Comparative Analysis of ARIMA and Transformer-Based Models for Gold Price Forecasting

Ziqi Zhao, Jiaming Huang, Yuhu Xue

Xi'an Jiaotong-Liverpool University, Suzhou, 215123, China

Abstract. In recent years, the powerful capabilities of deep learning models have attracted considerable attention from both financial investors and researchers. Emerging architectures such as Transformer and its variants (e.g., Autoformer, Informer) have demonstrated superior ability in capturing nonlinear patterns within long-term time series, leading to their widespread application in tasks such as forecasting financial market indices. However, it remains unclear whether such models are universally effective across all financial prediction scenarios. In this study, we focus on a specific domain of financial forecasting—gold price prediction—and conduct extensive experiments on two datasets. The results indicate that traditional statistical models, such as ARIMA, achieve higher predictive accuracy than deep learning models. This finding highlights the importance of carefully considering the inherent characteristics of time-series data. In contexts such as gold price prediction, where short-term linear patterns dominate, simple statistical model may prove more effective.

Keywords: ARIMA, Transformer, Gold Price Forecasting, Financial Time Series, Deep Learning.

1. Introduction

Gold is recognized as serving multiple roles in the global economy. Beyond its monetary and industrial functions, it is widely used for portfolio diversification and as a safe-haven asset in turbulent periods[1]. Therefore, gold prices are viewed as a key information variable, and accurate prediction of gold price has become crucial for policy-making and investment decisions.

Gold prices are driven by several factors—chiefly inflation, interest rates, the U.S. dollar exchange rate, oil prices, and overall financial conditions [2]. Relative to equities, gold prices are more heavily shaped by macroeconomic and sentiment dynamics and exhibit greater structural complexity. Traditional statistical models (e.g., ARIMA [3]) enable rapid forecasting from historical data, but errors accumulate as the horizon lengthens and multivariate relationships are not modeled. In response, research on gold-price prediction has increasingly dedicated to deep learning. Long-sequence time-series forecasting (LSTF [2]) can infer extended-horizon dynamics from historical time series and is therefore well suited to gold-price prediction. Within LSTF, Transformer [4] is valued for modeling global dependencies and—owing to its efficiency, flexibility, and scalability—has become the prevailing backbone for time-series tasks. Nevertheless, the quadratic cost of self-attention induces memory bottlenecks for long inputs and deep stacks and causes marked inference slowdowns with long contexts, constraining their suitability for LSTF. To mitigate these limitations, the Informer [5], the Autoformer [6], and the Reformer [7], which are based on Transformer, have been proposed. Thereby, substantial performance gains have been achieved in long-sequence time-series forecasting (LSTF).

Despite the strong performance of Transformer-based models in LSTF, in this paper, we show that ARIMA can outperform them on long-horizon gold forecasting under specific conditions. On two Kaggle datasets, ARIMA attains higher predictive accuracy at substantially lower computational cost than Transformer-based deep learning baselines. These findings suggest a practical guideline for the domain: when datasets are small and few exogenous variables are available, classical statistical models (e.g., ARIMA) can be more effective.

The contributions of this paper are as follows:

- We conduct extensive experiments on datasets that include multiple features. We find that under small-sample settings with few exogenous variables, ARIMA outperforms Transformer-based deep learning methods.

- We consider multiple drivers of gold price. Models are fitted on datasets containing stock prices, crude oil prices, interest rates, and related variables, and the impact of individual features on forecasting accuracy is systematically evaluated.

2. Relevant Work

Gold-price forecasting has traditionally been conducted with statistical models such as ARIMA[3], VAR [8], and GARCH [9]. By capturing short-term dynamics and linear structure, these models provide strong fit and interpretability, perform especially well on (near-)stationary data, and yield high short-horizon accuracy [10]. The forecasting process first models the conditional mean with ARIMA or VAR [8]. Then, volatility was modeled on the residuals using GARCH[9] to obtain confidence bands [11]. However, because this framework is essentially suited to short-horizon, near-linear dynamics, it is less effective when confronting the mid-and-long term nonlinear relationships prevalent in financial markets[12][13]. Additionally, traditional models tend to deteriorate during financial-market black swan events, frequently yielding lower predictive accuracy and related failures [14]. Therefore, traditional statistical models exhibit substantial limitations in forecasting gold prices [15][16].

To address these limitations, recurrent neural networks—particularly LSTM—have been adopted to model nonlinear dynamics and mid-and-long term relationships in financial time series [17][18][18]. While LSTM addresses issues of vanishing gradients through gating mechanisms[19] and performs well on short-horizon trend prediction [20], RNN-based models remain constrained by poor parallelism and long training times [21]and by limited capacity to capture long-range dependencies [22][23]. In complex financial-market settings, their predictive accuracy—and broader performance—still falls short of desired levels [24][25][26][26].

In recent years, Transformer models have been widely applied to time-series modeling tasks owing to their global attention mechanism and strong processing capacity [23]. They provide a powerful sequence-learning framework for representing long-range dependencies [27]. In gold-price forecasting, Transformer models can overcome the short-memory bottleneck of traditional approaches [28], help extract intrinsic patterns from sequential data in a more flexible and efficient manner [29]and have gradually become a vital direction in financial time-series modeling[30]. Specifically, Informer, as an improved Transformer architecture, enhances the efficiency and accuracy of long-horizon time-series forecasting via the ProbSparse self-attention mechanism and distillation operations [29], and It has shown strong empirical performance across financial-asset prediction tasks, including gold [31][32].

In our gold-price forecasting study, the Informer model was introduced, as an efficient, improved Transformer variant, it employs a probabilistic sparse self-attention mechanism that reduces computational complexity and substantially enhances modeling capacity and predictive accuracy for long sequences. Experiments indicate clear gains over traditional statistical baselines and a vanilla Transformer, with stronger generalization, more stable predictions, and sharper capture of market volatility, make it well suited to complex financial settings.

3. Model Architecture

In this chapter, we briefly introduce the model architectures we investigate, including ARIMA, Transformer, and some Transformer variants.

3.1 ARIMA

The AutoRegression Integrated Moving Average (ARIMA) model is a classical time series analysis tool, widely applied in trend forecasting problems in fields such as economics, finance, meteorology, and energy. The model is constructed through three components—AutoRegression (AR), Integration (I), and Moving Average (MA)—and has strong modeling and forecasting

capabilities. This section systematically introduces the theoretical foundation, modeling process, and usage conditions of the ARIMA model, and points out through analysis that despite the rise of modern deep learning methods, ARIMA still holds irreplaceable value. The core of ARIMA includes the AutoRegression (AR) part, the Integrated (I) part, and the Moving Average (MA) part, which we briefly introduce as follows:

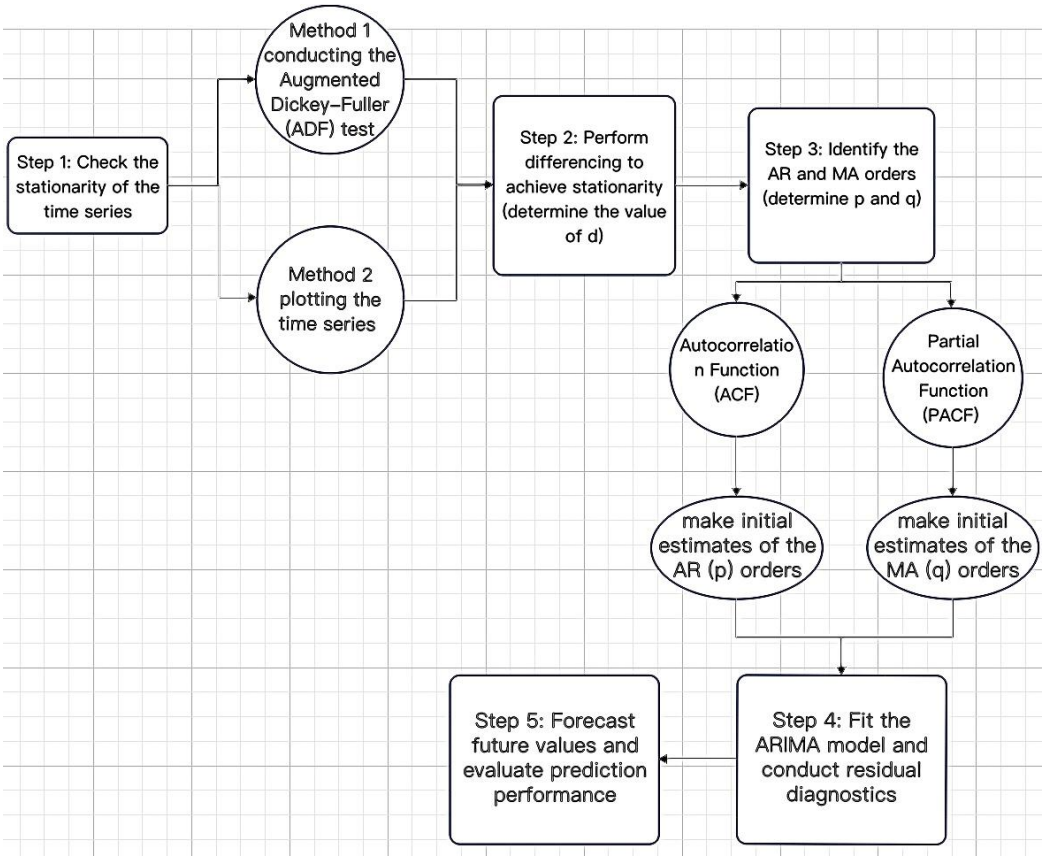

Figure 1. Modeling Procedure of ARIMA

AutoRegression (AR). The AutoRegression (AR) [33] part represents a linear relationship between the current value and several past values. Its mathematical expression is:

$$AR(p): X_t = \phi_1 X_{t-1} + \phi_2 X_{t-2} + \cdots + \phi_p X_{t-p} + \varepsilon_t, \tag{1}$$

where X_t is the value at time t, ϕ_i are the autoregressive coefficients, and ε_t is the white noise error term.The autoregressive coefficients ϕ_i describe the strength of the linear relationship between the current X_t and past values $X_{t-1}, X_{t-2}, \ldots, X_{t-p}$, reflecting the "memory" effect of past values on the present and determining the trend pattern. A larger$|\phi_i|$indicates stronger memory and slower changes; a smaller$|\phi_i|$ indicates weaker memory and faster changes, closer to white noise. The white noise error term ε_t introduces unpredictable random fluctuations (noise), simulating unexplained external disturbances such as sudden events or measurement errors, thus adding randomness and realism to the model.

Integrated (I). The Integrated (I) [34] part is used to transform a non-stationary time series into a stationary one. By applying differencing d times, the series becomes stationary. The integrated operator is $\nabla X_t = X_t - X_{t-1}$. If integrated is applied d times, it is denoted as:

$$(1 - B)^d X_t, \tag{2}$$

where B is the lag operator, i.e., $BX_t = X_{t-1}$. To determine the required number of Integrated steps, the ADF (Augmented Dickey-Fuller) test can be applied to check stationarity. If the series is

non-stationary (p-value > 0.05), a first Integration is performed; if still non-stationary, further Integration is applied until the series becomes stationary.

Moving Average (MA). The Moving Average (MA) [35]part represents the influence of a linear combination of past error terms on the current value. Its mathematical expression is:

$$MA(q): X_t = \mu + \varepsilon_t + \theta_1 \varepsilon_{t-1} + \theta_2 \varepsilon_{t-2} + \cdots + \theta_q \varepsilon_{t-q}, \tag{3}$$

where X_t is the observed value at time t, μ is the mean of the series, ε_t is white noise, and θ_j are the moving average coefficients. The moving average coefficients θ_j represent the effect of the j-th lagged error on the current value. By incorporating past prediction errors, the MA component serves as a "regression correction mechanism". It also captures short-term shock effects, allowing disturbances in the past to have "aftershock-like" influences on future values. In addition, it models the short-term memory of unexplainable components, thereby improving fitting accuracy.

Description of Integrated Model. The ARIMA model is denoted as:

$$ARIMA(p, d, q): \Phi(B)(1 - B)^d X_t = \Theta(B)\varepsilon_t, \tag{4}$$

where B is the lag operator, $\Phi(B)$ is the AR polynomial, $\Theta(B)$ is the MA polynomial. Figure 1 shows the modeling procedure of ARIMA. The ARIMA modeling process involves checking stationarity, applying differencing if needed, identifying AR and MA orders using ACF and PACF, fitting the model with residual diagnostics, and finally forecasting and evaluating performance.

3.2 Transformer

As one of the core architectures in the field of Natural Language Processing (NLP) [36], the Transformer has rapidly replaced traditional Recurrent Neural Networks (RNNs) [37] and Convolutional Neural Networks (CNNs) [38] in many tasks due to its parallel computing capability and superior performance. The central idea of the Transformer is to leverage the Self-Attention mechanism to perform global modeling of input sequences, thereby enabling efficient and parallel data processing.

The Transformer model is mainly composed of two parts: the Encoder and the Decoder. Each part consists of multiple identical layers stacked together. Each encoder layer includes a multi-head self-attention mechanism and a feed-forward neural network, with residual connections and layer normalization to ensure training stability [39]. The encoder enables global modeling of the relationships among tokens in the input sequence without relying on recurrence or convolution, thereby extracting contextual semantic representations. The decoder consists of multiple stacked decoder layers, each containing masked multi-head self-attention, encoder-decoder attention, and a feed-forward neural network, also combined with residual connections and layer normalization. During sequence generation, the masking mechanism ensures that the decoder can only access the current token and its preceding tokens. By interacting with the encoder outputs, the decoder incorporates source sequence information and generates the target sequence step by step. The Self-Attention mechanism allows the model to focus on different parts of the sequence when processing it. The input sequence is mapped into three vectors, namely $Q = XW^Q$, $K = XW^K$, and $V = XW^V$. Here, W^Q, W^K, W^V denotes the learnable weight matrices, $X \in \mathbb{R}^{n \times d}$ is the long input sequence (with length n and embedding dimension d).

For each token position i, its attention weight is determined by computing $Queryq_i$ and the similarity with all other tokens $Keyk_j$, using the attention scoring function:

$$Attention(Q,K,V)=softmax\left(\frac{QK^T}{\sqrt{d_k}}\right)V. \tag{5}$$

In this formulation, $QK^T \in \mathbb{R}^{n\times n}$ calculates the pairwise similarity between tokens. The scaling factor $\sqrt{d_k}$ prevents the dot product from becoming excessively large, which could otherwise negatively impact the $softmax$ outputs. The $softmax$ function transforms the scores into a probability distribution, which is then used to compute a weighted sum of the value vectors, yielding new contextualized token representations. This mechanism enables the model to dynamically adjust the importance of each token according to its context [40].

3.3 Modification of Transformer

Here, we present several representative Transformer variants, including Informer, Autoformer, and Reformer. These methods will serve as baselines to highlight the advantages of the ARIMA algorithm in gold price forecasting.

Informer. Zhou et al. (2021) proposed the Informer model[5]to address the high complexity and large memory consumption of Transformer. The ProbSparse Attention mechanism is the core modification of Informer compared to the standard Transformer. The key idea is based on a probabilistic sparsity assumption: only a small portion of query vectors have a decisive impact on the overall attention distribution. Specifically, for each query, the attention distribution is often highly skewed, with one or two values dominating. Therefore, Informer proposes that only the top-u queries (with the highest normalized entropy scores) need to be computed with full attention, while the remaining queries are approximated or ignored. With ProbSparse Attention, the computational complexity is reduced from $O(L^2)$ to $O(L\log L)$,where L is the sequence length,$\log L$ is the number of sparsely selected queries. This allows Informer to remain efficient and scalable even when processing very long time series.

Autoformer. The standard Transformer suffers from high computational cost and lack of structural priors, limiting its effectiveness in time series tasks [41]. To address this, Autoformer integrates Transformer's global modeling with trend–seasonality priors through a Series Decomposition Block [6]. A time series x(t) is decomposed into trend T(t) and seasonal S(t) components using a learnable moving average operator [42]. Implemented via 1D convolution, residual subtraction, and independent modeling of trend and seasonal parts, this design enhances interpretability, stability, and forecasting accuracy.

Reformer. Although the Transformer has achieved remarkable success in the field of natural language processing, it faces the problem of exponentially increasing computational and memory costs when dealing with long sequence tasks [7]. To address this challenge, Reformer introduces the Locality-Sensitive Hashing (LSH) attention mechanism and reversible residual layers, which significantly reduce both time and space complexity while preserving the performance of the transformer. Locality-Sensitive Hashing [43] is a probabilistic hashing method that maps vectors close in high-dimensional space into the same bucket. In Reformer, this idea is applied to self-attention: query–key vectors are grouped using a hash function, and attention is only computed among tokens within the same bucket. This sparsification dramatically reduces the computational cost of attention. The Reversible Residual Layer [44] is a structure that allows reconstruction of forward states during backpropagation without storing intermediate results, thereby significantly reducing memory consumption.

4. Experiment

In this section, we present the experimental setup, datasets, evaluation metrics, and the main experimental results.

4.1 Experimental Setup

All experiments were conducted on a workstation equipped with an NVIDIA RTX 4090 GPU, an Intel(R) Core (TM) i9-14900HX CPU (32 cores, ~2.2GHz), and 64 GB RAM, running Windows 11 as the operating system. The models were implemented using the Python programming language,

with PyTorch version 1.13.1+cpu as the deep learning framework. Model training and evaluation were conducted under the same random seed and batch settings. Early stopping was applied based on validation loss to prevent overfitting. All training time statistics refer to per-epoch runtime under these conditions.

4.2 Datasets

We conduct experiment on a dataset sourced from Kaggle. This dataset consists of eleven columns, including Date, SENSEX, Nifty, S&P, Oil Price, USD, Interest Rate, Gold Demand, S&P INR, Oil Price INR and Gold Price. These elements show different factors which influence gold price greatly [45].The datasets covers a large time information, ranging from January 4, 2010, to December 31, 2020, which allows the research to identify the primary determinants of gold price movements. The dataset comprises 2,670 samples and was utilized as input for the selected model, yielding valuable insights for the experiment.

4.3 Evaluation Metrics

To comprehensively evaluate the performance of the proposed model in forecasting gold price time series, two quantitative metrics were employed: Mean Squared Error (MSE) and Mean Absolute Error (MAE) [46].

Mean Squared Error (MSE): The MSE is defined as:

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

where n denotes the total number of forecast samples, y_i represents the actual value at the i-th time point, and $\hat{y}_i$ indicates the corresponding model prediction.

Mean Absolute Error (MAE): The MAE is defined as:

$$MAE = \frac{1}{n}\sum_{i=1}^{n}|\hat{y}_i - y_i|, \tag{7}$$

where the notations are consistent with those used in the MSE definition.

4.4 Comparison of Predictive Performance

We conduct a comparison on the Goldprice dataset using five models: the traditional statistical ARIMA model and four deep learning models—Autoformer, Informer, Reformer, and Transformer. Experimental results appear in Figure 3, and we make several key observations. First, although it is a traditional linear model, ARIMA demonstrates outstanding performance in this task, achieving an MSE of 0.0329 and an MAE of 0.1272 on the test set, and it outperforms all deep learning models. This result shows that for highly linear and interpretable financial time series, traditional models remain competitive. Its combination of efficient training and excellent error control makes ARIMA particularly valuable in scenarios with limited computational resources or strong requirements for interpretability. Second, Autoformer exhibits strong generalization and superior nonlinear modeling capability among the deep learning models. Although its errors exceed those of ARIMA, Autoformer outperforms all other deep models, benefiting from its trend–seasonal decomposition module and autocorrelation mechanism, which effectively capture both long-term and short-term structural patterns in gold prices. This architecture is especially suitable for time series with pronounced nonlinear variations. Third, Informer offers superior computational efficiency but slightly lower predictive accuracy than Autoformer. Reformer achieves the lowest validation loss and shows moderate generalization ability, making it suitable for balancing efficiency and stability. Transformer performs the worst, suffering from severe overfitting: while the training loss continuously decreases, the validation loss fluctuates significantly, resulting in the highest test error (MSE = 1.6219), which indicates unsuitability for this task.

From the perspective of error metrics, the superior performance of ARIMA in this experiment mainly reflects the pronounced linear trends and periodic structures in gold prices over the short to medium term (within 96 steps), as well as the careful selection of exogenous variables. These factors satisfy ARIMA's requirements for data stationarity and interpretability, underscoring its advantage as a traditional linear model for gold price forecasting under appropriate data conditions.

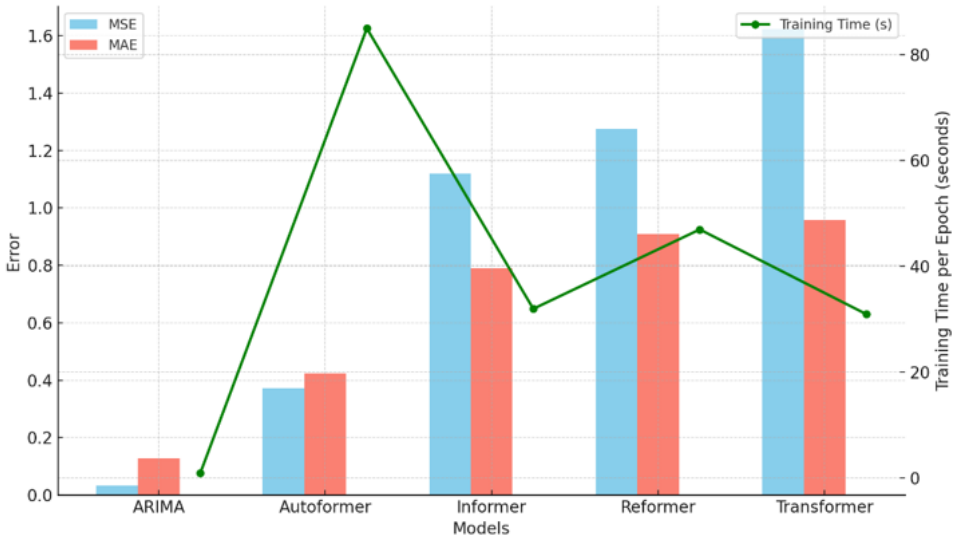

Figure 2. Model Comparison

4.5 The Discussion of Factors

To investigate the relative importance of external variables in gold price forecasting, the feature coefficients produced by the SARIMAX regression model are presented in Figure 4. These coefficients are automatically learned during model training; their signs indicate the direction of association between each variable and gold price fluctuations, while their absolute magnitudes reflect the relative importance of the variables. The results reveal that gold prices are most strongly influenced by international oil prices (Oil Price) and their valuation in Indian rupees (Oil Price INR), with coefficients of +5.015 and −5.331, respectively. This suggests that changes in oil prices affect gold through two primary channels: inflation expectations and input costs. The USD/INR exchange rate exhibits a strong negative correlation with gold prices (coefficient = −4.916), indicating that domestic currency depreciation tends to coincide with declines in gold prices, possibly due to the suppressive effect of rising gold import costs [47]. The interest rate variable shows a coefficient of −1.554, consistent with the classical financial theory that rising interest rates reduce the attractiveness of precious metal investments. Stock index–related variables, such as the S&P and S&P INR, have weaker impacts, with coefficients of −0.137 and +0.142, respectively, implying a modest safe-haven relationship between financial markets and gold. Other variables, such as SENSEX, Nifty, and Gold Demand, display small coefficient values, suggesting negligible influence. The autoregressive and moving average parameters of the model (ar. L1 = +0.72 and ma. L1 = −0.75) primarily capture autocorrelation in the error term and have no direct economic interpretation regarding external factors.

Overall, the model outputs indicate that gold price fluctuations are primarily driven by macroeconomic variables—particularly oil prices, exchange rates, and interest rates [48]—while conventional financial market indicators and short-term demand variables exert limited influence. These findings provide structural guidance for feature selection and importance attribution in subsequent deep learning models [49].

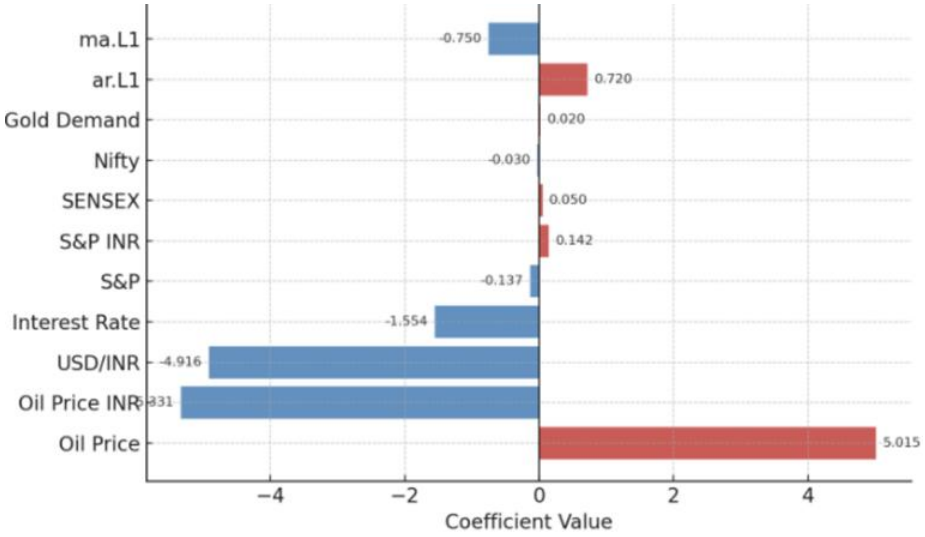

Figure 3. Feature Coefficients from ARIMA Model

5. Conclusion

In this paper, a systematic comparison is conducted between traditional statistical models and Transformer-based deep learning architectures, for the task of gold-price forecasting, including Informer, Autoformer, and Reformer. On the gold-price task, ARIMA, which yields the lowest test error (MSE = 0.0329; MAE = 0.1272), is found to achieve the best overall performance. Whereas Autoformer performs best among deep-learning approaches. Further variable regression analysis indicates that gold prices are significantly influenced by macroeconomic drivers such as oil prices, exchange rates, and interest rates, whereas index-based financial factors exert weaker effects. The evidence indicates that, in conditions where short-and-medium term linear structure prevails, classical ARIMA retains advantages in efficiency, robustness, and interpretability. It is thus well suited to resource-constrained and interpretability-critical financial forecasting.

References

[1] Qian Y, Ralescu D A, Zhang B. The analysis of factors affecting global gold price[J]. Resources Policy, 2019, 64: 101478.

[2] Toraman C, Basarir C, Bayramoglu M F. Determination of factors affecting the price of gold: A study of MGARCH model[J]. Business and economics research journal, 2011, 2(4): 37-50.

[3] Shumway R H, Stoffer D S. ARIMA models[M]//Time series analysis and its applications: with R examples. Cham: Springer International Publishing, 2017: 75-163.

[4] Han K, Wang Y, Chen H, et al. A survey on vision transformer[J]. IEEE transactions on pattern analysis and machine intelligence, 2022, 45(1): 87-110.

[5] Zhou H, Zhang S, Peng J, et al. Informer: Beyond efficient transformer for long sequence time-series forecasting[C]//Proceedings of the AAAI conference on artificial intelligence. 2021, 35(12): 11106-11115.

[6] Wu H, Xu J, Wang J, et al. Autoformer: Decomposition transformers with auto-correlation for long-term series forecasting[J]. Advances in neural information processing systems, 2021, 34: 22419-22430.

[7] Kitaev N, Kaiser Ł, Levskaya A. Reformer: The efficient transformer[J]. arXiv preprint arXiv:2001.04451, 2020.

[8] Sims C A. Macroeconomics and reality[J]. Econometrica, 1980, 48(1): 1–48.

[9] Bollerslev T. Generalized autoregressive conditional heteroskedasticity[J]. Journal of econometrics, 1986, 31(3): 307-327.

[10] Box G E P, Jenkins G M, Reinsel G C, Ljung G M. Time series analysis: forecasting and control[M]. 5th ed. John Wiley & Sons, 2015. 11

[11] Bollerslev T. Generalized autoregressive conditional heteroskedasticity[J]. Journal of Econometrics, 1986, 31(3): 307–327.

[12] Mandelbrot B. The variation of certain speculative prices[J]. The Journal of Business, 1963, 36(4): 394–419.

[13] Cont R. Empirical properties of asset returns: stylized facts and statistical issues[J]. Quantitative Finance, 2001, 1(2): 223–236.

[14] Taleb N N. The Black Swan: The Impact of the Highly Improbable[M]. Random House, 2007.

[15] Kristjanpoller W, Minutolo M C. Gold price volatility: A forecasting approach using the Artificial Neural Network–GARCH model[J]. Expert Systems with Applications, 2015, 42(20): 7245–7251.

[16] Arouri M E H, Lahiani A, Nguyen D K. Forecasting the conditional volatility of oil spot and futures prices with structural breaks and long memory models[J]. Energy Economics, 2012, 34(1): 283–293.

[17] Rumelhart D E, Hinton G E, Williams R J. Learning representations by back-propagating errors[J]. Nature, 1986, 323(6088): 533–536.

[18] Hochreiter S, Schmidhuber J. Long short-term memory[J]. Neural Computation, 1997, 9(8): 1735–1780.

[19] Bengio Y, Simard P, Frasconi P. Learning long-term dependencies with gradient descent is difficult[J]. IEEE Transactions on Neural Networks, 1994, 5(2): 157–166.

[20] Siami-Namini S, Tavakoli N, Namin A S. A comparative analysis of forecasting financial time series using ARIMA, LSTM, and BiLSTM[EB/OL]. arXiv:1811.11940, 2018.

[21] Sutskever I, Vinyals O, Le Q V. Sequence to sequence learning with neural networks[C]//Advances in Neural Information Processing Systems. 2014.

[22] Bahdanau D, Cho K, Bengio Y. Neural machine translation by jointly learning to align and translate[EB/OL]. arXiv:1409.0473, 2014.

[23] Vaswani A, Shazeer N, Parmar N, et al. Attention is all you need[C]//Advances in Neural Information Processing Systems. 2017.

[24] Fischer T, Krauss C. Deep learning with long short-term memory networks for financial market predictions[J]. European Journal of Operational Research, 2018, 270(2): 654–669.

[25] Bao W, Yue J, Rao Y. A deep learning framework for financial time series using stacked autoencoders and long-short term memory[J]. PLoS One, 2017, 12(7): e0180944.

[26] Livieris I E, Pintelas E, Pintelas P. A CNN–LSTM model for gold price time-series forecasting[J]. Neural Computing and Applications, 2020, 32(23): 17351–17360.

[27] Devlin J, Chang M W, Lee K, Toutanova K. BERT: Pre-training of deep bidirectional transformers for language understanding[EB/OL]. arXiv:1810.04805, 2018.

[28] Li S, Jin X, Xuan Y, Zhou X, Chen W, Wang Y X, Yan X. Enhancing the locality and breaking the memory bottleneck of transformer on time series forecasting[C]//Advances in Neural Information Processing Systems. 2019.

[29] Zhou H, Zhang S, Peng J, Zhang S, Li J, Xiong H, Zhang W. Informer: Beyond efficient transformer for long sequence time-series forecasting[C]//Proceedings of the AAAI Conference on Artificial Intelligence, 2021, 35(12): 11106–11115.

[30] Lim B, Arık S Ö, Loeff N, Pfister T. Temporal fusion transformers for interpretable multi-horizon time series forecasting[J]. International Journal of Forecasting, 2021, 37(4): 1748–1764.

[31] Ding Q, Wu S, Sun H, Guo J, Guo J. Hierarchical transformer with contrastive learning for long-term time series forecasting[J]. Expert Systems with Applications, 2023, 215: 119343.

[32] Xiong R, Yang Y, He D, Zheng K, Zheng S, Xing C, et al. On layer normalization in the transformer architecture[C]//International Conference on Machine Learning. 2023.

[33] Hamilton J D. Time series analysis[M]. Princeton university press, 2020.

[34] Chatfield C, Xing H. The analysis of time series: an introduction with R[M]. Chapman and hall/CRC, 2019.

[35] Box G E P, Jenkins G M, Reinsel G C, et al. Time series analysis: forecasting and control[M]. John Wiley & Sons, 2015.

[36] Hirschberg J, Manning C D. Advances in natural language processing[J]. Science, 2015, 349(6245): 261-266.

[37] Elman J L. Finding structure in time[J]. Cognitive science, 1990, 14(2): 179-211.

[38] LeCun Y, Bottou L, Bengio Y, et al. Gradient-based learning applied to document recognition[J]. Proceedings of the IEEE, 2002, 86(11): 2278-2324.

[39] Vaswani A, Shazeer N, Parmar N, et al. Attention is all you need[J]. Advances in neural information processing systems, 2017, 30.

[40] Alammar J. The illustrated transformer[J]. The Illustrated Transformer–Jay Alammar–Visualizing Machine Learning One Concept at a Time, 2018, 27(1).

[41] Zhou H, Zhang S, Peng J, et al. Informer: Beyond efficient transformer for long sequence time-series forecasting[C]//Proceedings of the AAAI conference on artificial intelligence. 2021, 35(12): 11106-11115.

[42] Chatfield C, Xing H. The analysis of time series: an introduction with R[M]. Chapman and hall/CRC, 2019.

[43] Indyk P, Motwani R. Approximate nearest neighbors: towards removing the curse of dimensionality[C]//Proceedings of the thirtieth annual ACM symposium on Theory of computing. 1998: 604-613.

[44] Gomez J, Barnett M A, Natu V, et al. Microstructural proliferation in human cortex is coupled with the development of face processing[J]. Science, 2017, 355(6320): 68-71.

[45] beekiran00. Gold-Price-prediction-Excel-and-R[CP/OL]. GitHub, 2022-01-23 [2025-09-11]. https://github.com/beekiran00/Gold-Price-prediction-Excel-and-R.

[46] Yanke Huang X Z. Analyzing Trends in Trading Patterns in Financial Markets Using Deep Learning Algorithms [J]. Journal of Electrical Systems, 2024, 20(3s): 1542–1555.

[47] Sahoo M, Nayak P P, Hanhaga M, et al. Exploring the asymmetric effect of remittance inflows on gold import demand: Evidence from a large gold-consuming and remittance-receiving Country [J]. Resources Policy, 2023, 85: 103900.

[48] M Dr K K, Palaniappan N, Sultana R, et al. The Impact of Macroeconomic Factors on Gold Prices [J]. INTERANTIONAL JOURNAL OF SCIENTIFIC RESEARCH IN ENGINEERING AND MANAGEMENT, 2024, 08(12): 1–5.

[49] Cheng X. A Comprehensive Study of Feature Selection Techniques in Machine Learning Models [J]. Insights in Computer, Signals and Systems, 2024, 1(1): 65–78.