Volume XI Issue 3 Year 2026 | Page 782-793 ISSN: 2527-9866
Received: 30-06-2026 | Revised: 31-07-2026 | Accepted: 06-08-2026

Comparison of Hyperparameter Optimization Methods for LSTM-Based XAU/USD Forecasting and Web-Based System Implementation

Muhamad Irsad Nizarudin[1], Yudie Irawan[2], Anteng Widodo[3]
[1,2,3] Department of Information Systems, Muria Kudus University, Indonesia
e-mail: 202253098@umk.ac.i [1], yudie.irawan@umk.ac.id[2], anteng.widodo@umk.ac.id[3]
*Correspondence: *202253098@umk.ac.id*

Abstract: *Unequal search budgets and single-seed experiments can confound comparisons of hyperparameter optimization methods in financial forecasting. This study compares Grid Search, Random Search, and Bayesian Optimization for tuning a Long Short-Term Memory model to forecast the XAU/USD closing price for the next trading day. The dataset comprised 1,705 daily observations from January 2020 to July 2026 using Open, High, Low, Close, and release-date-aligned United States inflation. Each method evaluated the same 32 configurations using three random seeds, resulting in 96 candidate-model evaluations per method. Performance was assessed on 37 independent testing dates and descriptively examined using a five-fold post-selection walk-forward diagnostic without repeating hyperparameter optimization within each fold. All methods selected the same configuration and produced a mean testing MAPE of 2.785052% and an ensemble MAPE of 2.696161%. Grid Search reached the final-best configuration earlier, but naïve persistence achieved the lowest MAPE of 1.358325%. Thus, optimization improved the LSTM relative to the predefined baseline but did not outperform persistence. The procedures were also implemented in a Streamlit application. The findings are limited to the examined dataset, search space, seeds, testing period, and computational environment.*

Keywords: *Gold Price Forecasting, Hyperparameter Optimization, Long Short-Term Memory (LSTM), Naïve Persistence, Streamlit.*

1. Introduction

Gold is widely used as a store of value and a protective asset, although its safe-haven properties vary with market and economic conditions [1], [2], [3]. Forecasting XAU/USD remains difficult because gold prices exhibit nonlinear and volatile behavior [4], [5]. Long Short-Term Memory (LSTM) models can capture temporal dependencies, but their performance depends on preprocessing, architecture, and hyperparameters [6], [7]. Grid Search, Random Search, and Bayesian Optimization explore hyperparameter configurations differently, while their outcomes depend on the search space, objective function, evaluation budget, and experimental conditions [8], [9].

Previous studies have applied tuned CNN–Bi-LSTM and standard LSTM models to gold-price forecasting [4], [5], [6]. Grid Search has been examined in financial time-series forecasting [10], and Bayesian Optimization to an Attention-LSTM model in another forecasting domain [11]. However, differences in datasets, input variables, forecasting horizons, architectures, validation protocols, computational budgets, random seeds, and benchmarks limit direct comparison across studies [12], [13], [14]. A controlled comparison of the three methods using the same XAU/USD dataset, LSTM architecture, discrete search space, evaluation budget, seeds, chronological partitions, selection criterion, and independent testing protocol therefore remains insufficiently addressed.

Fair comparison is important because random splitting may introduce future information, whereas unequal budgets and single-seed experiments may produce misleading conclusions. Evaluation should preserve chronological order and apply consistent preprocessing, search spaces, metrics, budgets, and random seeds [14], [15]. A naïve persistence benchmark is also required to determine whether an optimized model improves upon a simple forecasting rule [16].

This study addresses four questions: RQ1, whether the three methods select different configurations and produce different forecasting performance under identical conditions; RQ2, how they differ in trial-to-best and time-to-best; RQ3, whether the optimized LSTM models outperform the predefined baseline and naïve persistence benchmark; and RQ4, whether the experimental procedures can be implemented consistently in a Streamlit application.

The novelty is evaluative rather than algorithmic. Each method evaluates the same 32 configurations using three seeds, yielding 96 candidate-model evaluations per method. The study distinguishes forecasting accuracy from search efficiency and evaluates the selected models using seed-level uncertainty, ensemble predictions, statistical tests, persistence benchmarking, volatility-regime analysis, and a five-fold post-selection walk-forward diagnostic. The same procedures are implemented in Streamlit and verified through 22 black-box testing scenarios.

2. Methods

Research Design

This study used a quantitative comparative design to evaluate a predefined LSTM baseline, Grid Search, Random Search, and Bayesian Optimization for forecasting the XAU/USD closing price for the next trading day. A deterministic naïve persistence method was included as a benchmark. All LSTM scenarios used identical datasets, input features, prediction targets, architectures, chronological partitions, preprocessing procedures, training conditions, random seeds, and evaluation metrics. The three optimization methods were assigned the same 32-configuration discrete search space and 96-evaluation budget. They were compared based on selected configurations, forecasting performance, trial-to-best, and time-to-best.

Dataset and Data Preprocessing

The dataset combined daily XAU/USD Open, High, Low, and Close prices from January 1, 2020, to July 21, 2026, obtained through the Twelve Data Time Series API [17], with monthly United States Consumer Price Index data from the FRED CPIAUCNS series [18]. The year-on-year inflation rate was calculated as:

$$\text{Inflation}_t^{YoY} = \left(\frac{CPI_t}{CPI_{t-12}} - 1\right) \times 100\% \quad (1)$$

where CPI_t is the CPIAUCNS value for reference month t, and $_{CPIt-12}$ is the corresponding value for the same month of the previous year. After checking for missing values, invalid records, duplicates, and chronological consistency, weekend observations were removed. To prevent look-ahead bias, inflation values were aligned with trading observations according to their official release dates using a backward merge_asof, subject to *available_date* ≤ *trading_date*. The final inputs were Open, High, Low, Close, and year-on-year inflation, while the XAU/USD closing price for the next trading day was used as the prediction target.

Data Splitting, Normalization, and Sequence Construction

The 1,705 observations were divided chronologically without shuffling into 1,668 development observations and 37 independent testing observations. The development data were further split into 1,334 fit and 334 validation observations. Separate Min–Max scalers for the input features and target were fitted only to the fit data and applied unchanged to the validation and testing data:
Min–Max normalization transformed each variable into the range 0–1:

$$x_i' = \frac{x_i - x_{\min}}{x_{\max} - x_{\min}} \quad (2)$$

where X' represents the normalized value, X is the original value, and $X_{\min}$ and $X_{\max}$ are obtained exclusively from the fit dataset. A 30-observation window with five input features produced 1,304 fit, 334 validation, and 37 testing sequences. Predictions were inverse-transformed before evaluation, and preceding observations were used only as input context to prevent target leakage.

LSTM Architecture and Experimental Scenarios

All scenarios used one LSTM layer, one dropout layer, one dense layer with ReLU activation, and one linear output neuron with a 30×5 input. Adam and Mean Squared Error were used as the optimizer and loss function. The predefined baseline used 64 LSTM units, a dropout rate of 0.20, 32 dense units, a learning rate of 0.001, and a batch size of 32 [19].

Grid Search, Random Search, and Bayesian Optimization used the same 32-configuration search space presented in Table 1. Each configuration was evaluated using seeds 42, 123, and 2024, resulting in 96 evaluations per method. Grid Search followed a fixed exhaustive order, Random Search evaluated all configurations without replacement, and Bayesian Optimization used five initial configurations followed by a discrete Gaussian-process surrogate with Expected Improvement while excluding previously evaluated configurations.

Table 1. Hyperparameter Search Space and Training Configuration

Component	Candidate value	Status
LSTM units	32, 64	Optimized
Dropout rate	0.10, 0.20	Optimized
Dense units	16, 32	Optimized
Learning rate	0.0005, 0.001	Optimized
Batch size	16, 32	Optimized
Maximum epochs	50	Fixed
EarlyStopping patience	5	Fixed
Restore best weights	True	Fixed
Shuffle	False	Fixed
Random seeds	42, 123, 2024	Fixed
Bayesian initial configurations	5 per seed	Fixed
Selection objective	Minimum mean best validation loss	Fixed
HPO evaluations per method	96	Fixed

Models were trained for up to 50 epochs using EarlyStopping with a patience of five, restoration of the best weights, and disabled shuffling. The final configuration was selected according to the lowest mean best validation loss across the three seeds, with mean validation MAPE, mean validation RMSE, and configuration ID used sequentially for tie-breaking. The selected configurations were then retrained using the same seeds, and ensemble predictions were

obtained by averaging the aligned seed-level predictions. The independent testing dataset was excluded from hyperparameter optimization, configuration selection, scaler fitting, and EarlyStopping.

The model-development and evaluation experiments were conducted in Google Colab using Python 3.12.13, TensorFlow 2.20.0 and Keras 3.13.2 [20], NumPy 2.0.2, pandas 2.2.2, scikit-learn 1.6.1, and SciPy 1.16.3. Model training was performed using an NVIDIA Tesla T4 GPU. The web-based forecasting system was implemented in a separate local environment using Python 3.13.2, Streamlit 1.52.2 [21], Plotly 6.8.0, TensorFlow 2.21.0 and Keras 3.14.0 [20], NumPy 2.2.5, pandas 2.2.3, scikit-learn 1.6.1, and SciPy 1.15.3

Model Evaluation and Post-Selection Walk-Forward Diagnostic

The baseline and the configurations selected by Grid Search, Random Search, and Bayesian Optimization were evaluated using seeds 42, 123, and 2024. The independent testing dataset contained 37 observations from June 1 to July 21, 2026, and was excluded from hyperparameter optimization, configuration selection, scaler fitting, and EarlyStopping.

Seed-level MAE, RMSE, and MAPE were summarized using the mean and sample standard deviation. The 95% confidence interval was reported only for MAPE:

$$\mathrm{CI}_{95\%} = \bar{x} \pm t_{1-\alpha/2,n-1}\frac{s}{\sqrt{n}} \tag{3}$$

where $\bar{x}$ is the mean of the seed-level metric, $s$is the sample standard deviation, $n$is the number of random seeds $(n = 3)$, is the number of random seeds, and $t_{1-\alpha/2,n-1}$ is the critical value of the Student's t-distribution.

A naïve persistence model was included as a deterministic benchmark and was defined as:

$$\hat{y}_t = y_{t-1} \tag{4}$$

where $\hat{y}_t$is the forecast for trading date t, and y_{t-1} is the most recently observed actual closing price. For the first independent-testing prediction.y_{t-1} was the last actual closing price from the validation period.

where the first testing prediction used the last actual closing price from the validation period. Forecasting performance was evaluated using MAE, RMSE, and MAPE:

$$\mathrm{MAE} = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i| \tag{5}$$

$$\mathrm{RMSE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2} \tag{6}$$

$$\mathrm{MAPE} = \frac{100\%}{n}\sum_{i=1}^{n}\left|\frac{y_i - \hat{y}_i}{y_i}\right| \tag{7}$$

where y_i, and $\hat{y}_i$ are the actual and predicted closing prices, respectively, and $n = 37$. for independent testing. Pairwise comparisons of the aligned ensemble predictions used the

Diebold–Mariano test [22]. and the Wilcoxon signed-rank test with Holm correction at $\alpha = 0.05$. Practical differences were assessed using absolute and relative MAPE differences, while search efficiency was measured using mean trial-to-best and time-to-best.

For the post-selection walk-forward diagnostic, the 1,668-observation development dataset was divided into six chronological blocks, producing five expanding-window folds. Each fold used an 80:20 internal fit–validation split; scalers were fitted only to the internal fit data, and EarlyStopping used the internal validation data. Hyperparameter optimization was not repeated within the folds; the previously selected configurations were retrained for each fold and seed. Therefore, the results were interpreted descriptively rather than as nested model-selection performance.

The 37 testing dates were also classified into low-, medium-, and high-volatility regimes using terciles of realized absolute returns. Because the regimes were defined ex post, the results were interpreted descriptively. System verification compared the Streamlit application outputs with the functional requirements and Google Colab reference results.

3. Results and Discussion

Data Processing Results

The initial dataset contained 1,844 observations from January 1, 2020, to July 21, 2026. After removing 139 weekend records and completing checks for missing values, invalid records, duplicates, and chronological consistency, 1,705 observations remained. Release-date alignment confirmed that each inflation value was publicly available on or before its corresponding trading date, thereby preventing look-ahead bias.

The dataset was divided chronologically into 1,334 fit, 334 validation, and 37 independent-testing observations. Using a 30-observation window with five input features produced 1,304 fit, 334 validation, and 37 independent-testing sequences. Historical observations preceding the validation and testing periods were used only as input context, while all prediction targets remained within their respective chronological partitions.

Hyperparameter Optimization and Validation Results

Grid Search, Random Search, and Bayesian Optimization each completed 96 candidate-model evaluations and selected Configuration 1, consisting of 32 LSTM units, a dropout rate of 0.10, 16 dense units, a learning rate of 0.0005, and a batch size of 16. Because all methods evaluated the same complete search space using identical seeds and aggregation criteria, they selected the same configuration and produced identical validation results. Their main difference was search efficiency: Grid Search reached the final-best configuration earliest, followed by Bayesian Optimization and Random Search, as shown in Table 2.

Table 2. Selected Hyperparameter Configurations and Search-Efficiency Results

Parameter	Baseline	Grid Search	Random Search	Bayesian Opt
Configuration ID	–	1	1	1
LSTM units	64	32	32	32
Dropout rate	0.20	0.10	0.10	0.10
Dense units	32	16	16	16
Learning rate	0.0010	0.0005	0.0005	0.0005
Batch size	32	16	16	16
HPO evaluations	–	96	96	96
Mean trial-to-best	–	7.00	19.00	10.33
Mean time-to-best (s)	–	154.26	359.54	214.22

The optimized configuration reduced the mean best validation loss from 0.03958994 ± 0.02975872 for the baseline to 0.00539844±0.00132382. It also reduced mean validation MAPE from 4.335651±2.055216% to 1.848947±0.251343%, as summarized in Table 3. The wide baseline confidence interval resulted from estimation using only three seed-level observations and should be interpreted as high seed-level uncertainty rather than as a feasible negative MAPE range.

Table 3. Validation Performance Across Three Random Seeds

Scenario	MAE Mean ± SD	RMSE Mean ± SD	MAPE Mean ± SD (%)	MAPE 95% CI (%)
Baseline	192.347122 ± 95.307594	268.376640 ± 133.596404	4.335651 ± 2.055216	[−0.769790, 9.441091]
Optimized configuration	76.246728 ± 10.589434	106.425349 ± 13.276577	1.848947 ± 0.251343	[1.224578, 2.473317]

Note: Mean and sample SD were calculated across three random seeds; the 95% confidence interval was reported only for MAPE.

Testing Results

The four LSTM scenarios were evaluated using 37 independent-testing sequences. Because Grid Search, Random Search, and Bayesian Optimization selected the same configuration and were retrained using identical seeds and procedures, they produced identical seed-level and ensemble results and are therefore combined into one optimized-configuration row in Table 4.

Table 4. Independent Testing Performance Across Three Random Seeds

Scenario	MAE Mean ± SD	RMSE Mean ± SD	MAPE Mean ± SD (%)	MAPE 95% CI (%)	Ensemble MAPE (%)
Baseline	142.940291 ± 75.777358	161.882028 ± 75.461780	3.396194 ± 1.785084	[−1.038200, 7.830587]	2.913553
Optimized configuration selected by Grid Search, Random Search, and Bayesian Optimization	115.048896 ± 35.768145	134.704708 ± 35.182205	2.785052 ± 0.862715	[0.641948, 4.928155]	2.696161
Naïve persistence	56.388474	71.059018	1.358325	–	1.358325

Note: Mean and sample SD were calculated across three seeds for the LSTM scenarios. Naïve persistence was deterministic and therefore had no seed-level confidence interval.

The optimized configuration reduced mean testing MAPE by 17.99% relative to the baseline and reduced ensemble MAPE from 2.913553% to 2.696161%. However, naïve persistence achieved the lowest MAPE of 1.358325%. Differences between the baseline and optimized LSTM were not statistically significant after Holm correction, whereas persistence produced significantly lower errors than both LSTM configurations. The Holm-adjusted Diebold–Mariano p-values were 0.007476 against the baseline and 0.000846 against the optimized configuration; the corresponding Wilcoxon p-values were 0.000849 and 0.000170. These findings apply only to the 37 examined testing dates.

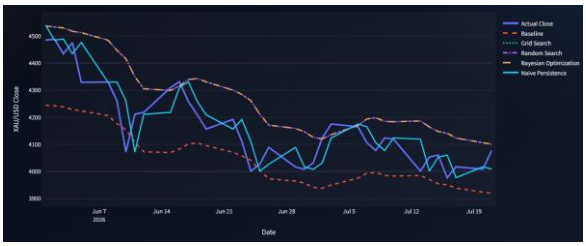

Figure 1. Actual and Predicted XAU/USD Closing Prices on the Independent Testing Dataset

Figure 1 shows that the three optimization methods produced overlapping ensemble predictions because they used the same selected configuration, random seeds, and retraining procedure. The optimized LSTM followed the actual series more closely than the baseline, although naïve persistence produced the lowest testing errors.

Post-Selection Walk-Forward Diagnostic Results

The baseline and optimized configuration were evaluated using a five-fold expanding-window post-selection walk-forward diagnostic. Each configuration was retrained using seeds 42, 123, and 2024 within each fold, while hyperparameter optimization was not repeated. Because Grid Search, Random Search, and Bayesian Optimization selected the same configuration, their results were identical and are combined into one optimized-configuration row in Table 5.

Table 5. Summary of the Five-Fold Post-Selection Walk-Forward Diagnostic

Scenario	Mean MAE ± SD	Mean RMSE ± SD	Mean MAPE ± SD (%)
Baseline	212.237494 ± 193.089550	233.050722 ± 209.649579	6.024930 ± 4.085232
Optimized configuration	286.016617 ± 386.436321	305.990663 ± 397.755120	7.484236 ± 7.900652

Note: Mean and sample SD were calculated across five chronological folds using ensemble predictions from three random seeds.

The baseline achieved a lower mean MAPE than the optimized configuration, whereas the optimized configuration performed better during independent testing. This difference indicates that relative model performance depended on the evaluated chronological period. Because hyperparameter optimization was performed before the walk-forward procedure and was not repeated within each fold, these results represent a post-selection fixed-configuration diagnostic and should be interpreted descriptively rather than as nested model-selection performance or evidence of temporal superiority.

System Interface Implementation

The forecasting pipeline was implemented in a Streamlit-based application [21] that reproduced the Google Colab preprocessing, 30×5 input structure, model configurations, seed-level predictions, ensemble evaluation, and post-selection walk-forward outputs. The application supported baseline and HPO execution, result visualization and management, and next-trading-day forecasting. The forecasting interface used the latest 30 completed trading observations, averaged predictions from models trained using seeds 42, 123, and 2024, and adjusted the forecast date to the next business day. Figure 2 presents the model-comparison and forecasting interfaces.

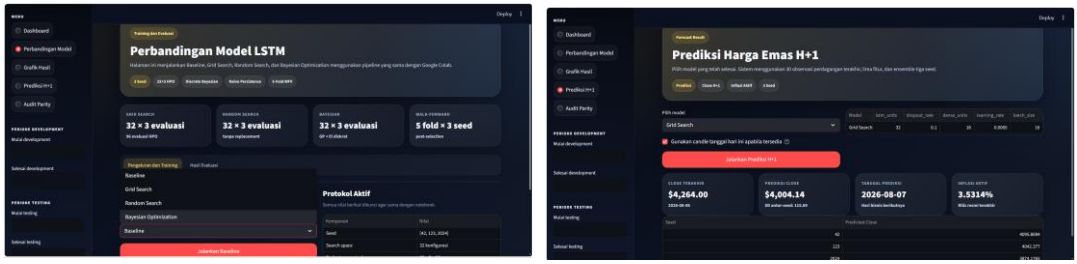

Figure 2. Main Interfaces of the Streamlit-Based Forecasting System: (a) Model Comparison; (b) Next-Trading-Day Forecasting

System Testing Results

Black-box testing evaluated 22 individual scenarios covering navigation, period validation, data processing, model execution, evaluation, result management, and next-trading-day forecasting. Table 6 summarizes the test inputs, expected outputs, observed outputs, and execution status.

Table 6. Individual Black-Box Testing Results

ID	Scenario and Input	Expected Output	Observed Actual Output	Status
T01	Open all navigation menus.	All application pages open successfully.	All five application pages opened successfully.	Passed
T02	Open the Dashboard using valid credentials.	Dashboard data are displayed.	Period settings, performance metrics, dataset summary, and five input features were displayed.	Passed
T03	Apply the default development and testing periods.	Dataset and sequence counts match the Google Colab results.	The dataset contained 1,705 observations. The development, fit, validation, and testing subsets contained 1,668, 1,334, 334, and 37 observations, respectively. The fit, validation, and testing subsets produced 1,304, 334, and 37 sequences.	Passed
T04	Enter valid custom development and testing periods.	Data are reprocessed according to the selected periods.	The selected periods were accepted, the data were reprocessed, and the processing audit was updated.	Passed
T05	Enter an invalid development period.	A development-period warning is displayed.	A warning concerning the invalid development period was displayed.	Passed
T06	Enter overlapping development and testing periods.	An overlap warning is displayed.	A warning concerning the overlapping periods was displayed.	Passed
T07	Set the testing start date after the testing end date.	A testing-period warning is displayed.	A warning concerning the invalid testing-period order was displayed.	Passed
T08	Set the testing end date after the current date.	A future-date warning is displayed.	A warning concerning the future testing date was displayed.	Passed
T09	Leave the API key empty or enter an invalid API key.	An API-key or data-loading error message is displayed.	An API-key or data-loading error message was displayed.	Passed
T10	Reload data using valid credentials.	XAU/USD and inflation data are loaded.	The OHLC market data and release-date-aligned inflation data were loaded and displayed.	Passed
T11	Run the predefined baseline model.	The three-seed evaluation is completed.	Training using three seeds, independent testing, and the five-fold post-selection walk-forward diagnostic were completed.	Passed
T12	Run Grid Search.	All 96 candidate-model evaluations are completed.	All 32 configurations were evaluated using three seeds, resulting in 96 evaluations, and Configuration 1 was selected.	Passed

ID	Scenario and Input	Expected Output	Observed Actual Output	Status
T13	Run Random Search.	Thirty-two unique configurations are evaluated for each seed without replacement.	Ninety-six evaluations were completed without replacement, and Configuration 1 was selected.	Passed
T14	Run Bayesian Optimization.	Thirty-two evaluations are completed for each seed.	Five initial configurations and 27 subsequent Bayesian trials were completed for each seed, and Configuration 1 was selected.	Passed
T15	Open the evaluation-results page.	The evaluation-result tables are displayed.	The hyperparameter, validation, independent-testing, walk-forward, and search-efficiency tables were displayed.	Passed
T16	Open the result-graphs page.	The evaluation graphs are displayed.	The actual-versus-predicted graph, testing-MAPE graph, and walk-forward graph were displayed.	Passed
T17	Open the parity-audit page.	The parity-audit results are displayed.	The dataset, HPO, Google Colab, statistical-testing, and volatility-regime audit results were displayed.	Passed
T18	Run next-trading-day forecasting.	Seed-level and ensemble forecasts are displayed.	Three seed-level predictions, the ensemble forecast, standard deviation, most recent closing price, and forecast date were displayed.	Passed
T19	Run forecasting when the most recent completed trading day is Friday.	The forecast date is adjusted to Monday.	The forecast date was automatically set to the following Monday.	Passed
T20	Refresh or restart the application.	Previously saved results remain available.	The saved experimental results were successfully reloaded after the application was refreshed or restarted.	Passed
T21	Delete one result and perform a full reset.	The selected and stored results are removed.	The selected result was deleted, and the full-reset function cleared the stored results and model checkpoints.	Passed
T22	Download the experimental-result export.	A ZIP export file is generated.	The ZIP export file was generated and made available for download.	Passed

Note: Each scenario was executed individually and classified as passed when the observed output matched the expected output. Detailed execution records were retained by the authors.

All 22 scenarios passed, confirming that the application handled data and period validation, model execution, evaluation, result management, and next-trading-day forecasting as specified. The dataset counts, sequence counts, HPO evaluations, and selected configuration were also consistent with the Google Colab reference results.

Model Comparison and Discussion

Under the equal-budget, multi-seed protocol, Grid Search, Random Search, and Bayesian Optimization evaluated the same complete search space and selected the same configuration. Consequently, they produced identical validation, independent-testing, and post-selection walk-forward results. Their main difference was search efficiency: Grid Search reached the final-best configuration earliest because its fixed evaluation order encountered that configuration relatively early in the examined discrete search space. This result should not be generalized to larger, continuous, or differently ordered search spaces.

Hyperparameter optimization improved the LSTM relative to the predefined baseline during validation and independent testing, but the difference between the baseline and optimized LSTM was not statistically significant after Holm correction. In contrast, naïve persistence produced significantly lower independent-testing errors than both LSTM configurations. Thus, improvement over another LSTM configuration did not establish practical forecasting value without comparison against a simple benchmark.

Table 7. Ensemble MAPE by Volatility Regime

Volatility regime	Number of dates	Baseline MAPE (%)	Optimized LSTM MAPE (%)	Persistence MAPE (%)
Low volatility	12	3.013185	2.237867	0.306737
Medium volatility	12	3.373460	2.202065	1.186477
High volatility	13	2.397057	3.575291	2.487650

Persistence achieved the lowest MAPE in the low- and medium-volatility regimes, whereas the baseline achieved the lowest MAPE in the high-volatility regime. The optimized configuration performed worst during high volatility, suggesting that the configuration selected from the main validation period was less responsive to abrupt price movements. Because the regimes were defined ex post from only 37 testing dates, these results should be interpreted descriptively.

The post-selection walk-forward diagnostic produced a different ranking: the optimized configuration performed better during independent testing, while the baseline achieved lower average errors across the five chronological folds. This inconsistency indicates that model performance depended on the evaluated period. Because hyperparameter optimization was not repeated within each fold, the walk-forward results do not represent nested model-selection performance or evidence of temporal superiority. Overall, the experiment demonstrates that equalizing search spaces, budgets, seeds, and selection criteria can change the interpretation of an HPO comparison, while the strong persistence results emphasize the importance of simple forecasting benchmarks.

4. Conclusion

This study compared Grid Search, Random Search, and Bayesian Optimization for tuning an LSTM model to forecast the XAU/USD closing price for the next trading day under an equal-budget, multi-seed protocol. Each method evaluated the same 32 configurations using three seeds and selected the same final configuration. Consequently, all methods produced identical forecasting results, with a mean independent-testing MAPE of 2.785052% and an ensemble MAPE of 2.696161%.

The optimized configuration achieved lower independent-testing errors than the predefined LSTM baseline, whose ensemble MAPE was 2.913553%, but did not outperform naïve persistence, which achieved the lowest MAPE of 1.358325%. The optimization methods therefore differed mainly in search efficiency rather than final accuracy. The post-selection walk-forward diagnostic also produced different rankings across chronological folds, indicating that the results depended on the evaluated period and should not be interpreted as evidence of general or temporal superiority. The main contribution of this study is a controlled HPO comparison that equalized the search space, evaluation budget, seeds, training conditions, and selection criteria while including a simple forecasting benchmark. The procedures were also implemented in a Streamlit application, and all 22 black-box scenarios passed. These findings are limited to the examined dataset, input features, 32-configuration search space, three seeds, 37-date testing period, five walk-forward folds, and computational environments. Future work should examine larger search spaces, additional seeds, longer testing periods, alternative benchmarks, and nested walk-forward optimization.

References

[1] H. Ben Ameur, F. Jamaani, and M. N. A. Alfoul, "Examining the safe-haven and hedge capabilities of gold and cryptocurrencies: A GARCH and regression quantiles approach in geopolitical and market extremes," *Heliyon*, vol. 10, no. 22, p. e40400, 2024, doi: 10.1016/j.heliyon.2024.e40400.

[2] M. Ryan, S. Corbet, and L. Oxley, "Is gold always a safe haven?," *Finance Research Letters*, vol. 64, p. 105438, Jun. 2024, doi: 10.1016/j.frl.2024.105438.

[3] K. Hisan, "Pengaruh Inflasi, Bi Rate Dan Nilai Tukar Rupiah Terhadap Harga Emas Di Indonesia," *Jemasi: Jurnal Ekonomi Manajemen dan Akuntansi*, vol. 21, no. 2, pp. 207–219, 2025, doi: 10.35449/jemasi.v21i2.1024.

[4] A. Amini and R. Kalantari, "Gold price prediction by a CNN-Bi-LSTM model along with automatic parameter tuning," *PLoS ONE*, vol. 19, no. 3, pp. 1–17, 2024, doi: 10.1371/journal.pone.0298426.

[5] M. R. Nurhambali, Y. Angraini, and A. Fitrianto, "Implementation of Long Short-Term Memory for Gold Prices Forecasting," *Malaysian Journal of Mathematical Sciences*, vol. 18, no. 2, pp. 399–422, 2024, doi: 10.47836/mjms.18.2.11.

[6] M. Owen, V. Vincent, R. Br Ambarita, and E. Indra, "Implementasi Metode Long Short Term Memory Untuk Memprediksi Pergerakan Nilai Harga Emas," *Jurnal Teknik Informasi dan Komputer (Tekinkom)*, vol. 5, no. 1, pp. 96–104, 2022, doi: 10.37600/tekinkom.v5i1.507.

[7] B. Bischl *et al.*, "Hyperparameter optimization: Foundations, algorithms, best practices, and open challenges," *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, vol. 13, no. 2, pp. 1–43, 2023, doi: 10.1002/widm.1484.

[8] B. Bischl *et al.*, "Hyperparameter optimization: Foundations, algorithms, best practices, and open challenges," *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, vol. 13, no. 2, pp. 1–43, 2023, doi: 10.1002/widm.1484.

[9] A. M. Vincent and P. Jidesh, "An improved hyperparameter optimization framework for AutoML systems using evolutionary algorithms," *Scientific Reports*, vol. 13, no. 1, pp. 1–19, 2023, doi: 10.1038/s41598-023-32027-3.

[10] Y. Zhao, W. Zhang, and X. Liu, "Grid search with a weighted error function: Hyper-parameter optimization for financial time series forecasting," *Applied Soft Computing*, vol. 154, p. 111362, 2024, doi: 10.1016/j.asoc.2024.111362.

[11] B. Jiang, H. Gong, H. Qin, and M. Zhu, "Attention-LSTM architecture combined with Bayesian hyperparameter optimization for indoor temperature prediction," *Building and Environment*, vol. 224, p. 109536, Oct. 2022, doi: 10.1016/j.buildenv.2022.109536.

[12] W. Chen, W. Hussain, F. Cauteruccio, and X. Zhang, "Deep Learning for Financial Time Series Prediction: A State-of-the-Art Review of Standalone and Hybrid Models," *Computer Modeling in Engineering \& Sciences*, vol. 139, no. 1, pp. 187–224, 2024, doi: 10.32604/cmes.2023.031388.

[13] H. Hewamalage, K. Ackermann, and C. Bergmeir, "Forecast evaluation for data scientists: common pitfalls and best practices," *Data Mining and Knowledge Discovery*, vol. 37, no. 2, pp. 788–832, 2023, doi: 10.1007/s10618-022-00894-5.

[14] X. Qiu *et al.*, "TFB: Towards Comprehensive and Fair Benchmarking of Time Series Forecasting Methods," *Proceedings of the VLDB Endowment*, vol. 17, no. 9, pp. 2363–2377, 2024, doi: 10.14778/3665844.3665863.

[15] Afridon, M., Tedyyana, A., Ratnawati, F., Julianto, A., & Faizi, M. N. (2024). Optimizing Data Security in Computer-Assisted Test Applications Through the Advanced Encryption Standard 256-Bit Cipher Block Chaining. *International Journal of Advanced Computer Science & Applications*, *15*(8), 163.

[16] S. Makridakis, E. Spiliotis, and V. Assimakopoulos, "The M4 Competition: 100,000 Time Series and 61 Forecasting Methods," *International Journal of Forecasting*, vol. 36, no. 1, pp. 54–74, 2020, doi: 10.1016/j.ijforecast.2019.04.014.

[17] "API Documentation," Twelve Data. Accessed: Aug. 06, 2026. [Online]. Available: https://twelvedata.com/docs/introduction/quickstart

[18] "Consumer Price Index for All Urban Consumers: All Items in U.S. City Average [CPIAUCNS]," Federal Reserve Bank of St. Louis. Accessed: Aug. 06, 2026. [Online]. Available: https://fred.stlouisfed.org/series/CPIAUCNS

[19] I Gusti Agung Putu Mahendra, Muhammad Ikhsan Wibowo, and I Gusti Ayu Nandia Lestari, "Comparative Forecasting of Kalimantan Palm Oil Production Using Classical, Machine Learning, and Deep Learning Models," *JURNAL TEKNOLOGI DAN OPEN SOURCE*, vol. 9, no. 1, pp. 290–301, Jun. 2026, doi: 10.36378/jtos.v9i1.5765.

[20] "Keras: The High-Level API for TensorFlow," TensorFlow. Accessed: Aug. 06, 2026. [Online]. Available: https://www.tensorflow.org/guide/keras

[21] "Streamlit Documentation," Streamlit. Accessed: Aug. 06, 2026. [Online]. Available: https://docs.streamlit.io/

[22] L. Coroneo and F. Iacone, "Testing for Equal Predictive Accuracy with Strong Dependence," *International Journal of Forecasting*, vol. 41, no. 3, pp. 1073–1092, 2025, doi: 10.1016/j.ijforecast.2024.11.003.

Acknowledgements

The authors thank the Department of Information Systems, Muria Kudus University, for its academic guidance and institutional support.