INTL JOURNAL OF ELECTRONICS AND TELECOMMUNICATIONS, 2026, VOL. 72, NO. 3, PP. 1-12
Manuscript received April 02, 2026; revised June, 2026. doi: 10.24425/ijet.2026.157958

MO-RSAR: multi-objective hyperparameter optimization of RSAR for financial time-series forecasting

Maja Czyżewska

***Abstract*—This paper empirically compares four architectures for financial time-series forecasting: LSTM, CNN, the original Regularized Self-Attention Regression (RSAR) model, and a multi-objective optimized RSAR variant, denoted MO-RSAR, obtained using the Non-dominated Sorting Genetic Algorithm II (NSGA-II). The models are evaluated on six datasets covering Forex, equity index and cryptocurrency markets, for short and long horizons. All models share a common preprocessing pipeline and evaluation framework and are assessed using standard error metrics, with emphasis on Mean Absolute Percentage Error (MAPE). MO-RSAR yields the lowest average prediction error across all datasets and provides significant gains for longer, more volatile horizons, while simpler architectures remain competitive for short-term forecasts. The key methodological contribution is the first empirical integration of the RSAR architecture with NSGA-II-based multi-objective hyperparameter optimization for financial time-series forecasting. The proposed framework treats RSAR configuration as a bi-objective search over accuracy and generalization (via the train-validation gap), and evaluates the resulting model under a unified protocol across heterogeneous markets and horizons.**

***Keywords*—LSTM; CNN; RSAR; NSGA-II; multi-objective optimization; deep learning; financial time-series forecasting; evolutionary algorithms**

I. Introduction

FORECASTING financial time series is a significant challenge due to nonlinear dynamics, regime shifts and high volatility observed in real markets [17][18]. Traditional econometric models often rely on restrictive assumptions about linearity and stationarity, which puts them at a disadvantage when it comes to capturing complex temporal dependencies commonly found in financial data. In contrast, deep learning architectures can learn nonlinear relations directly from data and have, in recent years, become a popular tool for modelling financial time series.

Among these architectures, Long Short-Term Memory (LSTM) networks are widely used to capture long-range temporal dependencies, while Convolutional Neural Networks (CNNs) are effective at extracting local patterns. Both approaches have been successfully applied to a variety of financial instruments, however their performance is often unstable across different markets and forecast horizons. Moreover, they may struggle to focus selectively on the most informative time steps within a sequence, because all observations are processed in a largely uniform manner.

Attention-based mechanisms address this limitation by prioritizing time steps within the input window. In practice, they learn which observations have a strong influence on the final prediction and which are relatively uninformative, effectively downweighing or ignoring the latter. The Regularized Self-Attention Regression (RSAR) model proposed by Zhou et al. combines an LSTM layer, a self-attention mechanism and a CNN component, together with explicit L1/L2 regularization, in order to enhance both predictive accuracy and regularization in financial price forecasting [1]. While RSAR has been shown to outperform simpler baseline models on precious metal prices, its effectiveness on other asset classes and for different forecast horizons is still not systematically evaluated.

Another key factor that influences model's performance is network configuration, i.e., hyperparameter selection. Deep neural networks typically involve many hyperparameters and choosing them appropriately is a difficult task on its own. Most publications focus solely on single-objective tuning based on validation error, which can lead to overfitting or reduce model's ability to generalize. To solve these problems, multi-objective optimization using evolutionary algorithms, such as Non-dominated Sorting Genetic Algorithm II (NSGA-II) can be used. It offers a principled way to search the hyperparameter space while simultaneously optimizing multiple, potentially conflicting objectives.

This paper focuses on an empirical comparison of four architectures for financial time-series forecasting: a baseline LSTM network, a CNN model, the original RSAR architecture, and a multi-objective optimized RSAR variant denoted as MO-RSAR, in which key hyperparameters are selected using NSGA-II. The models are evaluated on six real-world datasets spanning Forex, equity index and cryptocurrency markets, and are tested on both short and long forecast horizons. All networks share a common data preprocessing pipeline and a comparable training and evaluation framework, and their performance is assessed using standard error metrics, with particular emphasis on Mean Absolute Percentage Error (MAPE). This setup makes it possible to isolate the impact of architectural design and hyperparameter optimization on forecasting accuracy across different market conditions and horizons.

This work was financed by the Military University of Technology (WAT), Warsaw, Poland, under Project No. UGB 531-000091-W500-22.

Maja Czyżewska is with Military University of Technology, Warsaw, Poland (e-mail: maja.czyzewska@wat.edu.pl).

The main contribution of this study is the introduction of MO-RSAR, a multi-objective extension of the attention-based RSAR architecture for financial time-series forecasting. Unlike standard tuning procedures that optimize a single validation metric, the proposed approach uses NSGA-II to select RSAR configurations by jointly minimizing out-of-sample error and an explicit generalization proxy, namely the train-validation error gap. To the best of the author's knowledge, this is the first study that combines RSAR with NSGA-II in a multi-objective setting and evaluates the resulting model consistently across Forex, equity index and cryptocurrency markets under a unified experimental protocol.

II. Related work

The development of deep learning methods has had a substantial impact on financial time-series forecasting. Classical statistical models and traditional machine-learning algorithms often struggle to capture the complex nonlinear dependencies present in market data [19]. In recent years, numerous studies have shown that deep neural networks can effectively extract patterns from financial time series and outperform standard econometric approaches [1]. In particular, recurrent and convolutional neural networks, as well as their hybrids, have become important tools for forecasting stock prices, exchange rates, and commodity prices.

A. LSTM-based models in financial forecasting

A major milestone in recurrent architectures was the introduction of the Long Short-Term Memory (LSTM) network by Hochreiter and Schmidhuber (1997) [4]. LSTM was proposed as a remedy for the vanishing-gradient problem in standard RNNs, which had been analyzed earlier by Hochreiter (1991) [2] and Bengio et al. (1994) [3]. These works showed that gradient-based learning has severe difficulty capturing long-term dependencies: as sequence length increases, the influence of distant information quickly vanishes, making effective training difficult. The LSTM architecture, equipped with input, forget and output gates, addresses this limitation by regulating information flow, enabling recurrent networks to retain relevant information over longer horizons and discard irrelevant signals [2]-[4].

The effectiveness of LSTM in financial applications has been demonstrated in numerous studies. Fischer and Krauss (2017) applied LSTM networks to predict the direction of stock price movements for S&P 500 constituents. Their LSTM model significantly outperformed traditional classifiers, such as random forests and logistic regression, achieving higher average daily returns and a markedly better Sharpe ratio. They also reported that an investment strategy based on LSTM predictions yielded positive returns, whereas analogous portfolios constructed using conventional machine-learning models generated losses [5]. This suggests that the memory mechanism in LSTM enables the model to capture temporal patterns that simpler or memoryless models may fail to detect.

More recent work focuses on improving both LSTM architectures and their training procedures for financial data. Gülmez (2023) proposed using the Artificial Rabbits Optimization (ARO) metaheuristic to tune LSTM hyperparameters for predicting stock prices from the Dow Jones Industrial Average index. Several variants of LSTM were compared (including 1D/2D/3D configurations, a GA-optimized version and the ARO-based version), and the LSTM–ARO model achieved the lowest MSE/MAE and the highest R^2 among all tested networks [6]. This highlights how carefully chosen depth, number of units and learning parameters can substantially increase forecasting accuracy.

At the same time, Li (2024) stresses the risk of overfitting: in a comparison between LSTM and regression-based models, the author shows that LSTM may overfit when hyperparameters are poorly chosen, especially on relatively small datasets. Nevertheless, when properly tuned, LSTM clearly outperforms linear, Lasso, Ridge and polynomial regression in forecasting stock prices of major automotive companies, achieving the lowest RMSE and MAPE among all compared methods [7].

LSTM networks have also been successfully applied beyond equity markets. Madhika, Kusrini and Hidayat (2023) used LSTM to forecast gold prices and found that the LSTM model achieved much better accuracy than an ARIMA benchmark. In their experiments, LSTM attained very low error (RMSE $\approx$ 8.124, MAPE $\approx$ 2.3%), while the best ARIMA(0,1,1) configuration produced noticeably higher errors [8]. The performance gap is particularly visible for longer horizons: ARIMA can handle short-term linear trends, but LSTM better captures nonlinear and long-term patterns. Overall, the literature indicates that LSTM networks have become one of the leading tools for financial time-series forecasting, often outperforming simpler neural models and traditional statistical methods [5]-[8], [1].

B. CNN and CNN–LSTM hybrids for time-series data

In parallel with recurrent networks, substantial progress has been made in convolutional neural networks (CNNs). The CNN architecture originated from the Neocognitron model proposed by Fukushima (1980) [9] and was popularized by LeCun et al. (1998) [10] in handwritten digit recognition. CNNs perform hierarchical feature extraction via local convolutional filters, enabling robust pattern recognition that is relatively invariant to small translations and distortions.

Although CNNs are most often associated with image analysis, the same idea can be applied to time series using one-dimensional convolutions (1D CNN). Existing surveys and empirical studies indicate that, while 1D CNNs are less frequently used as stand-alone models in finance, they can still support time-series analysis by extracting local temporal patterns [11][12]. Velastegui et al. (2020) proposed a pure 1D CNN for time-series prediction, including financial data, and showed that the model can capture short-term patterns in price changes and compete with simpler neural baselines [12]. An important advantage of CNNs in this context is parameter sharing, which reduces model size, and pooling layers, which average out local fluctuations. These properties are valuable for noisy financial series, where short-lived spikes and drops can obscure the underlying trend.

Hybrid architectures combining CNN and LSTM are particularly attractive in this setting. The main idea is to let CNN layers perform preliminary feature extraction on short windows (e.g., several days of prices), and then feed these features into LSTM layers that model longer-term temporal dependencies. Lu et al. (2020) proposed a CNN-LSTM model for stock price prediction and conducted a comprehensive comparison with

several alternatives: MLP, stand-alone LSTM, stand-alone CNN and a simple CNN-RNN hybrid. The CNN-LSTM architecture achieved the highest predictive accuracy among all tested models [13]. Convolutional layers effectively extracted local characteristics from historical data, while LSTM layers learned longer-term patterns from the resulting feature sequences. This combination led to more accurate price forecasts across multiple stocks.

Li et al. (2020) observe in a broader survey that CNNs are increasingly used not only in image-related tasks but also in forecasting traffic flows, weather patterns and other time-series phenomena. This confirms the general applicability of convolutional models to sequential data. In summary, CNN-LSTM hybrids constitute a natural extension of pure LSTM or CNN models, enhancing the ability to recognize complex patterns in financial time series and improving forecasting performance [11][12][13].

C. Attention-based models and RSAR

One of the most recent trends in financial forecasting is the integration of attention mechanisms, originally introduced in the Transformer architecture by Vaswani et al. (2017) [14]. Attention allows a model to dynamically prioritize information in the input sequence by assigning higher weights to important elements and lower weights to less relevant observations. In sequence models, self-attention helps overcome limitations of architectures that may struggle to capture local and global patterns simultaneously.

Zhou et al. (2020) applied self-attention to precious metal price prediction and proposed the Regularized Self-Attention Regression (RSAR) model. RSAR combines three key components: an LSTM layer to capture long-term temporal dependencies, a self-attention mechanism to adaptively weight time steps in the input window, and a CNN layer to extract local patterns from the attention-weighted representation. Additionally, RSAR employs L1 regularization on biases, L2 regularization on weight matrices and a dedicated penalty on the attention map to prevent the attention distribution from collapsing onto a single time step [1].

Experimental results reported by Zhou et al. show that RSAR achieves strong predictive performance in precious metal forecasting, outperforming traditional machine-learning models (e.g., linear regression, support vector regression) as well as simpler deep-learning baselines such as stand-alone LSTM, stand-alone CNN and basic LSTM-CNN hybrids. This suggests that incorporating self-attention into hybrid recurrent-convolutional architectures can improve forecasting accuracy in financial prediction tasks [1]. More broadly, RSAR can be viewed as part of a wider trend towards incorporating attention-based or Transformer-inspired components into classical time-series models in finance [15][16].

D. Hyperparameter optimization in deep financial models

Across the literature, hyperparameter selection is consistently identified as a critical factor for the performance of deep models in finance. Studies such as Gülmez [6] and Li [7] show that metaheuristic optimization (e.g., evolutionary or swarm-based algorithms) can substantially improve LSTM forecasting accuracy by automatically tuning network depth, number of units and learning parameters. At the same time, these works underline the risk of overfitting when model capacity is high and hyperparameters are adjusted by trial and error [6][7].

In Zhou et al., the RSAR architecture is validated only on gold and palladium prices, using a small set of manually chosen configurations [1]. While sufficient to demonstrate the potential of RSAR, this approach leaves open important methodological questions, in particular how to adapt RSAR to markets with different volatility and noise characteristics, and how to explicitly control the trade-off between predictive accuracy and overfitting. Most existing studies treat hyperparameter tuning as a single-objective problem driven by a validation error metric, without modelling this trade-off in a principled manner.

The empirical study in this paper builds directly on these observations. LSTM, CNN, and CNN-LSTM-type hybrids are adopted as strong baselines [5]-[8], [11]-[13], while RSAR serves as the attention-based hybrid reference architecture [1]. Building on this gap, the paper introduces MO-RSAR, a multi-objective extension of RSAR in which NSGA-II is used to optimize hyperparameters with respect to both validation error and the train-validation gap. This provides a more systematic way to balance predictive accuracy against overfitting risk than manual, single-objective sensitivity analysis.

III. Objectives

The overall aim of this study is to evaluate, under a common experimental setup, the effectiveness of several deep learning architectures for financial time-series forecasting, with particular emphasis on the role of attention mechanisms and multi-objective hyperparameter optimization. The experimental design, outlined in the Introduction, is based on four neural network models and six real-world datasets from Forex, equity index, and cryptocurrency markets, with both short- and long-horizon forecasts.

Within this framework, the study addresses the following research questions:

1) *How do LSTM, CNN, RSAR, and MO-RSAR compare in terms of out-of-sample forecasting accuracy across the six datasets?*
2) *To what extent does NSGA-II-based multi-objective hyperparameter optimization improve RSAR out-of-sample performance relative to the original RSAR configuration?*
3) *Under which conditions (market type and data window length) does MO-RSAR provide the largest gains over the baseline architectures, and when do simpler models remain competitive?*

The main contributions can be summarized as follows:

1) *It proposes, to the best of the author's knowledge, the first empirical framework that integrates the RSAR architecture with NSGA-II-based multi-objective hyperparameter optimization for financial time-series forecasting, referred to in this paper as MO-RSAR.*
2) *It formulates RSAR configuration as a multi-objective search that simultaneously minimizes out-of-sample error and the train-validation gap, providing an explicit accuracy-generalization trade-off rather than single-metric tuning.*
3) *It delivers an empirical benchmark across Forex, equity index, and cryptocurrency markets, showing when MO-*

RSAR yields substantial gains and when simpler baselines remain preferable, especially for short and noisy windows.

IV. METHODS

A. Data sets

The empirical analysis is based on six time series of daily closing prices. The series are selected from three different classes of financial instruments so that the models can be evaluated under diverse market conditions. Each market has its own dynamics and reacts to different external factors, which makes the comparison more comprehensive. For each of the three instruments, two data ranges are considered: a longer historical sample and a shorter, more recent window. The following datasets are used:

1. *EUR/USD_all, 03.10.1975-12.03.2025*
2. *EUR/USD_recent, 12.03.2005-12.03.2025*
3. *SP500_all, 26.12.1979-12.03.2025*
4. *SP500_recent, 12.03.2005-12.03.2025*
5. *BTC_long, 15.04.2024-15.04.2025*
6. *BTC_recent, 15.12.2024-15.04.2025*

All series are downloaded from Investing.com as daily close quotes [21]. Each dataset is standardized using the mean and standard deviation computed on the full sample. Afterwards, each series is split chronologically into a training segment and an out-of-sample hold-out segment using a fixed cut-off date chosen separately for each instrument. Models are fitted on the training segment and assessed on the out-of-sample segment. For consistency, the same hold-out split is also used as the validation reference in the NSGA-II procedure. Therefore, the terms validation and test refer to the same out-of-sample hold-out period, depending on whether it is used for hyperparameter selection or for final performance reporting.

B. Input representation and forecasting task

In all experiments the models solve a one-step-ahead forecasting problem: given an input window of past prices, the network predicts the next-day closing price.

The input sequence is constructed using a sliding window of length $look\_back = 7$. For each time step t, the input vector contains the last seven standardized closing prices:

$$(x_{t-6}, x_{t-5}, \dots, x_t),$$

and the target value is the next observation x_{t+1}. This representation is identical across all four architectures, which makes it possible to attribute differences in accuracy to the model design rather than to feature engineering.

C. Model architectures

Four neural network architectures are compared under a common experimental setup: a baseline LSTM network, a CNN model, the original RSAR architecture and its multi-objective extension, denoted as MO-RSAR, whose hyperparameters are optimized using the NSGA-II algorithm. All models are implemented in Python using the Keras/TensorFlow framework and use a linear activation layer as the output layer, since the task is a regression problem on continuous price values.

1) LSTM baseline

As a natural baseline for financial time-series forecasting, a recurrent Long Short-Term Memory (LSTM) network is used. LSTM architectures are specifically designed for sequential data and are able to capture temporal dependencies over longer horizons thanks to their internal gating mechanism. The input, forget and output gates control how information is stored, updated and propagated through time, which is particularly important in financial data where events separated by several days may still influence the current dynamics.

In this study, the LSTM model consists of two stacked LSTM layers with 6 units each. The first layer returns the full sequence of hidden states, while the second returns only the last hidden state, which is then passed to a fully connected layer with linear activation to produce the final prediction. This configuration provides sufficient capacity to model nonlinear temporal patterns within a 7-step input window, while keeping the number of parameters moderate and reducing the risk of overfitting.

2) CNN baseline

The convolutional baseline treats each input window as a short one-dimensional signal and applies two convolution–pooling blocks. Each Conv1D layer with 64 filters and a kernel size of 2 scans the window and extracts local patterns in price changes. The subsequent max-pooling layers reduce the temporal resolution and dimensionality of the feature maps, with the expectation that the most important local features are preserved, while less informative fluctuations are suppressed.

After the convolution-pooling blocks, a dropout layer is applied. Dropout randomly switches off a fixed proportion of neurons during training, which forces the network to rely on a more distributed representation of information and helps to reduce overfitting and improve generalization. The feature maps are then passed through a flattening layer that reshapes them into a single feature vector. This vector is processed by a fully connected dense layer with 250 units, followed by an additional dropout layer and a ReLU activation. Finally, a single output neuron with linear activation produces the next-day price forecast.

3) Regularized Self-Attention Regression (RSAR)

The Regularized Self-Attention Regression (RSAR) model has a hybrid architecture that combines recurrent and convolutional components, linked through a self-attention layer and equipped with several regularization mechanisms. The motivation behind this design is to exploit the strengths of LSTM networks in modelling temporal dependencies and the ability of CNNs to extract local patterns, while the attention mechanism prioritizes the most informative time steps within the input window. Compared with standalone LSTM or CNN models, RSAR has higher capacity and computational cost, but may achieve more accurate forecasts in settings where prediction quality is prioritized over training efficiency.

At the first stage, an LSTM layer processes the input sequence of past prices and encodes it into a sequence of hidden states. This recurrent part is responsible for capturing temporal dependencies across the seven-day input window, including relationships between time steps that are not adjacent. The sequence of hidden states is then passed to a self-attention layer,

which learns to assign higher weights to more informative time steps and downweight less relevant observations. In this implementation, the attention layer is regularized in three ways: L2 regularization is applied to the weight matrices to limit the magnitude of the parameters, L1 regularization is applied to the bias terms to encourage sparsity, and an additional attention regularization penalty is imposed to prevent the attention distribution from collapsing onto a single time step. Together, these mechanisms help reduce overfitting and keep the effective complexity of the model under control.

The attention-weighted representation is subsequently fed into a one-dimensional convolutional neural network (CNN). The Conv1D layer with 64 filters and a kernel size of 2 scans the sequence of attention-enhanced features and extracts local patterns that may not be fully captured by the recurrent and attention components alone. A max-pooling layer reduces the temporal resolution of the feature maps, retaining the most important local responses while suppressing smaller fluctuations. The resulting feature maps are subsequently regularized by dropout, which randomly disables a fixed proportion of activations during training and thus improves generalization. The final output stage then produces the one-step-ahead price forecast through a linear output neuron.

A detailed mathematical description of the RSAR architecture, including the formulation of the self-attention mechanism and regularization terms, is provided in Zhou et al. [1]. The multi-objective optimization framework underlying MO-RSAR is described in the author's methodological study [20]. Here, RSAR serves as a strong hybrid baseline against which the effect of additional NSGA-II-based multi-objective hyperparameter tuning can be assessed.

4) MO-RSAR: multi-objective extension of RSAR

The fourth architecture is MO-RSAR, a multi-objective extension of RSAR in which key hyperparameters are selected using a multi-objective evolutionary algorithm. Instead of relying on manually chosen settings, the dropout rate, attention regularization weight, the number of LSTM units, and the number of CNN filters are optimized jointly using the Non-dominated Sorting Genetic Algorithm II (NSGA-II). The objectives are to minimize the validation mean squared error and the gap between training and validation error, so that the final configuration achieves a reasonable balance between accuracy and generalization.

Optimization problem formulation:

In this study, the selection of RSAR hyperparameters is formulated as a bi-objective optimization problem. The decision vector is defined as $x = (d, attn, u, f)$, where d denotes the dropout rate, $attn$ is the attention regularization weight, u is the number of LSTM units, and f is the number of CNN filters. For each candidate configuration x, an RSAR model is trained under the fixed training protocol described in Section IV-D and evaluated on the validation split. The optimization objectives are defined as $F(x) = (MSE_{val}(x), Gap(x))$, where $MSE_{val}(x)$ is the validation mean squared error, and $Gap(x)$ is the train-validation gap computed as $| MSE_{train}(x) - MSE_{val}(x) |$. The first objective promotes predictive accuracy, while the second controls generalization by penalizing configurations that fit the training segment much better than the validation segment.

The search is performed within predefined bounds for each decision variable, and NSGA-II is used to approximate the set of Pareto-optimal configurations. From the resulting Pareto front, one configuration is selected per dataset and reported as the MO-RSAR model in the empirical comparison. The detailed mathematical formulation of this framework – including the definition of the decision vector, search space, objective functions, and the NSGA-II search procedure – is presented in the author's methodological study on RSAR and NSGA-II and is only summarized here [20]. In the present work, NSGA-II is used to generate a set of Pareto-optimal RSAR configurations for each dataset. For the empirical comparison, one Pareto-optimal configuration per dataset is automatically designated as the MO-RSAR model and used alongside the LSTM, CNN, and original RSAR baselines.

The dataset-specific Pareto-selected MO-RSAR hyper-parameters are reported in Table I.

TABLE I
PARETO-SELECTED MO-RSAR HYPERPARAMETERS FOR EACH DATASET

Dataset	Dropout d	Attention regularization weight $attn$	LSTM units u	CNN filters f
EUR/USD_all	0.30	5.0×10^{-6}	6	16
EUR_USD_recent	0.40	6.3×10^{-4}	12	70
SP500_all	0.25	1.0×10^{-5}	16	30
SP500_recent	0.26	1.4×10^{-5}	13	25
BTC_long	0.31	3.0×10^{-2}	9	107
BTC_recent	0.24	1.0×10^{-1}	17	98

D. Training protocol and evaluation metrics

All four architectures are trained under a common protocol in order to make the comparison as fair as possible. For each dataset, models are trained on the training segment and evaluated on the out-of-sample segment defined in Section IV-A. The same input representation (7-day sliding window) is used throughout, as described in Section IV-B.

To reduce the impact of different training regimes, all baseline networks are trained for 100 epochs without shuffling, so as to preserve the temporal order of observations during learning. The LSTM and CNN baselines use the RMSprop optimizer and mean squared error (MSE) loss, whereas both RSAR-based architectures use Adam with the same MSE loss. The LSTM, CNN, and original RSAR models are trained with a mini-batch size of 80. In the MO-RSAR procedure, candidate configurations evaluated inside NSGA-II are trained for 20 epochs with a mini-batch size of 50, while the final selected Pareto-optimal configuration is retrained for 100 epochs with the same mini-batch size of 50.

Model performance is evaluated on the out-of-sample test segment using error metrics for regression problems: Root Mean Squared Error (RMSE), Root Mean Absolute Error (RMAE) and Mean Absolute Percentage Error (MAPE). For the given dataset with true prices $y_1, \dots, y_N$ and corresponding predictions $\hat{y}_1, \dots, \hat{y}_N$, the metrics are defined as:

$$RMSE = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2} \quad (1)$$

$$RMAE = \sqrt{\frac{1}{N}\sum_{i=1}^{N}|y_i - \hat{y}_i|} \quad (2)$$

$$MAPE = \frac{100}{N}\sum_{i=1}^{N}\left|\frac{y_i - \hat{y}_i}{y_i}\right| \quad (3)$$

All analyzed price series are strictly positive, so MAPE is well defined for all observations. In the empirical discussion, particular emphasis is placed on MAPE, as it expresses errors in relative terms and allows for a direct comparison of model performance across instruments with different price scales and volatility levels.

For the MO-RSAR configuration, the validation and test segments coincide and refer to the same out-of-sample hold-out split. This hold-out split is used inside NSGA-II to compute the validation MSE and the train-validation gap. After optimization, the selected Pareto-optimal configuration is retrained on the training segment for 100 epochs, and the final results are reported on the same out-of-sample segment for comparison with the LSTM, CNN, and original RSAR baselines. To improve reproducibility, a fixed NumPy random seed (seed = 7) was used in all experiments.

E. Loss-curve approximation (least squares method)

To summarize and compare the observed training dynamics for selected training runs, the observed training loss values $\{(t_i, y_i)\}_{i=1}^{n}$, where t_i denotes the epoch index and y_i the corresponding loss, were approximated with a parametric exponential model:

$$\hat{y}(t;\theta) = c + ae^{-bt}, \qquad a > 0, b > 0, c \geq 0 \quad (4)$$

The parameters $\theta = (a, b, c)$ were estimated using the least squares criterion by minimizing the sum of squared errors (SSE):

$$SSE(\theta) = \sum_{i=1}^{n}\left(y_i - \hat{y}(t_i;\theta)\right)^2 \quad (5)$$

$$\theta^* = \arg\min_{\theta} SSE(\theta) = \arg\min_{\theta}\sum_{i=1}^{n}\left(y_i - \hat{y}(t_i;\theta)\right)^2 \quad (6)$$

Here n is the number of observed epochs, t_i and y_i denote the epoch index and the observed loss respectively. The fitted loss at epoch t_i is denoted by $\hat{y}(t_i;\theta)$ for the exponential model and $\hat{y}(t_i;\phi)$ for the power-law model, where $\theta = (a, b, c)$ and $\phi = (a, b, c)$. The corresponding least squares estimates are θ^* and ϕ^*, respectively. The optimization was performed numerically using Excel Solver. In the exponential model, c is the plateau level, a sets the initial offset above the plateau, and b controls the exponential decay rate. In the power-law model, b governs the heaviness of the tail (slower decay for smaller b).

As an alternative functional form, a power-law model was also tested:

$$\hat{y}(t;\phi) = c + at^{-b}, \qquad a > 0, b > 0, c \geq 0 \quad (7)$$

$$SSE(\phi) = \sum_{i=1}^{n}\left(y_i - \hat{y}(t_i;\phi)\right)^2 \quad (8)$$

$$\phi^* = \arg\min_{\phi} SSE(\phi) = \arg\min_{\phi}\sum_{i=1}^{n}\left(y_i - \hat{y}(t_i;\phi)\right)^2 \quad (9)$$

The exponential specification in (4) was retained for reporting because it consistently yielded a lower SSE than the power-law alternative on the analyzed loss curves.

V. RESULTS

A. Overall comparison

Table II reports the out-of-sample errors of all four architectures (LSTM, CNN, RSAR and MO-RSAR) on the six datasets, measured in terms of RMSE, RMAE and MAPE. The discussion in this section focuses primarily on MAPE, while the remaining metrics follow the same qualitative patterns.

TABLE II
MAPE RESULTS FOR EACH EXPERIMENT AND OVERALL AVERAGE FOR THE MODELS

Dataset	LSTM	RSAR	MO-RSAR	CNN
EUR/USD_all	0.82%	0.95%	0.70%	1.44%
EUR_USD_recent	0.82%	1.34%	1.29%	2.16%
SP500_all	15.24%	5.29%	2.92%	18.3%
SP500_recent	14.64%	7.78%	6.08%	7.41%
BTC_long	3.2%	2.82%	2.87%	2.89%
BTC_recent	2.37%	5.76%	5.57%	3.19%
Average	6.18%	3.99%	3.24%	5.90%

Across all experiments, MO-RSAR achieves the best overall accuracy. It attains the lowest MAPE in three out of six cases and reduces the average MAPE of the original RSAR architecture by approximately 16% as summarized in Table II. The classical RSAR model remains a strong competitor and provides robust performance on most datasets, in particular on the S&P 500 index. The LSTM baseline wins on two datasets, both corresponding to shorter recent windows, while the CNN baseline never achieves the best MAPE but in some settings comes close to the leading models and often tracks the general shape of price movements correctly.

The following subsections discuss the results separately for long historical windows and for shorter, recent samples.

B. Long historical windows

1) EUR/USD (1975-2025)

TABLE III
RESULTS FOR EUR/USD_ALL DATASET (1975-2025)

Model	MAPE	RMAE	RMSE
LSTM	8.22e-01	9.51e-02	1.18e-02
RSAR	9.52e-01	1.01e-01	1.30e-02
MO-RSAR	6.97e-01	8.70e-02	9.83e-03
CNN	1.44e+00	1.24e-01	1.92e-02

The quantitative results for the full EUR/USD sample are reported in Table III. For the full EUR/USD sample spanning almost fifty years, all models achieve relatively low percentage errors, but their ranking differs substantially. The CNN baseline performs the worst, with MAPE around

1.4% and higher RMSE and RMAE values. This suggests that, despite being able to detect local patterns, the convolutional architecture struggles to remain accurate over a very long horizon with changing market regimes.

The LSTM network attains a clearly lower error (MAPE ≈ 0.82%), outperforming both CNN and the original RSAR model, whose MAPE is slightly below 1%, as shown in Table III. One likely reason is that the RSAR configuration was originally tuned for precious metal prices and does not fully exploit its potential on the EUR/USD series without further adaptation.

MO-RSAR achieves the best result on this dataset, with MAPE reduced to approximately 0.7% (Table III). This corresponds to a reduction of about 26% relative to the classical RSAR configuration. The optimized combination of dropout rate, attention regularization weight, number of LSTM units and number of CNN filters leads to a better balance between model capacity and generalization, which translates into lower out-of-sample errors.

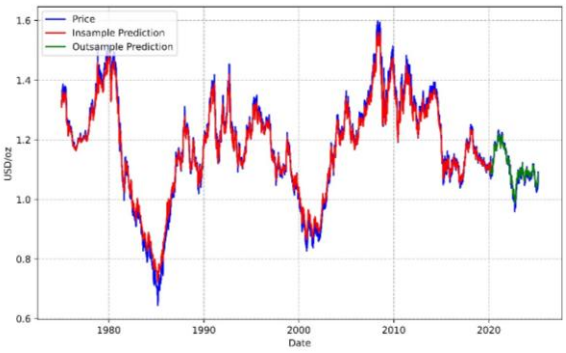

Fig. 1. MO-RSAR predictions for the EUR/USD exchange rate (1975-2025).

Figure 1 illustrates the predicted and true EUR/USD exchange rate for the full historical window (1975-2025) obtained using MO-RSAR. The predicted trajectory closely follows the actual price dynamics over both the training and test segments, capturing long-term trends as well as major regime changes. No systematic bias or lag can be observed in the out-of-sample period, which is consistent with the low MAPE reported in Table III.

An inspection of the training and validation loss curves (not shown for brevity) indicates stable convergence, with no evident signs of severe overfitting or underfitting. This suggests that the multi-objective hyperparameter optimization successfully balances model capacity and generalization for long and structurally complex exchange rate series.

2) S&P 500 (1979-2025)

The performance comparison for the full S&P 500 index series is summarized in Table IV. In this case, the differences between architectures become much more pronounced. The LSTM and CNN baselines exhibit very high errors (MAPE in the range of 15-18%), accompanied by large RMSE and RMAE. This reflects substantial absolute deviations between predicted and true index levels. A key factor is the position of the train-test split, which falls immediately after the COVID-19 crash in March 2020. The largest single-day drop of the index (around -9.5%) appears only in the last two overlapping training windows and always as the extreme point at the end of the sequence. The models do not see any data from the subsequent recovery in the training phase. As a result, the LSTM and CNN baselines effectively learn to treat this event as noise and adapt too slowly in the early part of the test period, which leads to large errors.

TABLE IV
RESULTS FOR SP500_ALL DATASET (1979-2025)

Model	MAPE	RMAE	RMSE
LSTM	1.52e+01	2.70e+01	8.88e+02
RSAR	5.29e+00	1.62e+01	3.81e+02
MO-RSAR	2.92e+00	1.19e+01	2.09e+02
CNN	1.83e+01	2.84e+01	8.32e+02

Both RSAR variants behave much more robustly in this scenario. The classical RSAR model achieves MAPE of about 5.3%, already a major improvement over LSTM and CNN. MO-RSAR further reduces the error to roughly 2.9%, which is the best result among all models and corresponds to a relative MAPE reduction of about 45% compared to the baseline RSAR configuration. This large gain confirms that multi-objective tuning of dropout, attention regularization and layer sizes can substantially improve performance on long, volatile index series.

Figure 2 presents the predicted and true S&P 500 index levels for the full historical window (1979-2025) obtained using MO-RSAR. The model adapts relatively quickly after the COVID-19 market crash in March 2020 and follows the subsequent recovery without a pronounced lag. Compared to the baseline architectures, MO-RSAR exhibits substantially lower bias in the out-of-sample period, which is consistent with the low MAPE reported in Table IV.

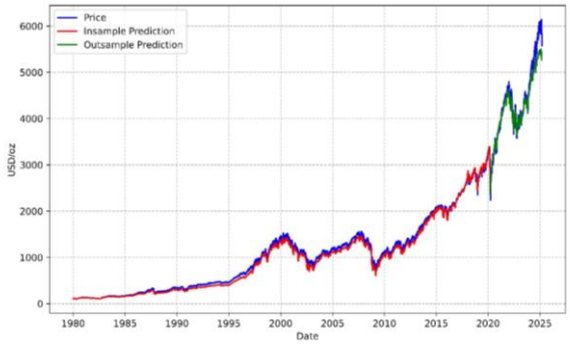

Fig. 2. MO-RSAR predictions for the S&P 500 index (1979-2025).

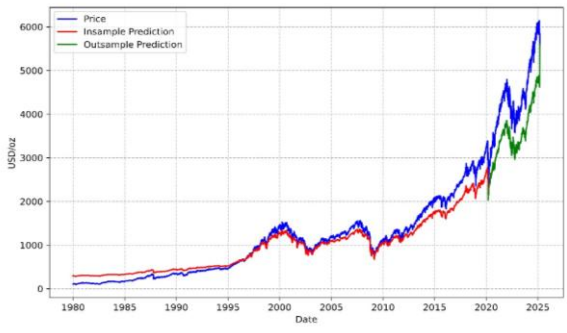

Fig. 3. CNN predictions for the S&P 500 index (1979-2025).

Interestingly, as illustrated in Figure 3, even though the CNN model performs poorly in terms of error metrics, its predicted price path follows the overall shape of the index reasonably well, but with a noticeable downward bias. In other words, the network tends to systematically underpredict the level of the index while correctly capturing the direction of movements. This explains the high MAPE values despite visually plausible trajectories.

3) *Bitcoin (long window)*

TABLE V
RESULTS FOR BTC_LONG DATASET (15.04.2024-15.04.2025)

Model	MAPE	RMAE	RMSE
LSTM	3.20e+00	5.11e+01	3.41e+03
RSAR	2.82e+00	4.83e+01	2.92e+03
MO-RSAR	2.87e+00	4.88e+01	2.93e+03
CNN	2.89e+00	4.88e+01	3.156e+03

The results for the one-year Bitcoin dataset are summarized in Table V. For this dataset, all architectures achieve MAPE in the range of 2.8–3.2%. The LSTM model reaches the highest error (around 3.2%), which is still acceptable given the extreme volatility and limited predictability of cryptocurrency prices. The other three models are very close to each other: the classical RSAR achieves the lowest MAPE (≈ 2.8%), the MO-RSAR is slightly worse (≈ 2.9%), and CNN is almost indistinguishable from MO-RSAR, as shown in Table V.

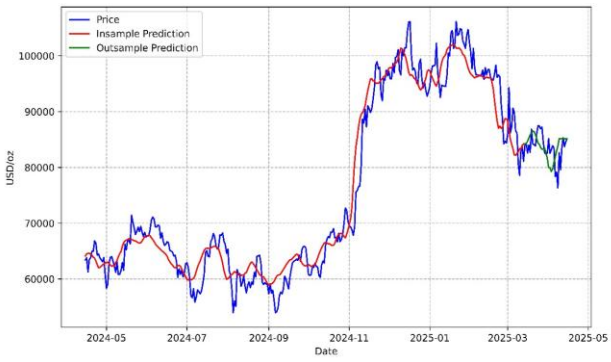

Fig. 4. MO-RSAR predictions for the Bitcoin price (long window).

Figure 4 illustrates the predicted and true Bitcoin prices over the one-year test window obtained using MO-RSAR. The predicted trajectory follows general direction and major swings of the price series, while smoothing short-term fluctuations. This behaviour is consistent with the relatively small differences in MAPE between all compared architectures in Table V.

The differences between RSAR and MO-RSAR on this dataset are therefore marginal, and the multi-objective optimization does not yield a visible advantage. This may indicate that, over such a short horizon with highly noisy dynamics, all models face a similar irreducible error and additional architectural sophistication cannot reduce it much further. In this regime, the hybrid RSAR architecture behaves comparably to simpler approaches.

C. *Short recent windows*

1) *EUR/USD (2005-2025)*

The quantitative comparison for the shorter EUR/USD sample is reported in Table VI.

TABLE VI
RESULTS FOR EUR/USD_RECENT DATASET (2005-2025)

Model	MAPE	RMAE	RMSE
LSTM	8.19e-01	9.50e-02	1.16e-02
RSAR	1.34e+00	1.20e-01	1.82e-02
MO-RSAR	1.29e+00	1.17e-01	1.73e-02
CNN	2.16e+00	1.52e-01	2.77e-02

For this shorter, twenty-year EUR/USD sample, the LSTM model achieves the best performance, with MAPE remaining at approximately 0.82%, similar to the value obtained on the longer series. In contrast, both RSAR variants and the CNN baseline deteriorate compared to the full-history experiment. The classical RSAR reaches MAPE of about 1.33%, and MO-RSAR improves it only slightly to around 1.29%, which corresponds to a modest 4% reduction in MAPE (Table VI).

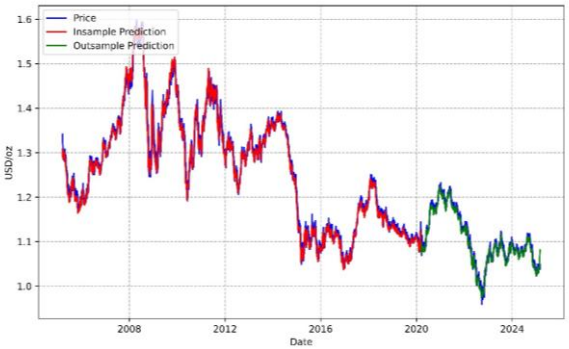

Fig. 5. LSTM predictions for the EUR/USD exchange rate (2005-2025).

Figure 5 illustrates the predicted and true EUR/USD exchange rate for the shorter, recent window (2005-2025) obtained using the LSTM model. The predicted trajectory closely follows the actual price dynamics in the out-of-sample period, which is consistent with the lowest MAPE reported in Table VI.

This behaviour suggests that restricting the training data to a more recent but shorter period may remove some historical patterns that were useful for the hybrid RSAR architecture, while the LSTM is able to capture the dominant structure of the exchange rate from a smaller sample. Eliminating very old observations may also help the LSTM focus on more relevant, recent dynamics, which stabilizes or even slightly improves its accuracy.

2) *S&P 500 (2005-2025)*

The results for the reduced S&P 500 sample are summarized in Table VII. On the reduced S&P 500 sample (2005-2025), the LSTM error remains high (MAPE ≈ 14.6%), which indicates persistent difficulty in modelling this index, especially around large crashes and sharp trend reversals. The classical RSAR performance also degrades compared to the full series, with MAPE increasing from approximately 5.3% to about 7.8%. The likely reason is the loss of long-term information that had helped the model learn the structure of the index over multiple cycles.

TABLE VII
RESULTS FOR SP500_RECENT DATASET (2005-2025)

Model	MAPE	RMAE	RMSE
LSTM	1.46e+01	2.66e+01	9.08e+02
RSAR	7.77e+00	1.96e+01	5.67e+02
MO-RSAR	6.08e+00	1.71e+01	4.37e+02
CNN	7.41e+00	1.83e+01	3.64e+02

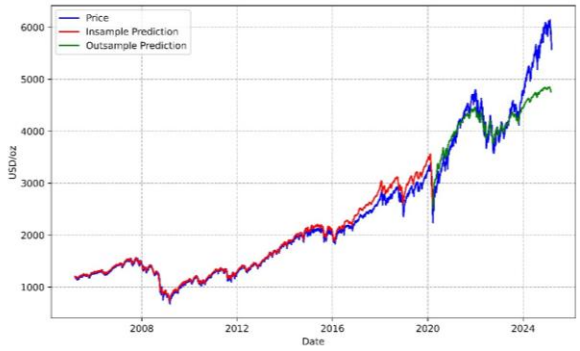

Fig. 6. CNN predictions for the S&P 500 index (2005-2025).

As illustrated in Figure 6, the CNN model performs substantially better on the reduced S&P 500 sample than on the full historical series. Its MAPE decreases from approximately 18.3% to about 7.4%, as reported in Table VII. Removing the earliest decades reduces the variability range of the index and yields a more homogeneous dataset, which facilitates the extraction of local temporal patterns by convolutional filters. Nevertheless, the CNN predictions still exhibit a noticeable downward bias, indicating systematic underestimation of index levels despite correctly capturing the direction of market movements.

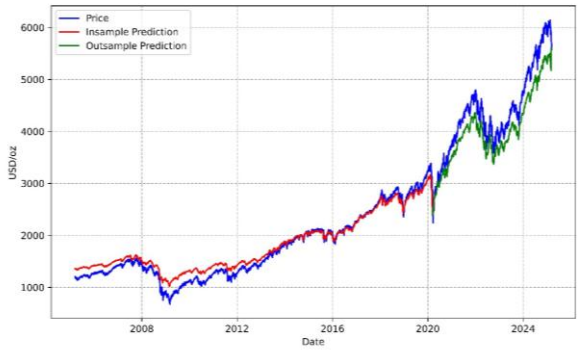

Fig. 7. MO-RSAR predictions for the S&P 500 index (2005-2025).

Figure 7 presents the predictions obtained using MO-RSAR for the reduced S&P 500 window. Although the forecasting error increases compared to the full historical series, the MO-RSAR remains the most accurate architecture, achieving MAPE of approximately 6.1% (Table VII). The hyperparameters selected for this dataset differ from those for the longer S&P 500 sample, which indicates that the optimization procedure successfully adapts the architecture to the changed statistical properties of the data.

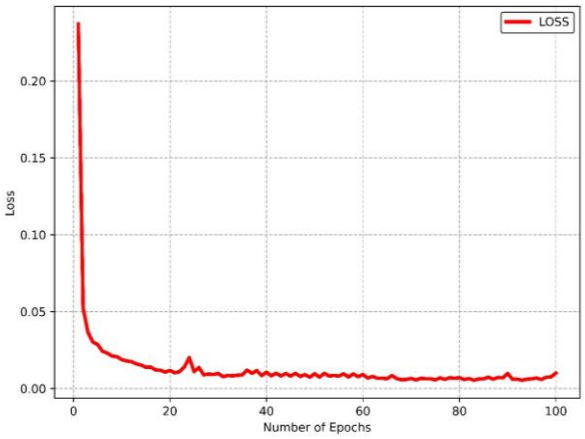

Fig. 8. Training loss of the MO-RSAR model for the S&P 500 index (2005-2025).

Figure 8 shows the training loss curve of MO-RSAR. The loss decreases rapidly during the initial epochs and then stabilizes without exhibiting an increasing trend, suggesting stable training and no evident signs of severe overfitting or underfitting. The remaining prediction errors are therefore likely related primarily to data complexity.

To provide a compact quantitative summary of the observed training dynamics for the analyzed 100-epoch training run, the loss trajectory in Figure 8 was additionally approximated with the exponential model introduced in Section IV.E: $\hat{y}(t;\theta) = c + ae^{-bt}$. The fitted parameters $\theta^* = (a^*, b^*, c^*)$ were obtained by least squares (Excel Solver) and yielded a low sum of squared errors (SSE). This indicates that the convergence pattern is well captured by a fast initial decay followed by a near-constant plateau. The parameter interpretation follows Section IV.E (with c^* as the asymptotic loss floor, a^* as the initial amplitude and b^* the convergence rate). For this run, $\theta^* = (0.5696, 1.1728, 0.0109)$ and $SSE = 0.0080$. The power-law alternative resulted in a higher residual error ($SSE = 0.0454$), therefore the exponential specification was retained.

3) Bitcoin (recent window)

TABLE VIII
RESULTS FOR BTC_RECENT DATASET (15.12.2024-15.04.2025)

Model	MAPE	RMAE	RMSE
LSTM	2.37e+00	4.42e+01	2.68e+03
RSAR	5.76e+00	6.88e+01	5.56e+03
MO-RSAR	5.57e+00	6.77e+01	5.42e+03
CNN	3.19e+00	5.12e+01	3.39e+03

The results for the shortest Bitcoin dataset are reported in Table VIII. For the four-month window, the ranking of models changes noticeably. The LSTM baseline achieves the lowest error, with MAPE reduced to approximately 2.4%, improving on its performance on the long Bitcoin window. The CNN model exhibits a slight deterioration (MAPE ≈ 3.2%), accompanied by an increase in RMSE. In contrast, both RSAR variants perform substantially worse:

the classical RSAR reaches MAPE of about 5.8%, and MO-RSAR about 5.6% (Table VIII).

Figure 9 illustrates the predicted and true Bitcoin prices for the four-month window obtained using the LSTM model. The predicted trajectory captures the general direction of price movements, although short-term turning points are occasionally followed with a slight delay, which is consistent with the relatively low MAPE reported in Table VIII.

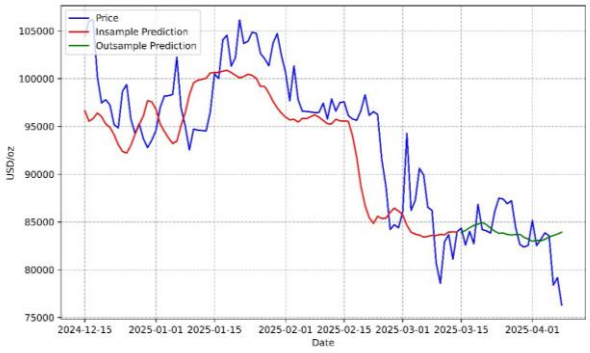

Fig. 9. LSTM predictions for the Bitcoin price (recent window).

Figure 10 shows the training loss curve of the LSTM model. The loss continues to decrease throughout the training process without reaching a clear plateau, indicating mild underfitting under the fixed training schedule.

Taken together, these observations suggest that, for very short and highly volatile time series, hybrid architectures with a large number of trainable parameters may not fully exploit their capacity due to limited data availability, whereas simpler recurrent models such as LSTM provide a more favourable balance between model capacity and data support.

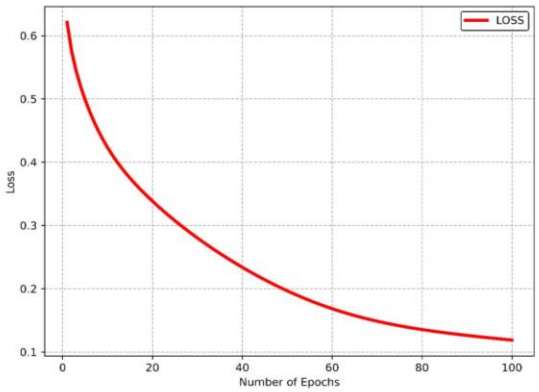

Fig. 10. Training loss of the LSTM model for the Bitcoin price (recent window).

To provide a compact quantitative summary of the observed training dynamics for the analysed 100-epoch training run, the loss trajectory in Figure 10 was additionally approximated with the exponential model introduced in Section IV.E: $\hat{y}(t;\theta) = c + ae^{-bt}$. The fitted parameters $\theta^* = (a^*, b^*, c^*)$ were obtained by least squares (Excel Solver) and yielded a low sum of squared errors (SSE). This indicates that the convergence pattern is well captured by a fast initial decay followed by a near-constant plateau. The parameter interpretation follows Section IV.E (with c^* as the asymptotic loss floor, a^* as the initial amplitude and b^* the convergence rate). For this run, $\theta^* = (0.4812, 0.0396, 0.1199)$ and $SSE = 0.0126$. The power-law alternative resulted in a higher residual error ($SSE = 0.1186$), therefore the exponential specification was retained.

VI. Discussion and General Observations

From a global perspective, MO-RSAR emerges as the most accurate and consistent architecture. Across all six datasets, it achieves the lowest average MAPE (3.24%), compared with 3.99% for the classical RSAR, 5.90% for the CNN baseline and 6.18% for LSTM. In three out of six experiments it delivers the best result, while the baseline RSAR wins once and LSTM is best in two cases, both corresponding to shorter recent windows. The relative improvement of MO-RSAR over the original RSAR ranges from a slight deterioration of about 2% on the shortest Bitcoin sample to a gain of roughly 45% on the full S&P 500 series, with an average reduction in MAPE of approximately 16%. These figures confirm that multi-objective hyperparameter optimization can significantly enhance the hybrid RSAR architecture, particularly on long and structurally complex financial time series.

At the same time, the results show that this additional complexity does not uniformly translate into improvements across all scenarios and horizons. The RSAR and MO-RSAR models are relatively heavy architectures with a large number of trainable parameters. This complexity is clearly beneficial on long and information-rich series, such as the full S&P 500 history or the full EUR/USD sample, where MO-RSAR delivers substantial reductions in MAPE compared to both the classical RSAR and the simpler baselines. However, on shorter and more limited datasets – especially the four-month Bitcoin window – both RSAR variants perform noticeably worse than LSTM and even CNN, despite using the same training protocol.

The comparison across market types also highlights distinct modelling challenges. EUR/USD appears to be relatively predictable under the proposed setup, with MAPE below 1% for both horizons. The S&P 500 series is the most challenging, largely due to the extreme shock around the COVID-19 crash near the train-test boundary, which substantially affects baseline models. Bitcoin exhibits moderate error levels (≈2-3% MAPE) on the long window, but becomes substantially harder on the four-month window, where short-term noise dominates and the data window is very limited.

Several practical implications follow. First, MO-RSAR is most beneficial when the time series is long enough to support training a higher-capacity hybrid model and when robustness to regime changes is important. Second, for short horizons or limited datasets, simpler architectures (LSTM, and in some cases CNN) can be competitive or even preferable. Third, the computational overhead of NSGA-II should be justified by the expected accuracy gain. For scenarios where RSAR and MO-RSAR perform similarly (e.g., the one-year Bitcoin series), the optimized approach may not be cost-effective.

One methodological limitation should also be noted. In the present study, standardization was performed using the mean and standard deviation computed on the full sample rather than

on the training segment only. Although this preprocessing choice was applied consistently across all compared architectures, it may introduce a mild look-ahead effect and should therefore be interpreted as a limitation of the current experimental design.

These limitations point to several directions for future work. One natural extension would be to design a lighter variant of RSAR, retaining the overall LSTM-attention-CNN structure but reducing its depth or width, for example by decreasing the number of filters and units or by simplifying the attention component. Another promising avenue is to adapt the NSGA-II search space to the characteristics of the dataset, for instance by narrowing or shifting the ranges of hyperparameters or by introducing additional decision variables. In the present study, the same generic hyperparameter ranges were used across all markets and horizons. Allowing the optimization procedure to operate on problem-specific ranges might improve performance on short windows without sacrificing accuracy on longer ones. Finally, future research could consider extending the multi-objective framework to include training-related parameters (such as the number of epochs or early stopping criteria), so that the trade-off between model complexity, convergence speed and generalization is controlled in a more explicit and flexible way.

CONCLUSION

This paper investigated the effectiveness of several deep learning architectures for financial time-series forecasting, with particular emphasis on attention mechanisms and multi-objective hyperparameter optimization. Four models were compared under a unified experimental setup: LSTM, CNN, the classical RSAR architecture, and MO-RSAR, a multi-objective extension of RSAR optimized using NSGA-II.

The main contribution of this work is the introduction of MO-RSAR, a multi-objective extension of the attention-based RSAR architecture for financial time-series forecasting. In this framework, RSAR configuration is formulated as a bi-objective search that jointly minimizes out-of-sample forecasting error and the train-validation gap, rather than relying on manual or single-metric tuning. To the best of the author's knowledge, this specific RSAR-NSGA-II coupling in a multi-objective setting has not been previously reported for RSAR-based financial forecasting under a unified benchmark across heterogeneous markets.

The evaluation was conducted on six real-world datasets covering Forex, equity index, and cryptocurrency markets, and both long and recent data windows, enabling a comprehensive comparison across diverse market conditions. The results demonstrate that MO-RSAR provides the best overall performance. On average, it achieves the lowest prediction error across all datasets and improves upon the classical RSAR architecture. The benefits of multi-objective optimization are especially pronounced for long and structurally complex time series, such as the full S&P 500 and EUR/USD histories, where robustness to regime changes plays a critical role.

At the same time, the experiments show that increased model complexity does not guarantee superior performance in all settings. For short and highly volatile time series, particularly the four-month Bitcoin window, simpler architectures such as LSTM achieve better or comparable accuracy. In these cases, higher-capacity hybrid attention-based models appear to be constrained by limited data and do not fully exploit their capacity under a fixed training budget.

The comparative analysis across asset classes highlights distinct forecasting challenges. Exchange rates exhibit relatively stable dynamics and remain highly predictable under the proposed setup. Equity index series are strongly affected by rare but extreme events, which can significantly challenge simpler models. Cryptocurrency prices combine high volatility with limited historical depth, making short-horizon forecasting particularly difficult and sensitive to model capacity.

Overall, the findings indicate that the choice of forecasting architecture should depend on the length of the available time series, the level of market volatility, and practical computational constraints. While MO-RSAR is well suited for long-term and structurally complex forecasting tasks where accuracy is the primary objective, simpler models such as LSTM or CNN may be more appropriate for short horizons or limited datasets. The proposed MO-RSAR framework supports model selection in financial forecasting applications and highlights the practical value of multi-objective optimization for improving predictive performance.

REFERENCES

[1] J. Zhou, Z. He, Y. Song, H. Wang, X. Yang, W. Lian, "Precious metal price prediction based on deep regularization Self-Attention Regression", IEEE, vol. 8, pp. 2178-2187, Jan. 2020. doi: 10.1109/ACCESS.2019.2962202

[2] S. Hochreiter, Untersuchungen zu dynamischen neuronalen Netzen, Diploma thesis, Institut für Informatik, Technische Universität München, 1991, [Online] Available: https://people.idsia.ch/~juergen/SeppHochreiter1991ThesisAdvisorSchmidhuber.pdf, Accessed: Feb. 18, 2026.

[3] Y. Bengio, P. Simard, P. Frasconi, "Learning long-term dependencies with gradient descent is difficult.", IEEE transactions on neural networks, vol. 5 (2), pp. 157-166, Mar. 1994. doi: 10.1109/72.279181

[4] S. Hochreiter, J. Schmidhuber, "Long short-term memory", Neural Computation, vol. 9 (8), pp. 1735-1780, Nov. 1997. doi: 10.1162/neco.1997.9.8.1735

[5] T. Fisher, C. Krauss, "Deep Learning with Long Short-Term Memory Networks for Financial Market Predictions.", European Journal of Operational Research, vol. 270 (2), pp. 645-669, Dec. 2017, doi: 10.1016/j.ejor.2017.11.054

[6] B. Gülmez, "Stock price prediction with optimized deep LSTM network with artificial rabbits optimization algorithm", Expert Systems with Applications, vol. 227, Article no. 120346, 2023. doi: 10.1016/j.eswa.2023.120346

[7] J. Li, "Comparison of Regressions and Multi-feature LSTM in Stock Price Prediction for Top Car Companies", Advances in Economics, Management and Political Sciences, vol. 85, pp. 170-180, May 2024. doi: 10.54254/2754-1169/85/20240869

[8] Y. R. Madhika, K. Kusrini, T. Hidayat, "Gold Price Prediction Using ARIMA and LSTM Models", Sinkron, vol. 7 (3), pp. 1255-1264, Jul. 2023. doi: 10.33395/sinkron.v8i3.12461

[9] K. Fukushima, "Neocognitron: A Self-organizing Neural Network Model for a Mechanism of Pattern Recognition Unaffected by Shift in Position", Biological Cybernetics, vol. 36, pp. 193-202, Apr. 1980. doi: 10.1007/BF00344251

[10] Y. LeCun, L. Bottou, Y. Bengio, P. Haffner, "Gradient-Based Learning Applied to Document Recognition", IEEE, vol. 86 (11), pp. 2278-2324, Dec. 1998. doi: 10.1109/5.726791

[11] Z. Li, F. Liu, W. Yang, S. Peng, J. Zhou, "A Survey of Convolutional Neural Networks: Analysis, Applications, and Prospects", IEEE, vol. 33 (12), pp. 6999-7019, Apr. 2020. doi: 10.1109/TNNLS.2021.3084827

[12] R. Velastegui, L. Zhinin-Vera, G. E. Piliza, O. Chang, "Time series prediction by using Convolutional Neural Networks", Proceedings of the

Future Technologies Conference (FTC), Vancouver, pp. 499-51, 2020. doi: 10.1007/978-3-030-63128-4_38

[13] W. Lu, J. Li, Y. Li, A. Sun, and J. Wang, "A CNN-LSTM-Based Model to Forecast Stock Prices", Complexity, vol. 2020, pp. 1-10, Nov. 2020. doi: 10.1155/2020/6622927

[14] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, I. Polosukhin, "Attention is all you need", Neural Information Processing Systems, vol. 30, pp. 5998-6008, Jun. 2017. doi: 10.48550/arXiv.1706.03762

[15] Y. Li, X. Zhang, X. Zhu, "Application of LSTM and Attention Mechanism for Stock Price Prediction and Analysis" in Proc. 2023 2nd Int. Conf. on Artificial Intelligence, Internet and Digital Economy (ICAID 2023), Chengdu, China, pp. 553-56, Aug. 2023 doi: 10.2991/978-94-6463-222-4_60

[16] Q. Zhang, C. Qin, Y. Zhang, F. Bao, C. Zhang, and P. Liu, "Transformer-based attention network for stock movement prediction," Expert Systems With Applications, vol. 202, Art. no. 117239, 2022. doi: 10.1016/j.eswa.2022.117239

[17] R. Cont, "Empirical properties of asset returns: stylized facts and statistical issues", Quantitative Finance, vol. 1 (2), pp. 223-236, Mar. 2001. doi: 10.2469/faj.v68.n2.6

[18] A. W. Lo, "The adaptive markets hypothesis: Market efficiency from an evolutionary perspective", The Journal of Portfolio Management, vol. 30 (5), pp. 15-29, 2004. doi: 10.3905/jpm.2004.442611

[19] S. Lahmiri, "A comparison of ARIMA and ANN models for financial time series prediction", Management Science Letters, vol. 2, pp. 2551-2556, 2011. doi: 10.5267/j.msl.2012.07.009

[20] M. Czyżewska, "Multi-Objective Optimization of Regularized Self-Attention Regression Models Using NSGA-II: A Methodological Framework," accepted for presentation at the 46th IBIMA Computer Science Conference, Ronda, Spain, Nov. 26–27, 2025, and for inclusion in the conference proceedings. [Online]. Available: https://ibima.org/accepted-paper/multi-objective-optimization-of-regularized-self-attention-regression-models-using-nsga-ii-a-methodological-framework/

[21] Investing.com, "Historical data." [Online]. Available: https://www.investing.com/. Accessed: Mar. 12, 2025.