EAI Endorsed Transactions
on Internet of Things
Research Article EAI.EU

A Novel Gated Fusion CNN-LSTM Model for Multi-Horizon Intraday Gold Price Forecasting

Nguyen Hoang Ha[1], Cuong H. Nguyen-Dinh[2], and Truong An Binh[1*]

[1] Hue University of Sciences, Hue, Vietnam
[2] University of Finance – Marketing, Ho Chi Minh City, Vietnam

Abstract

Gold serves as a key inflation hedge and portfolio stabilizer, making accurate price forecasting essential for investors. Hourly gold prices exhibit pronounced non-linearity and microstructure noise that limit traditional econometric models, motivating a shift toward deep learning. Existing CNN-LSTM architectures cascade convolutional and recurrent layers sequentially, without a mechanism to reconcile their complementary representations. We propose a Dual-Branch CNN-LSTM architecture with Gated Fusion, combining a convolutional-recurrent deep branch with a parallel raw-input skip branch, adaptively merged by a learned gate inspired by the Gated Multimodal Unit. Input sequences are restructured via sliding windows across three forecast horizons (24→1, 48→2, and 72→3 hours). The model is validated on 37,140 hourly XAUUSDm observations (Exness, 2020-2026) against 1D-CNN, LSTM, and Sequential CNN-LSTM baselines, achieving the best or tied-best accuracy across all configurations. For the 24-hour horizon, MAE = 9.00 USD, R^2 = 0.9994, and DA = 52.6%, on the original USD scale. Diebold-Mariano tests confirm significant gains over the 1D-CNN and Sequential CNN-LSTM baselines ($p < 0.001$), while McNemar tests confirm directional-accuracy gains in 8 of 9 comparisons ($p < 0.05$). These results show a modest but robust improvement over prior CNN-LSTM designs, offering a reliable foundation for risk management pending economic backtesting.

Keywords: gold price forecasting, CNN-LSTM, gated fusion, deep learning, time series forecasting, directional accuracy

Received on 28 April 2026, accepted on 02 August 2026, published on 13 August 2026

doi: 10.4108/eetiot.12806

*Corresponding author. Email: truonganbinh@husc.edu.vn

1. Introduction

Financial time series forecasting has become one of the most active application areas of applied machine learning over the past decade, driven by the sheer scale of assets under management whose returns depend on accurately anticipating short-term price movements; recent syntheses of this literature, such as the reviews by Zhang et al. [1] and Li and Law [2], document a rapidly accumulating body of work spanning equities, foreign exchange, and commodities. Within this broader field, gold occupies a distinctive position: it functions as a central instrument within global financial markets, serving as a hedge against inflation and a stabilizer during periods of economic uncertainty, and investors and central banks utilize the metal to mitigate portfolio risk, causing its valuation to fluctuate in response to macroeconomic variables, geopolitical events, and currency exchange rates. The precise forecasting of gold prices therefore constitutes a critical task for risk management strategies and algorithmic trading systems, particularly within high-frequency trading environments where price movements occur at millisecond-to-hourly intervals and where even a modest, consistent edge in predictive accuracy can translate into materially better hedging and trading outcomes.

Forecasting approaches for gold and related commodities have evolved through several overlapping generations, each achieving clear gains over its predecessor while leaving specific weaknesses unresolved. Classical econometric

models, most notably the Autoregressive Integrated Moving Average (ARIMA) family for the conditional mean and the Generalized Autoregressive Conditional Heteroskedasticity (GARCH) family for conditional variance, were the first tools applied to this problem and remain valued for their interpretability and low data requirements; however, as Li and Law [2] note, both families formally assume linear dependence and, in ARIMA's case, constant variance, assumptions that break down under the volatility clustering and heavy-tailed shocks characteristic of high-frequency financial data. Machine learning (ML) and deep learning methods emerged specifically to relax these assumptions: Foroutan and Lahmiri [3] showed that recurrent architectures, particularly Gated Recurrent Units (GRUs), outperform standard regression techniques across crude oil, gold, and silver, and the broader capacity of Long Short-Term Memory (LSTM) networks to retain information over extended sequences has made them a default choice for financial time series analysis. Yet standalone Recurrent Neural Networks (RNNs) still struggle to extract local, short-horizon features from raw high-frequency data, where microstructure noise and short-term volatility can obscure the very trends the recurrent component is meant to track. To close this specific gap, prior work [4], [5] proposed hybrid Convolutional Neural Network (CNN)-LSTM frameworks in which convolutional layers extract and denoise local features before an LSTM layer models their temporal evolution, and demonstrated that this synergy mitigates the individual weaknesses of standalone CNN or LSTM models. These hybrid architectures, however, share a structural limitation that has received comparatively little attention: convolutional and recurrent components are combined through a single, fixed, sequential pipeline, in which the CNN's output is the LSTM's only input, and the model has no explicit mechanism for judging, on a sample-by-sample basis, whether its own learned features are actually more informative than the raw input signal they were extracted from.

This unexamined assumption, that a sequential CNN-then-LSTM pipeline's learned representation is always preferable to the raw input it was derived from, is the central motivation for the present study: if the assumption does not hold uniformly across samples and market conditions, a fixed pipeline can only ever be as good as its weakest case, whereas a model that could adaptively choose, per sample, how much to rely on learned versus raw features should be able to do at least as well and potentially better. Motivated by this observation, we propose a Dual-Branch CNN-LSTM architecture with Gated Fusion, in which a convolutional-recurrent deep branch and a raw-input skip branch are trained in parallel rather than sequentially, and are combined through a learned, per-sample gate that makes the reliance on convolutional-recurrent features an explicit, data-dependent quantity instead of a fixed architectural choice. The objectives of this study are threefold: first, to determine whether this gated dual-branch design measurably outperforms standalone 1D-CNN, standalone LSTM, and a literal reimplementation of the Sequential CNN-LSTM Baseline [4] across multiple forecast horizons; second, to establish this comparison on a dataset and statistical testing protocol rigorous enough to support the resulting claims, namely several years of hourly, multi-regime gold price data evaluated with formal significance tests rather than point-estimate comparisons alone; and third, to report these results honestly, distinguishing genuine directional predictive signal from the scale-driven, near-saturated R^2 values that high-frequency price-level forecasting tends to produce regardless of model quality.

This study makes four primary contributions to the domain of financial forecasting. First, we propose a Dual-Branch CNN-LSTM architecture with Gated Fusion, in which a convolutional-recurrent deep branch and a raw-input skip branch are trained in parallel and adaptively combined through a learned, per-sample gate, moving beyond the sequential single-branch designs that dominate prior CNN-LSTM literature. Secondly, we benchmark the proposed architecture against three strong baselines, including a literal reimplementation of the Sequential CNN-LSTM Baseline drawn from prior literature, on 37,140 hourly XAUUSDm observations spanning 6.3 years (January 2020-April 2026) across three forecast configurations, addressing the limited dataset scope and weak baselines of earlier single-month studies. Thirdly, we establish a rigorous statistical evaluation protocol that reports all error metrics on the original USD price scale and applies Diebold-Mariano and McNemar tests to confirm, respectively, the statistical significance of the proposed model's magnitude-error and directional-accuracy advantages over each baseline. Finally, we explicitly disentangle genuine predictive signal from magnitude-fit artifacts by contrasting the near-saturated R^2 values common to all models with the more diagnostic Directional Accuracy metric, presenting the results as evidence of architectural and statistical improvement rather than an overstated or validated trading signal.

The remainder of this paper is organized as follows. Section 2 synthesizes the existing literature regarding financial time series forecasting, with specific emphasis on deep learning applications and the development of hybrid architectures. Section 3 details the proposed methodology, delineating the data preprocessing pipeline and the mathematical formulation of the proposed Dual-Branch CNN-LSTM with Gated Fusion architecture. Section 4 presents the experimental setup, describes the dataset characteristics, and analyzes the empirical results obtained from the comparative evaluation against benchmark models. Finally, Section 5 summarizes the findings and outlines potential directions for future research.

2. Related Works

The rapid accumulation of research on financial time series forecasting has necessitated periodic synthesis of the field, and several survey studies chart the historical trajectory from classical econometrics toward increasingly data-driven paradigms. Zhang et al. [1] reviewed deep learning models for financial price forecasting published between 2020 and 2022, documenting an accelerating shift away from purely statistical techniques as practitioners sought architectures

capable of capturing the non-linear, high-frequency dynamics that characterize commodities such as gold, crude oil, and foreign exchange. Li and Law [2] extended this synthesis by systematically cataloguing deep learning methodologies for temporal-event modeling, explicitly outlining the structural limitations of classical statistical models in high-accuracy forecasting scenarios and organizing the field's fundamental architectural families. Kong et al. [6] further consolidated this fragmented literature by proposing general paradigms for deep time series forecasting, focusing on how the underlying decomposition and representation of a time series determines the choice of feature-extraction strategy across domains as varied as economics and energy. Taken together, these surveys frame gold and related commodity price forecasting as a research object whose methodological history has progressed through four broad, partially overlapping waves: traditional statistical models, standalone machine learning and deep learning architectures, spatio-temporal and decomposition-based hybrids, and, most recently, transformer- and graph-based architectures. The remainder of this section reviews each wave in turn, situates the proposed Dual-Branch CNN-LSTM with Gated Fusion architecture within it, and identifies the specific gap this study addresses.

Long before the deep learning era, gold and commodity price forecasting relied primarily on linear and conditional-variance econometric models, most notably the ARIMA family for the conditional mean and the GARCH family for conditional variance, precisely because both offer closed-form estimation, interpretable parameters, and modest data requirements. As Li and Law [2] document in their systematic review, these models formally assume linear dependence, stationarity, and, in the case of ARIMA, constant variance, assumptions that hold reasonably well for low-frequency, long-horizon macroeconomic series but break down under the volatility clustering and heavy-tailed shocks characteristic of high-frequency commodity data. Illustrating the field's response to this specific limitation, Nallamothu et al. [7] extended the GARCH family itself by incorporating skewness and kurtosis terms (SKGARCH) to more explicitly model the non-normal, heavy-tailed volatility of gold prices, and then paired this enhanced statistical model with an LSTM network to jointly capture higher-moment volatility structure and non-linear temporal dependence. This same classical-versus-learned tension recurs outside precious metals: within EAI Endorsed Transactions on Internet of Things itself, Limbare and Agarwal [28] compared ARIMA, Holt-Winters, and LSTM models for automotive supply-chain demand forecasting and likewise found all three approaches viable without resolving which architecture best captures cross-domain time-series structure, a gap this study addresses directly for the CNN-LSTM fusion mechanism in the gold-price setting. While such statistical-deep-learning hybrids meaningfully improve on pure ARIMA/GARCH baselines, they still inherit the core constraint that the variance or mean equation must be specified a priori in a fixed parametric form, which limits their capacity to adapt to the regime shifts and microstructure noise present in hourly, multi-year gold price data. The present study departs from this paradigm entirely: rather than specifying any parametric mean or variance equation, the proposed Dual-Branch CNN-LSTM with Gated Fusion learns both local (convolutional) and long-range (recurrent) structure directly from raw log-return sequences, removing the linearity and fixed-form variance assumptions that constrain even the hybridized statistical models discussed above.

Beyond parametric statistical models, a broad body of work has applied general-purpose machine learning and deep learning algorithms directly to gold and related commodity price data, typically as standalone architectures rather than as decomposition- or spatial-feature hybrids. On the classical machine learning side, Cohen and Aiche [8] applied Random Forest and Gradient Boosted Regression Trees to identify that international stock indices and the VIX volatility index serve as useful lagged predictors of gold prices, while Jabeur et al. [9] combined XGBoost with Shapley additive explanations (SHAP) to both improve accuracy over alternative machine learning methods and interpret which macro-financial factors drive gold price movements. Abu-Doush et al. [10] and Weng et al. [11] instead focused on optimizing the training of simpler neural architectures, respectively an archive-based Harris Hawks Optimizer for a Multilayer Perceptron and a genetic-algorithm regularization scheme for an online sequential Extreme Learning Machine, to escape local optima and handle the singular-matrix instabilities that arise when training on volatile commodity data. On the deep learning side, recurrent architectures have been refined along several complementary axes: Memon et al. [12] and Wang et al. [13] respectively augmented GRU networks with multi-headed attention plus skip connections and with stacked-layer attention to better weight informative historical time steps; Shahi et al. [14] and Chi et al. [15] incorporated exogenous signals, financial news sentiment and engineered technical indicators respectively, into LSTM/GRU pipelines; and Alzakari et al. [16] combined economic variables with an LSTM-RNN model normalized via Z-scores to outperform support vector regression. At a larger comparative scale, Foroutan and Lahmiri [3] benchmarked sixteen architectures across crude oil, gold, and silver to find GRU consistently most accurate, Manogna et al. [17] similarly found LSTM and GRU superior to classical stochastic and machine learning baselines across twenty-three agricultural commodities, and Xu et al. [18] extended this comparative logic to crude oil by combining multivariate variable selection with a non-linear weighted ensemble of multiple models. Collectively, this body of work establishes that both classical ML and standalone deep learning architectures reliably outperform linear statistical baselines, and that attention mechanisms, exogenous features, and metaheuristic optimization each yield incremental accuracy gains. However, none of these studies extracts explicit local spatial patterns from the raw price sequence before recurrent modeling, a limitation that becomes acute for high-frequency data, where microstructure noise and short-term volatility can obscure the very trends that RNN-family models are meant to track. The present study addresses this specific gap by prepending a convolutional feature-extraction stage to the recurrent branch, rather than relying solely on attention, exogenous features, or optimizer refinements to compensate for it.

To address precisely this gap, the inability of standalone RNNs to extract local spatial patterns from raw high-frequency sequences, a distinct research stream has integrated convolutional layers with recurrent sequence models into unified spatio-temporal architectures. Livieris et al. [4] proposed one of the most widely referenced instances of this paradigm, stacking convolutional layers ahead of an LSTM layer so that the CNN component performs internal representation learning and noise filtering while the LSTM component models the resulting feature sequence's temporal dynamics; this study's Sequential CNN-LSTM Baseline is a literal reimplementation of that architecture. You et al. [5] extended the same CNN-LSTM paradigm by incorporating external uncertainty metrics, the economic policy uncertainty index and a volatility index, to quantify how macro-level uncertainty modulates gold price fluctuations. Both studies demonstrate that combining convolutional and recurrent components in a single pipeline meaningfully improves on standalone CNN or LSTM baselines by decoupling spatial feature extraction from temporal sequence modeling. However, both architectures process the two components strictly sequentially and in a single branch: the CNN's output is the LSTM's only input, with no direct path from the raw input to the prediction head, and the two representations are never explicitly weighed against one another. This design implicitly assumes that convolutional feature extraction is always beneficial and never discards information the raw input would have preserved, an assumption this study's own results call into question, since the Sequential CNN-LSTM Baseline is shown in Section 4 to underperform a plain LSTM on directional accuracy despite matching it closely on magnitude error. The proposed Dual-Branch CNN-LSTM with Gated Fusion directly targets this architectural constraint: by running the convolutional-recurrent pathway in parallel with a raw-input skip branch and combining the two through a learned, per-sample gate rather than a fixed sequential pipeline, the model can adaptively fall back on the unfiltered raw signal whenever the deep branch's learned representation is less informative, the concrete point of departure from Livieris et al. [4] that motivates this study's architecture.

A parallel research stream has pursued signal decomposition as a pre-processing step, on the premise that splitting a noisy, non-stationary price series into cleaner frequency sub-components before modeling should ease the burden on the downstream predictor. Early instantiations of this idea relied on classical, fixed-resolution decompositions: Zhang et al. [19], for instance, applied the Discrete Wavelet Transform, which decomposes a series into a pre-specified, fixed number of frequency sub-bands chosen in advance by the researcher, for noise reduction before feeding the denoised series into an optimized LSTM-P network, reporting higher precision for Bitcoin and gold price forecasting than a standard LSTM. Subsequent work replaced this fixed-resolution decomposition with adaptive, data-driven alternatives: Liang et al. [20] employed Improved Complete Ensemble Empirical Mode Decomposition with adaptive noise (ICEEMDAN), which determines its own number and shape of intrinsic mode functions from the data rather than a pre-set decomposition level, before modeling each resulting sublayer with an integrated LSTM-CNN-CBAM (Convolutional Block Attention Module) network. Zhang et al. [21] pushed the decomposition idea further still with a secondary decomposition scheme, arguing that most existing decomposition-ensemble models discard the residual error information left over after the primary decomposition step, and consequently applying Variational Mode Decomposition to the price series itself and a separate Complete Ensemble Empirical Mode Decomposition to the resulting error sequence, explicitly correcting for what earlier fixed-level decompositions ignored. This progression, from fixed-resolution wavelet sub-bands, to adaptive ensemble decomposition, to secondary error-decomposition, demonstrably improves the resulting model's ability to isolate genuine trend and cyclical components from noise. Its cost, however, is substantial added pipeline complexity and computational overhead: each decomposition stage must itself be fit, validated, and inverted, and the number of downstream sub-models scales with the number of decomposed components. The present study achieves comparable noise-robustness through an architectural rather than a signal-processing mechanism, the convolutional branch's local filtering combined with the gated fusion's ability to down-weight noisy learned features in favor of the raw skip-branch signal, without introducing a separate decomposition stage or its associated per-component modeling overhead.

A further refinement combines signal decomposition with metaheuristic or statistical optimization routines that automatically tune the decomposition's own parameters or the downstream model's hyperparameters, rather than fixing them by hand. Li et al. [22] addressed the non-stationary nature of gold futures by pairing Variational Mode Decomposition with the Iterated Cumulative Sums of Squares (ICSS) algorithm, which statistically detects structural breaks in variance to adaptively identify when the market regime, and therefore the appropriate decomposition, has shifted, before modelling each regime segment with a Bidirectional GRU; the resulting system maintained its accuracy advantage even under an algorithmic-trading simulation. In the energy domain, Yang et al. [23] combined Iterative Filtering Decomposition with a Kernel-based Extreme Learning Machine whose hyperparameters were tuned by a multi-objective Sine Cosine Algorithm, a population-based metaheuristic in the same family as particle swarm or genetic-algorithm search, jointly optimizing point-forecast accuracy and interval-forecast reliability rather than accepting a single hand-tuned configuration. Both studies show that wrapping a decomposition-ensemble pipeline in an automatic, data-driven optimization loop measurably improves on decomposition schemes with manually fixed parameters, since the optimizer can adapt the decomposition or model configuration to the specific volatility regime of the data at hand. This automatic tuning, however, does not eliminate the fundamental overhead problem identified in the preceding paragraph: it adds an outer optimization loop on top of an already multi-stage decomposition-ensemble pipeline, typically at substantial additional training cost, and the

optimizer's search itself introduces new hyperparameters, such as population size, iteration budget, and objective weighting, that must be chosen. The present study's gated fusion mechanism can be read as a lightweight, end-to-end alternative to this class of methods: the gate g is itself a learned, per-sample optimizer over how much to trust the deep branch versus the raw input, trained jointly with the rest of the network via standard backpropagation rather than as a separate metaheuristic search wrapped around a fixed pipeline.

Most recently, the field has begun moving beyond recurrent and decomposition-based backbones altogether, adopting transformer and graph-based architectures that were originally developed outside of financial forecasting. Bridging the decomposition and modern-architecture literatures, Dai et al. [24] proposed a parallel hybrid model for crude oil that retains secondary decomposition as a pre-processing step but replaces the recurrent backbone with a Transformer for global dependency extraction running alongside a CNN for local feature extraction, fusing the two through a fully connected layer. Zhao et al. [25] combined LSTM units directly with a Transformer through a multi-scale feature fusion module, allowing the model to integrate short-horizon technical indicators with longer-horizon macroeconomic signals for gold futures forecasting. Moving beyond sequence-only architectures entirely, Xue et al. [26] embedded a gated fusion mechanism inside a graph neural network for stock movement prediction, using a learned gate to adaptively balance coarse- and fine-grained multi-scale features across a stock relationship graph rather than relying on static concatenation or attention-based weighting, a recent, cross-domain confirmation that gated fusion mechanisms of the kind adopted in this study remain an actively developing technique in financial forecasting research more broadly. Alongside these transformer- and graph-based advances, large language models have very recently begun to be explored as general-purpose sequence forecasters, though their application to high-frequency single-asset commodity price prediction, as opposed to text-augmented or multi-asset settings, remains largely untested and falls outside the scope of the comparative evaluation in this study. Across this modern wave, self-attention and graph relational structure demonstrably capture long-range and cross-sectional dependencies that convolutional-recurrent hybrids cannot represent directly, but typically at the cost of substantially larger parameter counts, greater data requirements, and reduced interpretability. The present study intentionally targets a different point on this trade-off curve: it aims for a compact, parameter-efficient architecture, a two-branch CNN-LSTM with a single gating unit, suited to a single-asset, hourly-resolution forecasting task, rather than the larger-scale, richer-relational-structure setting these transformer- and graph-based methods are designed for.

Across the six strands of literature reviewed above, classical statistical models, standalone machine learning and deep learning architectures, sequential CNN-LSTM/CNN-GRU hybrids, decomposition-ensemble methods, optimization-augmented decomposition, and the emerging transformer/graph-based wave, the scientific community has made clear and cumulative progress: each successive strand demonstrably improves on some specific weakness of its predecessor, whether that is linearity, the absence of local feature extraction, noise from raw high-frequency data, or a fixed rather than adaptive decomposition. What remains comparatively underexplored, however, is the interface between the convolutional and recurrent components once they are combined: the CNN-LSTM literature reviewed in this section, and directly reimplemented as this study's Sequential CNN-LSTM Baseline, treats that interface as a fixed, one-way pipeline, implicitly assuming the convolutional branch's learned features are always preferable to the raw input rather than testing that assumption per sample. This is precisely the research gap this study addresses. By introducing a Dual-Branch CNN-LSTM architecture with Gated Fusion, in which a convolutional-recurrent deep branch and a raw-input skip branch are trained in parallel and adaptively combined through a learned, per-sample gate, this study reframes the convolutional-recurrent interface from a fixed architectural choice into an explicit, data-dependent quantity, while deliberately avoiding the added decomposition and optimization overhead reviewed above and remaining within the compact, single-asset scope for which a two-branch CNN-LSTM is well suited, as detailed in the Methodology that follows.

3. Methodology

This section delineates the comprehensive methodological framework adopted for the present investigation, systematically outlining the procedural sequence from data ingestion to performance assessment. Initial discourse focuses on the implementation of the sliding window technique, a critical mechanism for transforming continuous time series data into discrete supervised learning segments to facilitate temporal feature mapping (3.1). Subsequently, the structural configurations of the four distinct deep learning architectures are detailed, providing the technical specifications required for the comparative analysis of their predictive capabilities (3.2). The exposition further formalizes the algorithmic structure of the model training pipeline (3.3) and the specific computational logic utilized for the derivation of future price projections (3.4). Finally, Section 4 establishes the evaluative criteria and statistical significance tests employed to facilitate a rigorous comparative assessment of model efficacy and robustness.

3.1. Sliding Window-Based Temporal Data Transformation

The dataset is a multivariate time series $X \in R^{m \times n}$, where m is the number of timesteps and n is the number of features. It can be represented as a sequence of feature vectors $X = (x_1, x_2, \dots, x_m)$, where each $x_t \in R^n$. To prepare this sequential data for the supervised learning models, a sliding window technique is applied. Let w be the input window size and f be the forecast horizon. The time series is transformed

into a set of input-output pairs. An input sequence $X^{(i)}$ is defined as the sequence of w full feature vectors starting at time i: $X^{(i)} = (x_i, x_{i+1}, \dots, x_{i+w-1})$. Let c_t represent the 'Close' price at timestep t. The corresponding target sequence $Y^{(i)}$ consists only of the 'Close' price for the f steps following the input window: $Y^{(i)} = (c_{i+w}, c_{i+w+1}, \dots, c_{i+w+f-1})$

This sliding window transformation maps the original 2D tensor $X \in R^{m \times n}$ into two new tensors: a 3D tensor of input samples $X \in R^{k \times w \times n}$ and a 2D tensor of target samples $Y \in R^{k \times f}$, where $k = m - w - f + 1$ is the total number of generated samples. This resulting dataset facilitates model training. Algorithm 1 presents the pseudocode of the sliding window operation.

Algorithm 1: Sliding window technique

```
function slidingWindow(data, window, forecast)
    data_size = |data|
    X = []
    y = []
    for i from 0 to data_size - window - forecast:
        X = X U data[i : i + window, :]
        y = y U data[i + window : i + window + forecast, -1]
    return X, y
end function
```

3.2. Architectures of the Four Evaluated Models and Training Protocol

This study implements and evaluates four deep learning architectures, structured to directly isolate the contribution of the proposed fusion mechanism. Two single-branch baselines — a standalone 1D-CNN and a standalone LSTM — establish lower bounds for convolutional-only and recurrent-only feature extraction. A third baseline, Sequential CNN-LSTM, is a literal reimplementation of the architecture proposed in prior CNN-LSTM literature [4]: two stacked Conv1D layers (64 then 128 filters) feed a single LSTM layer (200 units) in a strictly sequential, single-branch pipeline, with no direct connection from the raw input to the output stage. This baseline is included specifically to give an explicit, like-for-like comparison against the prior CNN-LSTM literature this study builds on.

The proposed architecture, Dual-Branch CNN-LSTM with Gated Fusion, departs from this single-branch design in two respects. First, it processes the input through two parallel branches: a deep branch (Conv1D(64) → Conv1D(128) → LSTM(200), identical in depth to the Sequential CNN-LSTM Baseline) that learns non-linear spatiotemporal features, and a skip branch (Flatten→Dense(32)) that preserves the raw input signal without passing through any convolutional or recurrent abstraction. Second, the two branches are not concatenated with a fixed dense layer, as a naive two-branch design would do; instead they are combined through a gated fusion unit, adapted from the Gated Multimodal Unit [27], that learns a per-sample gate g in [0,1] controlling how much weight the deep branch receives relative to the skip branch: fused = g × h_deep + (1−g) × h_skip, where h_deep and h_skip are tanh-projected representations of the two branches and g = sigmoid(Dense([branch_deep, branch_skip])). This makes the reliance on learned convolutional-recurrent features versus raw input an explicit, data-dependent quantity rather than a fixed architectural choice — the concrete architectural difference from the Sequential CNN-LSTM Baseline [4] that motivates this study. All four architectures terminate in the same "common head" (Dense(32, ReLU) → Dropout(0.2) → linear output) and are trained with identical optimizer, loss, regularization (L2, SpatialDropout), and early-stopping settings to ensure a fair comparison. A visual representation of the proposed architecture and its gated fusion mechanism is illustrated in Fig 1.

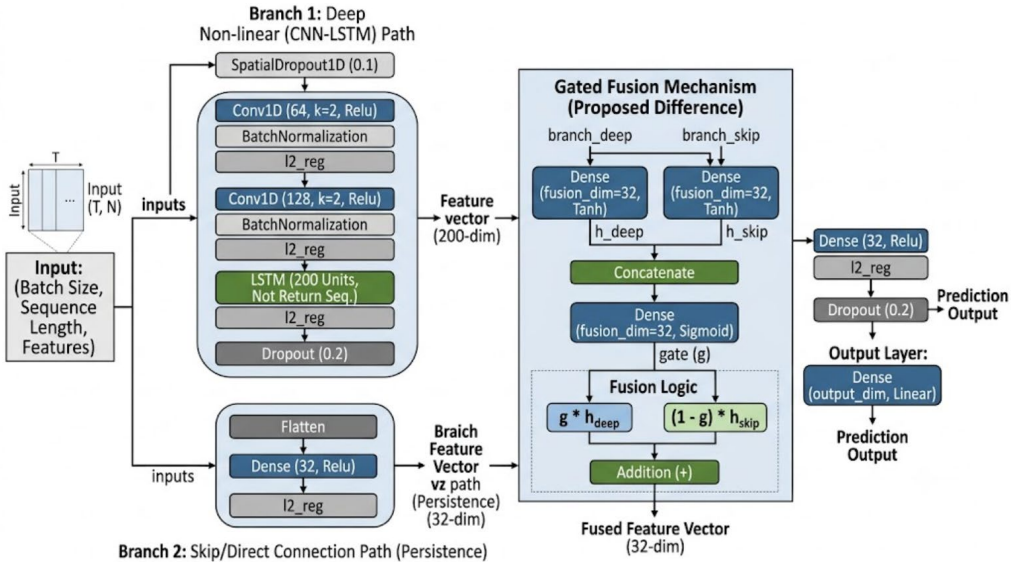

Figure 1. Architecture of the proposed Dual-Branch CNN-LSTM with Gated Fusion model

3.3. Model Training Process

Algorithm 2 formalizes the procedural framework governing data preparation and model optimization. Upon retrieval of the historical dataset, a strict partitioning protocol immediately stratifies the corpus into training (70%), validation (15%), and testing (15%) subsets, a measure designed to forestall data leakage and ensure rigorous evaluation. Feature normalization proceeds by fitting a scaler exclusively to the training partition; these parameters subsequently transform all subsets, thereby preserving the statistical independence of the validation and test data. Following this standardization, the slidingWindow function—previously defined in Algorithm 1—structures the data into input tensors X and target tensors y. The pipeline concludes as the selected architecture instantiates and optimizes parameters via the training tensors X_{train}, y_{train}, yielding a calibrated model alongside artifacts required for the inference phase.

Algorithm 2: Model Training Pipeline

```
function train_model(file, w, f, model_type, patience)
    data = read_data(file)
    train, val, test = temporal_split(data, 70, 15)
    scaler = fit_scaler(train)
    train_s, val_s, test_s = scaler.transform(train, val, test)
    X_train, y_train = create_sliding_window(train_s, w, f)
    X_val, y_val = create_sliding_window(val_s, w, f)
    model =
initialize_model(model_type).compile(optimizer='Adam',
loss='MSE')
    early_stop_cb = EarlyStopping(monitor='val_loss',
patience=patience)
    model.fit(X_train,y_train,validation_data=(X_val,y_val),
callbacks=[early_stop_cb])
    return model, scaler, X_test, y_test // X_test/y_test
creation assumed
end function
```

3.4. Predicting the Next n-Hour Gold Prices

The operational deployment of the forecasting framework utilizes a structured inference protocol, formally delineated in Algorithm 3, which integrates the pre-trained deep learning architecture with the normalization parameters established during the training phase. Upon the acquisition of raw market data, the input feature vectors undergo a standardization process and subsequent dimensionality restructuring to align with the specific tensor shape requirements dictated by the network topology. Following the execution of the forward pass, which generates predictions within the normalized interval, a reconstruction mechanism addresses the dimensionality constraints inherent in the multivariate scaler implementation. Specifically, the restoration of the predicted 'Close' prices to the original domain necessitates the injection of the scalar model outputs into a pre-initialized template matrix, thereby facilitating an inverse transformation that isolates and recovers the target variable while maintaining statistical consistency.

Algorithm 3: Predict next n-hour Gold prices

```
function predict_next_n_hours(raw_data, trained_model,
scaler):
    input_features = raw_data['open', ..., 'close']
    scaled_input = scaler.transform(input_features)
    reshaped_input = reshape(scaled_input)
    scaled_prediction =
trained_model.predict(reshaped_input)
    dummy_array = initialize_array(shape=(n, n_features))
    dummy_array[:, 4] = scaled_prediction[:, 0]
    inverse_transformed_output =
scaler.inverse_transform(dummy_array)
    final_price = inverse_transformed_output[:, 4]
    return final_price
end function
```

4. Experiment

The dataset used is XAUUSDm hourly (H1) Open-High-Low-Close-Volume (OHLCV) data from the Exness broker (Over-the-Counter (OTC) / Contract for Difference (CFD) via MetaTrader 5, UTC timezone), spanning January 2, 2020 to April 15, 2026 — 37,140 observations (~6.3 years), with zero missing values in the raw OHLCV columns and 1,621 time gaps (>1 hour, predominantly weekend/holiday closures, largest gap 3 days 6 hours). All OHLCV features are converted to log-return/ratio form, anchored to the previous close, to remove the scale drift induced by gold's sustained multi-year uptrend; a MinMaxScaler is fit exclusively on the training partition and applied to validation/test to prevent leakage. Three forecast configurations are evaluated, defined by (input window, forecast horizon) in hours: (24, 1), (48, 2), and (72, 3). The dataset is partitioned 70/15/15 into training, validation, and test sets in chronological order.

The study trains and evaluates all four architectures described in Section 3.2 (1D-CNN, LSTM, Sequential CNN-LSTM, and the proposed Dual-Branch CNN-LSTM with Gated Fusion) under identical optimizer, loss, regularization, and early-stopping settings, so that observed performance differences can be attributed to architecture rather than to training-budget differences.

The implementation and training of all models were conducted within the Google Colab[1] environment, leveraging its GPU support to accelerate computational processes. The TensorFlow[2] framework served as the primary deep learning library for constructing and training the neural network architectures. This setup provided a robust and efficient platform for executing the extensive series of experiments required for this research.

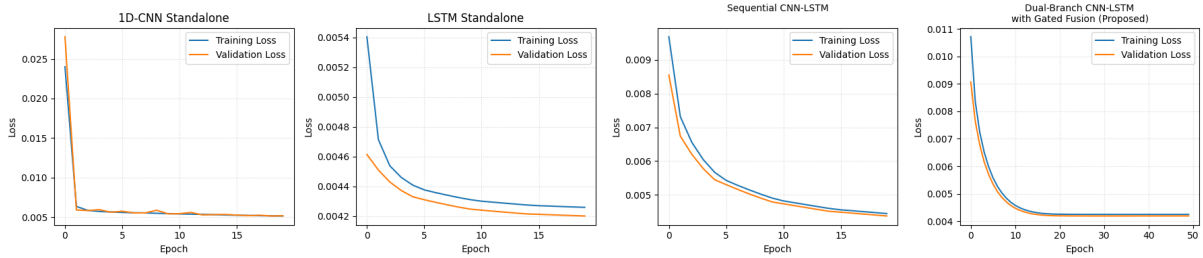

Fig 2. Training and validation loss curves for all four architectures, 24-hour input / 1-hour forecast configuration

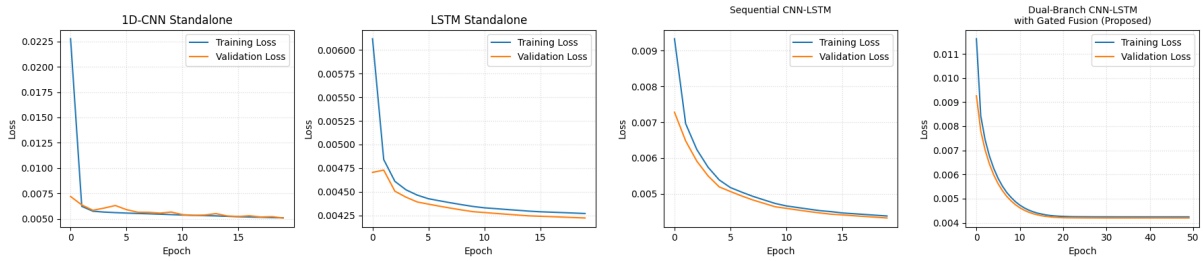

Fig 3. Training and validation loss curves for all four architectures, 48-hour input / 2-hour forecast configuration

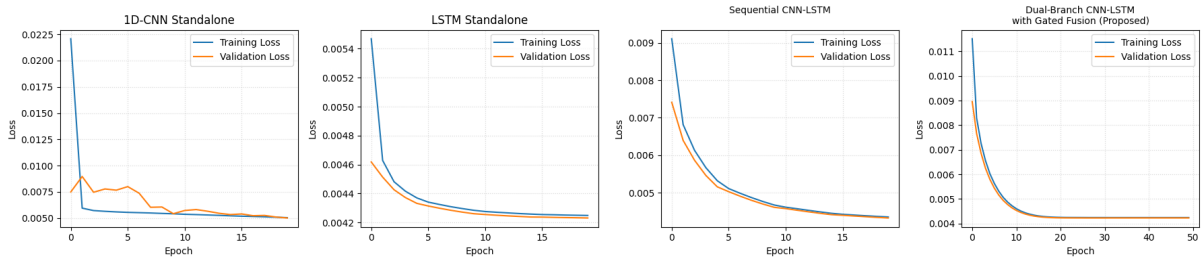

Fig 4. Training and validation loss curves for all four architectures, 72-hour input / 3-hour forecast configuration

Figures 2-4 show the training and validation loss curves for all four architectures across the three forecast configurations (24→1h, 48→2h, 72→3h). All models exhibit a sharp initial decrease in training loss followed by convergence to a stable value, with validation loss tracking training loss closely and without divergence as epochs increase, indicating that none of the four architectures overfit the training data under the early-stopping settings used.

In order to evaluate the trained models, the Mean Squared Error (MSE), Mean Absolute Error (MAE), Root Mean Squared Error (RMSE), and Coefficient of Determination (R^2) metrics are computed, with formulas listed in Eqs (1), (2), (3), and (4), respectively. The predicted values of the trained models over three test sets are visualized in Fig. 5, Fig. 6, and Fig. 7, while the evaluation results are presented in Table 1.

$$MSE(y, \hat{y}) = \frac{1}{N}\sum_{i=0}^{N-1}(y_i - \hat{y}_i)^2 \quad (1)$$

$$MAE(y, \hat{y}) = \frac{1}{N}\sum_{i=0}^{N-1}|y_i - \hat{y}_i| \quad (2)$$

$$RMSE(y, \hat{y}) = \sqrt{MSE(y, \hat{y})} \quad (3)$$

[1] https://colab.google/

[2] https://www.tensorflow.org/

$$R^2(y,\hat{y}) = 1 - \frac{\sum_{i=0}^{N-1}(y_i - \hat{y}_i)^2}{\sum_{i=0}^{N-1}(y_i - \bar{y})^2} \quad (4)$$

where N is the total samples; y and $\hat{y}$ are true and predicted values, respectively; and $\bar{y} = \frac{1}{N}\sum_{i=0}^{N-1} y_i$.

In addition to MSE, MAE, RMSE, and R^2 (Eqs. 1-4), Directional Accuracy (DA) is reported to assess whether a model correctly predicts the sign of the price change, independent of its magnitude:

$$\text{DA} = \frac{1}{N}\sum_{i=1}^{N} \mathbb{I}\left(\text{sign}(y_i - y_{i-1}) = \text{sign}(\hat{y}_i - y_{i-1})\right) \quad (5)$$

Following the reviewer comment on scale, all reported MAE, RMSE, Mean Absolute Percentage Error (MAPE), and R^2 values are computed after inverse-transforming model outputs from the normalized training scale back to the original USD price level; DA is computed on the corresponding sign of the (inverse-transformed) price change, not on the normalized residual.

Point-estimate differences in MAE/RMSE between models can arise from sampling noise rather than genuine accuracy differences. We therefore apply the Diebold-Mariano test (Diebold and Mariano, 1995; Harvey-Leybourne-Newbold small-sample correction) to the paired residual series of the proposed model against each baseline, testing the null hypothesis of equal expected loss. Because directional accuracy is a paired binary outcome (correct/incorrect sign per sample) rather than a continuous loss, we additionally apply McNemar's test (exact binomial variant for small disagreement counts, chi-square with continuity correction otherwise) to the paired hit/miss sequences, testing whether disagreements between the proposed model and each baseline are symmetric. Results of both tests are reported alongside Table 1. Evaluation of model performances.

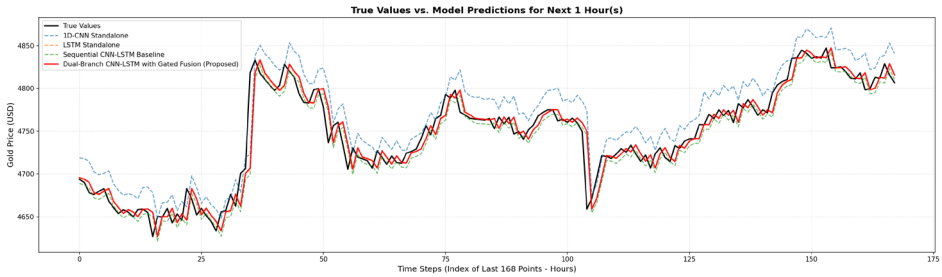

Fig 5. True values vs. model predictions (all four architectures) for the 24-hour input / 1-hour forecast configuration

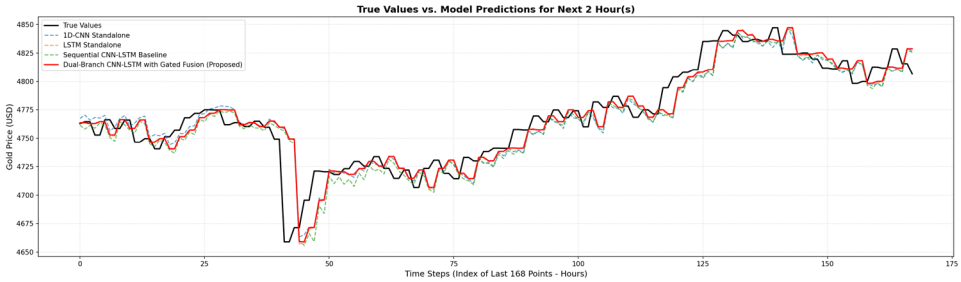

Fig 6. True values vs. model predictions (all four architectures) for the 48-hour input / 2-hour forecast configuration

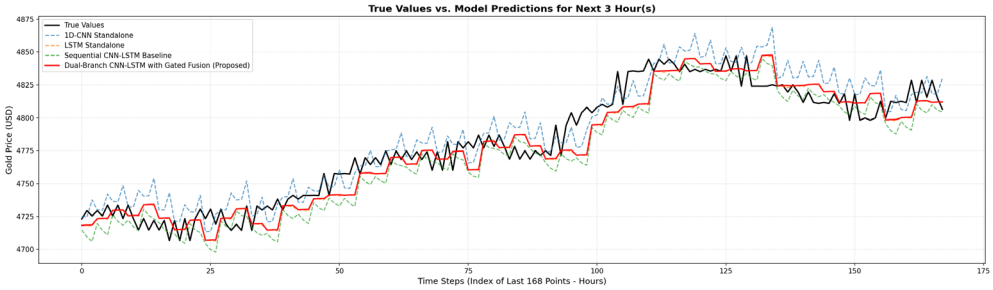

Fig 7. True values vs. model predictions (all four architectures) for the 72-hour input / 3-hour forecast configuration

Table 1 reports MAE, RMSE, MAPE, and R^2 on the original USD price scale, together with DA, across all three forecast configurations.

Table 1. Evaluation of model performances

Window (h)	Model	MAE (USD)	RMSE (USD)	MAPE (%)	R^2	DA (%)
24 → 1	1D-CNN	20.02	24.37	0.4933	0.99857	52.41
24 → 1	LSTM	9.03	15.74	0.2131	0.99940	48.49
24 → 1	Sequential CNN-LSTM	10.19	16.47	0.2418	0.99935	47.43
24 → 1	**Dual-Branch CNN-LSTM**	**9.00**	**15.72**	**0.2124**	**0.99941**	**52.57**
48 → 2	1D-CNN	12.09	19.88	0.2899	0.99905	51.20
48 → 2	LSTM	10.98	19.12	0.2590	0.99912	49.95
48 → 2	Sequential CNN-LSTM	11.42	19.49	0.2689	0.99909	47.41
48 → 2	**Dual-Branch CNN-LSTM**	**10.93**	**19.12**	**0.2579**	**0.99912**	**52.54**
72 → 3	1D-CNN	15.29	25.67	0.3617	0.99842	51.76
72 → 3	LSTM	12.58	21.89	0.2966	0.99885	50.02
72 → 3	Sequential CNN-LSTM	13.53	22.53	0.3195	0.99878	47.63
72 → 3	**Dual-Branch CNN-LSTM**	**12.54**	**21.88**	**0.2957**	**0.99885**	**52.53**

The proposed Dual-Branch CNN-LSTM with Gated Fusion attains the best or tied-best value on every metric in every configuration. Its magnitude-error advantage over the standalone LSTM is marginal (e.g., 24→1h: MAE 9.00 vs 9.03 USD), while its advantage over the 1D-CNN and the Sequential CNN-LSTM Baseline [4] is substantial across all three configurations (e.g., 24→1h: MAE 9.00 vs 20.02 and 10.19 USD, respectively). Notably, the Sequential CNN-LSTM Baseline, despite closely matching the proposed model's magnitude-error metrics, records the lowest DA of the four architectures in every configuration (47.4-47.6%, below the 50% random-guess threshold), whereas the proposed model is the only architecture whose DA consistently exceeds 52% across all configurations — indicating that its dual-branch, gated design translates into a genuine (if modest) edge in predicting the direction of price movement, not merely the magnitude.

Table 2. Diebold-Mariano test (squared-error loss, proposed model vs. each baseline)

Baseline	24→1h	48→2h	72→3h
1D-CNN	-46.69 / <.001	-5.84 / <.001	-11.46 / <.001
LSTM	-1.22 / .223	-0.21 / .831	-0.33 / .743
Sequential CNN-LSTM	-9.20 / <.001	-3.77 / <.001	-6.51 / <.001

Table 3. McNemar test (directional hit/miss, proposed model vs. each baseline)

Baseline	24→1h	48→2h	72→3h
1D-CNN	1.25 / .263	8.58 / .003	6.26 / .012

LSTM	14.57 / <.001	14.58 / <.001	20.07 / <.001
Sequential CNN-LSTM	14.59 / <.001	29.52 / <.001	42.26 / <.001

The Diebold-Mariano test (Table 2) shows that the proposed model's continuous-error advantage over 1D-CNN and the Sequential CNN-LSTM Baseline is statistically significant ($p < 0.001$) in all three configurations, but its difference from the standalone LSTM is not significant ($p = 0.22$-0.83) in any configuration — the two models are, in terms of magnitude error, statistically indistinguishable. The McNemar test (Table 3) tells a different story for directional accuracy: the proposed model significantly outperforms both LSTM and the Sequential CNN-LSTM Baseline ($p < 0.001$) in all three configurations and significantly outperforms 1D-CNN in two of three (48→2h, 72→3h; the 24→1h difference is not significant, $p = 0.263$). Overall, across the 9 pairwise directional-accuracy comparisons (three baselines × three configurations), the proposed model's McNemar advantage is statistically significant ($p < 0.05$) in 8 of 9 cases; the sole exception is the 24→1h comparison against 1D-CNN, where the two models' DA (52.57% vs 52.41%) is statistically indistinguishable. This makes the directional-accuracy gain — not the magnitude-error reduction, where the proposed model ties with plain LSTM — the most consistent, statistically supported finding of this study.

The empirical advantage of the proposed architecture is attributable to the complementary computational mechanisms utilized for noise reduction and temporal abstraction, combined with the gated fusion mechanism that adaptively weights the deep (CNN→LSTM) branch against the raw-input skip branch on a per-sample basis. Specifically, the intrinsic volatility and microstructure noise characterizing high-frequency gold price data often obstruct the feature extraction capabilities of standalone LSTM networks when they operate directly on raw sequences. By preceding the recurrent layers with 1D-CNN operations and retaining a direct skip path to the raw input, the proposed framework executes an initial dimensionality reduction and spatial filtering process while preserving a fallback signal the gate can rely on when the deep branch's learned representation is less informative, unlike the single-branch Sequential CNN-LSTM Baseline, which has no such fallback.

5. Conclusion

The present investigation established a Dual-Branch CNN-LSTM framework with Gated Fusion to address the complexities inherent in gold price forecasting, benchmarked directly against a standalone 1D-CNN, a standalone LSTM, and a literal reimplementation of the Sequential CNN-LSTM Baseline architecture from prior literature [4]. Empirical evaluations conducted across three forecast configurations on 6.3 years of hourly XAUUSDm data demonstrate that the proposed architecture attains the best or tied-best MAE, RMSE, R^2, and DA in every configuration, with its directional-accuracy advantage confirmed by McNemar's test and its magnitude-error advantage over 1D-CNN and the Sequential CNN-LSTM Baseline confirmed by the Diebold-Mariano test.

The systematic training pipeline, utilizing a sliding window algorithm for multivariate data transformation, ensures that the model captures the non-linear dynamics of intraday trading environments without incurring the computational overhead of secondary decomposition methods. By partitioning the dataset into distinct training, validation, and testing subsets, the study maintained a rigorous evaluative protocol that prevented data leakage and confirmed the generalization capabilities of the hybrid model. The consistent convergence observed in the training and validation loss curves further validates the stability of the optimization process and the effectiveness of the dropout regularization mechanisms. These procedural advancements contribute a standardized methodology for the application of deep learning to financial time series characterized by microstructure noise and irregular volatility.

A caveat is warranted regarding the R^2 values in Table 1, which remain very high (0.998-0.999) across all models and configurations, including the weakest baseline. This is expected and should not be read as evidence of strong short-horizon predictability: the forecasting target is the absolute USD price level 1-3 hours ahead, and gold's hour-to-hour price change is small relative to its overall price range over the 2020-2026 sample, so even a naive persistence forecast would already achieve R^2 in a similar range. The more informative metric for assessing whether a model captures genuine predictive signal, rather than simply tracking a slow-moving level, is DA. Here the picture is far more modest: DA ranges from 47.4% to 52.6% across all models and configurations — only the proposed model consistently exceeds the 50% coin-flip threshold, and by no more than about 2.6 percentage points. We report and discuss this distinction explicitly to avoid the overoptimistic-performance impression that near-perfect R^2 scores can otherwise create, and we frame the results of this study as evidence of a modest, statistically detectable directional edge rather than a validated forecasting or trading signal.

This study has two limitations that bound the claims made above. First, no economic or transaction-cost evaluation (e.g., Sharpe ratio, backtest with realistic spread / slippage / commission, or profit-and-loss simulation) was performed; the reported DA advantage, while statistically significant, does not by itself establish that the proposed model would be profitable net of trading costs, particularly given that its directional edge over chance is modest (2-3 percentage points). Consequently, the results presented here should be read as an architectural and statistical improvement over prior CNN-LSTM designs, not as a validated basis for algorithmic trading or live deployment. Second, the robustness checks performed (multi-seed re-training on a fixed chronological split) test sensitivity to random initialization only, and do not test stability across distinct market regimes (e.g., the 2020 COVID shock, the 2022 rate-hike cycle, and 2024-2026 volatility) that a walk-forward evaluation would probe; the two forms of robustness are complementary, not substitutable. Future work should prioritize a dedicated economic backtest with realistic cost assumptions and a walk-

forward evaluation across market regimes before any consideration of practical trading use, together with the extension of the gated fusion mechanism to attention-based and multi-asset settings.

Funding Information

This study was funded by Hue University, Viet Nam (grant number DHH2025-01-225).

References

[1] Zhang C, Sjarif NNA, Ibrahim R. Deep learning models for price forecasting of financial time series: a review of recent advancements: 2020-2022. WIREs Data Min Knowl Discov. 2024;14(1):e1519. doi:10.1002/widm.1519.

[2] Li W, Law KLE. Deep learning models for time series forecasting: a review. IEEE Access. 2024;12:92306-92327. doi:10.1109/ACCESS.2024.3422528.

[3] Foroutan P, Lahmiri S. Deep learning systems for forecasting the prices of crude oil and precious metals. Financ Innov. 2024;10(1):111. doi:10.1186/s40854-024-00637-z.

[4] Livieris IE, Pintelas E, Pintelas P. A CNN-LSTM model for gold price time-series forecasting. Neural Comput Appl. 2020;32(23):17351-17360. doi:10.1007/s00521-020-04867-x.

[5] You W, Chen J, Xie H, Ren Y. Which uncertainty measure better predicts gold prices? New evidence from a CNN-LSTM approach. N Am J Econ Finance. 2025;76:102375. doi:10.1016/j.najef.2025.102375.

[6] Kong X, et al. Deep learning for time series forecasting: a survey. Int J Mach Learn Cybern. 2025;16(7-8):5079-5112. doi:10.1007/s13042-025-02560-w.

[7] Nallamothu S, Rajyalakshmi K, Arumugam P. Gold price prediction using skewness and kurtosis based generalized auto-regressive conditional heteroskedasticity approach with long short term memory network. J Inst Eng India Ser B. 2024;105(6):1715-1727. doi:10.1007/s40031-024-01070-7.

[8] Cohen G, Aiche A. Forecasting gold price using machine learning methodologies. Chaos Solitons Fractals. 2023;175:114079. doi:10.1016/j.chaos.2023.114079.

[9] Jabeur SB, Mefteh-Wali S, Viviani JL. Forecasting gold price with the XGBoost algorithm and SHAP interaction values. Ann Oper Res. 2024;334(1-3):679-699. doi:10.1007/s10479-021-04187-w.

[10] Abu-Doush I, Ahmed B, Awadallah MA, Al-Betar MA, Rababaah AR. Enhancing multilayer perceptron neural network using archive-based Harris hawks optimizer to predict gold prices. J King Saud Univ Comput Inf Sci. 2023;35(5):101557. doi:10.1016/j.jksuci.2023.101557.

[11] Weng F, Chen Y, Wang Z, Hou M, Luo J, Tian Z. Gold price forecasting research based on an improved online extreme learning machine algorithm. J Ambient Intell Humaniz Comput. 2020;11(10):4101-4111. doi:10.1007/s12652-020-01682-z.

[12] Memon BA, Tahir R, Naveed HM, Cheng K. Forecasting gold and platinum prices with an enhanced GRU model using multi-headed attention and skip connection. Miner Econ. 2025. doi:10.1007/s13563-025-00520-y.

[13] Wang J, Li Y, Wang T, Li J, Wang H, Liu P. A gold futures price forecast model based on SGRU-AM. IEEE Access. 2021;9:146745-146754. doi:10.1109/ACCESS.2021.3122140.

[14] Shahi TB, Shrestha A, Neupane A, Guo W. Stock price forecasting with deep learning: a comparative study. Mathematics. 2020;8(9):1441. doi:10.3390/math8091441.

[15] Chi DTK, Kien HNT, Nguyen TQ. Enhancing forex market forecasting with feature-augmented multivariate LSTM models using real-time data. Knowl Based Syst. 2025;330:114500. doi:10.1016/j.knosys.2025.114500.

[16] Alzakari SA, Alhussan AA, Qenawy AST, Elshewey AM, Eed M. An enhanced long short-term memory recurrent neural network deep learning model for potato price prediction. Potato Res. 2024. doi:10.1007/s11540-024-09744-x.

[17] Manogna RL, Dharmaji V, Sarang S. Enhancing agricultural commodity price forecasting with deep learning. Sci Rep. 2025;15(1):20903. doi:10.1038/s41598-025-05103-z.

[18] Xu Y, Liu T, Fang Q, Du P, Wang J. Crude oil price forecasting with multivariate selection, machine learning, and a nonlinear combination strategy. Eng Appl Artif Intell. 2025;139:109510. doi:10.1016/j.engappai.2024.109510.

[19] Zhang X, Zhang L, Zhou Q, Jin X. A novel Bitcoin and gold prices prediction method using an LSTM-P neural network model. Comput Intell Neurosci. 2022;2022:1-12. doi:10.1155/2022/1643413.

[20] Liang Y, Lin Y, Lu Q. Forecasting gold price using a novel hybrid model with ICEEMDAN and LSTM-CNN-CBAM. Expert Syst Appl. 2022;206:117847. doi:10.1016/j.eswa.2022.117847.

[21] Zhang Y, Peng Y, Song Y. Metal commodity futures price forecasting based on a hybrid secondary decomposition error-corrected model. J Big Data. 2025;12(1):166. doi:10.1186/s40537-025-01240-4.

[22] Li Y, Wang S, Wei Y, Zhu Q. A new hybrid VMD-ICSS-BiGRU approach for gold futures price forecasting and algorithmic trading. IEEE Trans Comput Soc Syst. 2021;8(6):1357-1368. doi:10.1109/TCSS.2021.3084847.

[23] Yang W, Sun S, Hao Y, Wang S. A novel machine learning-based electricity price forecasting model based on optimal model selection strategy. Energy. 2022;238:121989. doi:10.1016/j.energy.2021.121989.

[24] Dai Z, Huang H, Jiang Q, Chen Y. A parallel combined Transformer-CNN model using secondary decomposition for crude oil forecasting. Expert Syst Appl. 2026;299:129968. doi:10.1016/j.eswa.2025.129968.

[25] Zhao Y, Guo Y, Wang X. Hybrid LSTM-Transformer architecture with multi-scale feature fusion for high-accuracy gold futures price forecasting. Mathematics. 2025;13(10):1551. doi:10.3390/math13101551.

[26] Xue X, Duan P, Liu Z, Chu Q, Zhang C, Zhang B. Gated fusion enhanced multi-scale hierarchical graph convolutional network for stock movement prediction. In: Neural Information Processing (ICONIP 2025). Lecture Notes in Computer Science, vol 16311. Singapore: Springer; 2026. p. 381-395. doi:10.1007/978-981-95-4381-6_26.

[27] Arevalo J, Solorio T, Montes-y-Gómez M, González FA. Gated multimodal units for information fusion. arXiv:1702.01992 [Preprint]. 2017.

[28] Limbare A, Agarwal R. Demand forecasting and budget planning for automotive supply chain. EAI Endorsed Trans IoT. 2023;10. doi:10.4108/eetiot.4514.