architecture, a design that leverages the complementary strengths of spatial feature extraction and temporal sequence modeling.

3.1 Bi-LSTM-CNN Architecture

The Long Short-Term Memory (LSTM) network is a specialized RNN architecture designed to capture long-term dependencies while mitigating the vanishing gradient problem. By integrating a Bi-directional component, the model processes input sequences in both forward and backward directions, thereby accessing context from both ends of the lookback window.

This architecture is improved by the introduction of Convolutional Neural Networks (CNN) as a localized feature extraction layer. The CNN identifies spatial patterns across the 11 input features before the sequence is further passed to the Bi-LSTM, which is responsible for the time-wise analysis of the 30 days window. This specific hybrid configuration was shown to have superior performance in the literature, achieving R^2 values as high as 0.95 [2].

The model was implemented in Python, following the architecture design proposed in [3]. The model's internal layers and the internal transformations of the input tensors are depicted in Figure 3

The primary layer in the architecture is the CNN with 64 filters and a kernel size of 5, padding of 2. This layer intakes tensors of dimensions 11×30, representing 30 timestamps and 11 features. The CNN scans along the timestep axis, multiplying each block of 5×11 features with 64 filter matrices. Each multiplication is summed

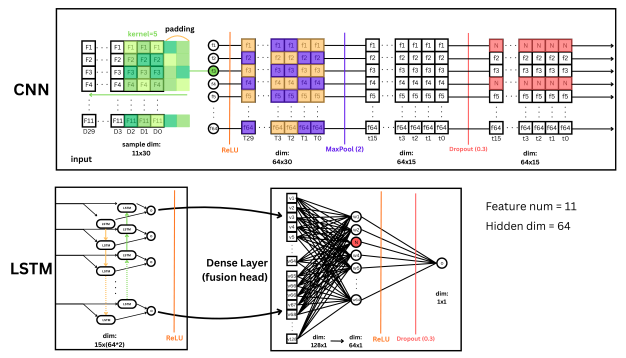


Figure 3: Bi-LSTM architecture implementation. "Hidden dim" hyperparameter controls the complexity of the model (number of internal nodes).

afterward to output a single value for that filter, resulting in 64 × 30 feature tensor. It is then passed to the MaxPool layer of size 2, applying the ReLU activation function in between. The MaxPool layer operates in the timestep dimension, picking the maximum weights for each 2-timestep pair, thus reducing the dimensions in half. The resulting tensor is fed into the LSTM layer, with a Dropout layer in between to enforce generalization. The Bi-LSTM layer traverses the timesteps in forward and backward directions and outputs a 64 × 1 vector at each end, serving as summaries. After another ReLU activation function, the vectors are concatenated and forwarded to the final fusion head layer. Finally, the dense layer is applied, reducing the dimension to 64×1, followed by the last ReLU function and a Dropout layer before outputting the final next-day Close price prediction.

3.2 Training

To evaluate the model's predictive robustness while respecting the chronological nature of financial data, an expanding window cross-validation strategy was employed ([17]). Unlike standard K-fold cross-validation, which shuffles data and inadvertently introduces temporal leakage by training on future prices to predict the past, the expanding window technique ensures that we train with data up to a certain point in time and validate on the future, iteratively increasing the training window.

As illustrated in Figure 4, the dataset was split into 5 consecutive folds. In each iteration, the training window (green in the figure) expands by 1 year, while the validation and testing splits shift one year forward. This setup allows for the evaluation of the model during different market tendencies of volatility (2021–2022) and growth (2023–2025).

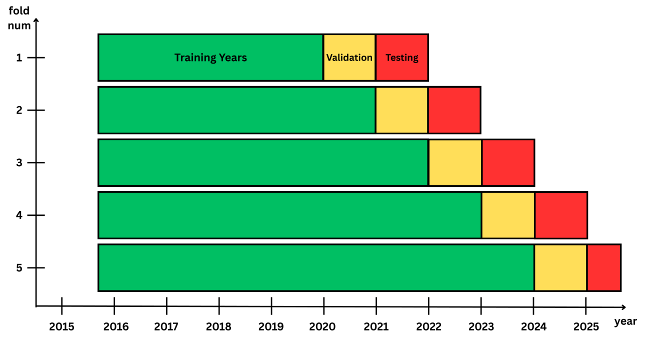


Figure 4: Datasplit used for training the model.

During each fold, the model is trained with a maximum of 100 epochs. To prevent overfitting, training is stopped by an `EarlyStopping` mechanism with a patience of 10 epochs, monitoring validation loss. Furthermore, a `ReduceLROnPlateau` schedule was implemented to dynamically adjust the learning rate, halving it when the validation loss plateaued for more than 10 epochs. More details on the chosen training parameters are described in Table 4.

Hyperparameter	Value
Hidden dimension	64
Batch size	64
Epochs	100
Loss function	Mean Squared Errors (MSE)
Optimizer	Adam
Learning Rate	0.001
Scheduler	ReduceLROnPlateau
Sched. patience	10
Sched. factor	0.5
EarlyStopping patience	10

Table 4: Hyperparameters chosen for the training

3.3 Results

The model's performance is evaluated across several metrics: Mean Average Error (MAE), Epsilon Hit (percentage of predictions within a 0.2% error margin), Directional Accuracy, and the Sharpe Ratio. Additionally, a backtesting simulation was conducted to assess the financial viability of the model's signals.

As shown in Table 5, the model demonstrated strong predictive power during the 2021–2022 period, with directional accuracy $0.54 - 0.55$ and a Sharpe Ratio of $0.58 - 1.55$. However, performance deteriorated from 2023 onward due to the unprecedented surge in gold prices, which the model had never encountered before in the training data. From 2023 to the present, the gold price has more than doubled. The model struggled to adapt to this new market regime but shows signs of a adaptation, characterized by a surge in directional accuracy in the testing year 2025, as the model had 2023 year data in the training set.

Indicator	2021	2022	2023	2024	2025
MAE	0.65	0.81	0.70	0.87	1.11
Epsilon Hit (0.2%)	0.25	0.17	0.24	0.13	0.15
Directional Accuracy	0.54	0.55	0.49	0.48	0.50
Sharpe Ratio	1.55	0.58	-0.65	-0.85	-0.96

Table 5: Performance indicators across five testing folds (2021–2025)

Meanwhile, Table 5 provides an overview of the last 4 folds for the years 2022–2025, plotting a graph with predictions versus actual prices. One could notice that the model seems more stable in 2022 but is chaotic in the other years.

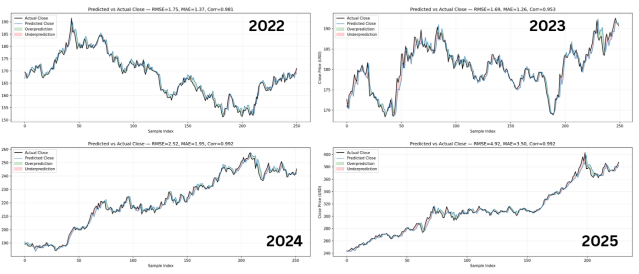


Figure 5: Model predictions vs. actual values for test years 2021–2025

In a backtest simulation, a financial decision to buy, hold, or go short is performed if the predicted price change is greater than 0.1%. The transaction price was set to 0.05%, while the initial capital was set to $1, 000. Figure 6 depicts the simulations performed over the last 4 epochs for each test year.

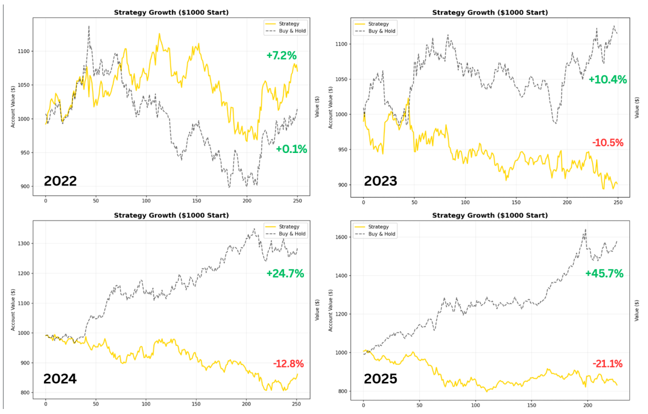


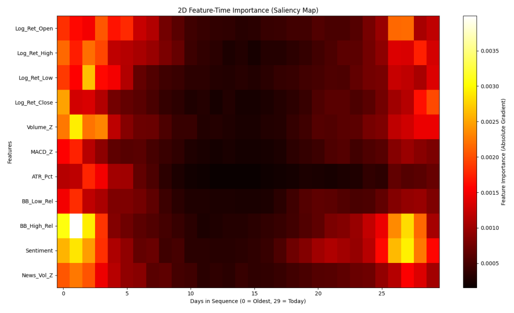
Figure 6: Simulation for test years 2022–2025. Model options: short, hold, buy. Transaction cost is 0.05%, initial capital $1000. Log returns are shown for the portfolio options at the end of the year.

The Figure reveals that 2023–2025 years would have been very unprofitable, with losses ranging from −10.5% to −20.1%. In a real scenario, it would have been more profitable to just buy and hold the stock. However, in the years 2021 (not shown here) and 2022, the model demonstrates very strong potential with profits +19.9% in 2021 and +7.2% in 2022, outperforming the buy and hold strategy. Therefore, in its current state, the model outperforms only when there is sufficient volatility rather than consistent growth.

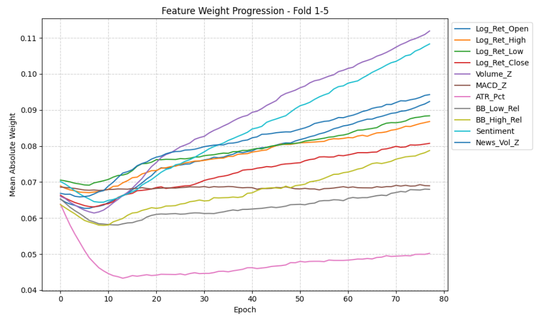
3.4 Feature Importance and Interpretability

A feature analysis was conducted to aid in interpreting the model's decisions. The saliency heatmap (Figure 7a) provides insights into the CNN's decision-making process. Meanwhile, in 7b a Moving Average Weight is plotted for each of the features, showing how the model assigns weight to each feature as the training progresses.

The model exhibits a "U-shaped" temporal focus, primarily activating features from the most recent 10 days and the earliest 5 days (days 25–30) of the look-back window. Interestingly, the intermediary period contributes negligibly to the final output. The most significant feature found by the model was the Bollinger Bands High indicator on day 29, which appears in white in the heatmap. In 7b, the least decisive feature appears to be ATR (which is also well reflected in the saliency



(a)



(b)

Figure 7: Saliency heatmap for CNN filter on last fold (a), and Moving Average Weight graph over epochs for each feature (b)

map), while the most contributive features seem to be Sentiment and Volume. MACD and Bollinger Bands Low in the graph grow very slowly and seem to be redundant (this is also reflected in the saliency heatmap by dark lines).

Overall, the model is capable of using the given input tensors and features to predict future prices with potentially profitable returns, but it struggles with consistent stock growth. Additionally, the feature analysis suggests that the input tensor organization could be improved by excluding the useless information.

4 Application

To facilitate user interaction while abstracting the underlying implementation details, a cross-platform application was developed. The system utilizes a Java-based Graphical User Interface (GUI) paired with a Python backend to provide closing price predictions.

4.1 Workflow

The application uses communication channels via the Transmission Control Protocol (TCP). Upon the launch

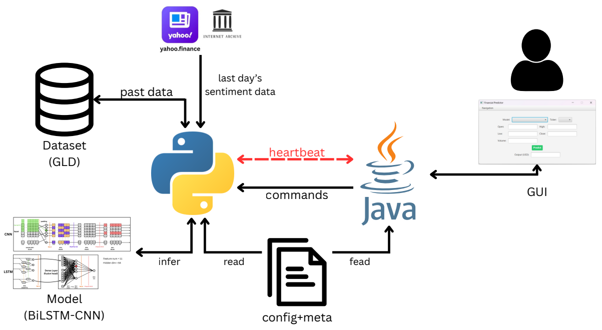


Figure 8: Application workflow

of the application, the Java host initializes a logging subsystem and dynamically allocates two free TCP ports. Shortly after, the Java process spawns the Python process through a Command Prompt application.

Communication is established on the local loopback address (`127.0.0.1`) using a dual-port strategy:

- **Heartbeat Port:** A dedicated channel for status monitoring to ensure that the backend process remains responsive.
- **Command Port:** A channel for transmitting user inputs and receiving model predictions.

Before making the predictions, the Python side has to "warm up" — load all the necessary libraries. This might take around 30 seconds. Nevertheless, the user sees the usable GUI and can select the inference model, choose the ticker to use, input OHLCV metrics for the day, and receive the prediction in USD as soon as everything is loaded. The workflow visualization is provided in Figure 8.

Once valid market data (OHLC) is received, the Java host transmits these parameters to the Python engine. Internally, the backend performs several critical steps:

(1) **Data Integration:** The 29 last historical dataset samples from the dataset are fetched and joined with the user input.
(2) **Feature Engineering:** The engine automatically calculates the required technical indicators (RSI, MACD, etc.) and normalizes the data according to the metadata saved during the training phase.
(3) **Automated Sentiment Fetching:** To improve user experience, the system independently fetches

current news articles, processes them via FinBERT to generate sentiment scores, and calculates the daily news volume.

(4) **Inference:** A 30-day temporal window is constructed and passed through the Bi-LSTM-CNN model to generate the final prediction.

In case of incorrect arguments or errors, the system warns the user through popups and red-highlights (in case some data is missing).

5 Conclusion

This Bachelor Semester Project (BSP) successfully implemented a hybrid Bi-LSTM-CNN architecture for predicting the next-day closing price of the GLD ETF. By integrating eleven distinct features—including historical OHLCV data, technical indicators, and FinBERT-derived news sentiment—the project demonstrated the potential of combining spatial feature extraction with temporal sequence modeling.

The experimental results revealed a high degree of sensitivity to market regimes. While the model achieved a remarkable +19.9% return during the volatile 2021–2022 period, its performance deteriorated during the bullish trend of 2023–2025. This performance gap highlights a critical challenge relevant in computational finance: the difficulty of generalizing models across diverse economic cycles. The project suggests that while the Bi-LSTM-CNN is highly effective at identifying mean-reverting patterns, it requires further exposure to sustained growth datasets to remain profitable in all market conditions.

Furthermore, the interpretability analysis provided valuable insights into feature importance and temporal relevance. The discovery that the model primarily relies on the "head" and "tail" of the 30-day lookback window—effectively ignoring the 15 day period in-between—suggests potential for future optimizations in window formation. The significant contribution of Sentiment and Volume features validates the project's logic that market psychology is a non-trivial predictor of gold price movement.

Beyond the theoretical foundation, this project incorporated the practical application of the model through Java-GUI application development, demonstrating the importance of networking API-based communication channels in polyglot applications, where one language is used for user interactions and another for predictions. Ultimately, this project reinforced core competencies in data normalization, feature selection, and inter-process communication, providing a solid foundation for future projects involving neural networks.

References

[1] Alberto Antonio Agudelo Aguirre, Néstor Darío Duque Méndez, and Ricardo Alfredo Rojas Medina. 2021. Artificial intelligence applied to investment in variable income through the MACD (moving average convergence/divergence) indicator. *Journal of Economics, Finance and Administrative Science* 26, 52 (08 2021), 268–281. arXiv:https://www.emerald.com/jefas/article-pdf/26/52/268/1348499/jefas-06-2020-0203.pdf doi:10.1108/JEFAS-06-2020-0203

[2] Amirhossein Amini and Robab Kalantari. 2024. Gold price prediction by a CNN-Bi-LSTM model along with automatic parameter tuning. *PLOS ONE* 19 (03 2024), e0298426. doi:10.1371/journal.pone.0298426

[3] M. Khairul Anam, Sarjon Defit, Haviluddin Haviluddin, Lusiana Efrizoni, and Muhammad Firdaus. 2024. Early Stopping on CNN-LSTM Development to Improve Classification Performance. *Journal of Applied Data Sciences* 5, 3 (2024), 1175–1188. doi:10.47738/jads.v5i3.312

[4] Ran Aroussi. 2023. *yfinance: Download market data from Yahoo! Finance's API.* https://github.com/ranaroussi/yfinance Python library for financial data acquisition.

[5] Alamir Awad, Saleh Elkaffas, and Mohamed Fakhr. 2023. Stock Market Prediction Using Deep Reinforcement Learning. *Applied System Innovation* 6 (11 2023), 106. doi:10.3390/asi6060106

[6] Wei Bao, Jun Yue, and Yulei Rao. 2017. A deep learning framework for financial time series using stacked autoencoders and long-short term memory. *PLoS ONE* 12, 7 (July 2017), e0180944. doi:10.1371/journal.pone.0180944

[7] Adrien Barbaresi. 2021. *Trafilatura: A Python library for text discovery and extraction.* doi:10.5281/zenodo.4540303

[8] Bhanujyothi C. and I. Jacob. 2025. A Hybrid CNN-LSTM Attention-Based Deep Learning Model for Stock Price Prediction Using Technical Indicators. *Engineering, Technology and Applied Science Research* 15 (10 2025), 28012–28017. doi:10.48084/etasr.12685

[9] Rajneesh Chaudhary. 2025. Advanced Stock Market Prediction Using Long Short-Term Memory Networks: A Comprehensive Deep Learning Framework. arXiv:2505.05325 [cs.CE] https://arxiv.org/abs/2505.05325

[10] GDELT Project. 2024. The GDELT Project: A Global Database of Society. https://www.gdeltproject.org/ Accessed: 2026-01-09.

[11] Google Cloud. 2024. BigQuery: A Scalable, Cost-Effective, and Serverless Data Warehouse. https://cloud.google.com/bigquery Cloud computing platform for data analytics.

[12] Internet Archive. 2026. The Wayback Machine. http://archive.org/wayback/ Accessed: 2026-01-09.

[13] Jainendra Kumar Jain and Ruchit Agrawal. 2024. FB-GAN: A Novel Neural Sentiment-Enhanced Model for Stock Price Prediction. In *FINNLP*. https://api.semanticscholar.org/CorpusID:269951050

[14] Jainendra Kumar Jain and Ruchit Agrawal. 2024. FB-GAN: A Novel Neural Sentiment-Enhanced Model for Stock Price Prediction. In *Proceedings of the Joint Workshop of the 7th Financial Technology and Natural Language Processing, the 5th Knowledge Discovery from Unstructured Data in Financial Services, and the 4th Workshop on Economics and Natural Language Processing*, Chung-Chi Chen, Xiaomo Liu, et al. (Eds.). Association for Computational Linguistics, Torino, Italia, 85–93. https://aclanthology.org/2024.finnlp-1.9/

[15] Riya Kalra, Tinku Singh, Suryanshi Mishra, Satakshi Satakshi, Naveen Kumar, Taehong Kim, and Manish Kumar. 2024. An efficient hybrid approach for forecasting real-time stock market indices. *Journal of King Saud University - Computer and Information Sciences* 36 (09 2024), 102180. doi:10.1016/j.jksuci.2024.102180

[16] Sarat Nayak, Bijan Misra, and Prof. Dr. H. Behera. 2014. Impact of Data Normalization on Stock Index Forecasting. *International Journal of Computer Information Systems and Industrial Management Applications* 6 (12 2014), 357–369.

[17] Muhammad Rizky Nurhambali, Yenni Angraini, and A. Fitrianto. 2024. Implementation of Long Short-Term Memory for Gold Prices Forecasting. *Malaysian Journal of Mathematical Sciences* 18 (06 2024), 399–422. doi:10.47836/mjms.18.2.11

[18] Eduardo Ogasawara, Leonardo Martinez, Daniel de Oliveira, Idonot Send Email To Zimbrao, Gisele Pappa, and Marta Mattoso. 2010. Adaptive Normalization: A novel data normalization approach for non-stationary time series. *Proceedings of the International Joint Conference on Neural Networks*, 1–8. doi:10.1109/IJCNN.2010.5596746

[19] Tachapong Panpoonsup, Chaklam Silpasuwanchai, Chanapa Pananookooln, and Matthew Dailey. 2022. Evaluating the Effectiveness of Sentiment-Based Models for Stock Price Prediction. *SSRN Electronic Journal* (01 2022). doi:10.2139/ssrn.4185669

[20] Chi Sun, Xipeng Qiu, Yige Xu, and Xuanjing Huang. 2019. How to Fine-Tune BERT for Text Classification?. In *Chinese Computational Linguistics*. Springer, 194–206. doi:10.1007/978-3-030-32381-3_16

[21] Xiaolu Wei, Hongbing Ouyang, and Muyan Liu. 2022. Stock index trend prediction based on TabNet feature selection and long short-term memory. *PLOS ONE* 17 (12 2022). doi:10.1371/journal.pone.0269195

[22] Ziyue Zhao. 2025. Stock Price Prediction and Analysis Using Neural Network Models. *Applied and Computational Engineering* 157 (05 2025), 37–42. doi:10.54254/2755-2721/2025.PO23491

[23] Xinyuan Zheng. 2023. Stock Price Prediction Based on CNN-BiLSTM Utilizing Sentiment Analysis and a Two-layer Attention Mechanism. *Advances in Economics, Management and Political Sciences* 47 (12 2023), 40–49. doi:10.54254/2754-1169/47/20230369

[24] Wanbao Zhou. 2021. Analysis of Feature Selection for Stock Price Prediction with LSTM: A Case Study on China's New Energy Leading Stocks. *Journal of Information and Computing Science* 16, 2 (2021), 108–117. doi:10.2024-JICS-22368