

Dual-Task Financial Forecasting with HSAT: Integrating Market Trends and Closing Price Prediction

Tareq Fatihi Mustafa

Computer Department, Faculty of Science, Soran University, Erbil, Kurdistan Region, Iraq

Email: 402238182008@soran.edu.iq

Asst. Prof. Muhammadamin Muhammadali Daneshwar

Computer Department, Faculty of Science, Soran University, Erbil, Kurdistan Region, Iraq

Email: Muhhammad.amin@soran.edu.iq

ARTICLE INFO

Article History:

Received: 20/6/2025

Accepted: 7/8/2025

Published: Autumn2026

Keywords: *Stock prediction, transformer, financial time series, technical indicators, trend classification, price regression, patch-based attention, deep learning.*

Doi:

10.25212/lfu.qzj.11.4.49

ABSTRACT

Precise stock prediction is very important in risk management and investment optimization under volatile financial markets. Recent advances in deep learning provide new avenues for modeling such time series data. This paper introduces HSAT (Hybrid Stock Attention Transformer), a unified transformer-based architecture designed for two core financial forecasting tasks: trend direction classification (up/down) and closing price regression. Unlike traditional models that rely solely on raw historical price data, HSAT integrates a comprehensive set of engineered technical indicators—including Relative Strength Index (RSI), Moving Average (MA), Moving Average Convergence Divergence (MACD), momentum, daily price differences (Close_diff), and volume—to enrich the input representation. The model uses a patch-based temporal encoding system that lets it quickly find both local and global patterns in noisy, changing financial time series. HSAT was tested on a dataset of 25 high-volume stocks from five major international markets over a period of ten years. The architecture was independently trained and optimized for each forecasting task. Results show that HSAT achieves strong and robust performance: for trend classification, it consistently outperforms both traditional

machine learning models and state-of-the-art transformer variants, reaching an average F1 score of 0.70 and maintaining reliable prediction accuracy across all tested stocks, without exhibiting class bias. HSAT shows competitive error metrics (RMSE, MAE) for price regression compared to baseline models. It makes stable predictions even when the market is volatile. The HSAT architecture is modular and flexible, which makes it good for future improvements in real-world financial applications, such as integrating data from multiple sources and modelling trends.

1. Introduction

Economic modelling, algorithmic trading, and financial research all continue to place a lot of emphasis on stock market forecasting because of its significant effects on portfolio management, investment choices, and systemic risk reduction. Predicting a stock's direction (i.e., whether it will move upward or downward) or closing price is essential for both short-term and long-term trading strategies. As financial markets become more volatile and globally interconnected, stock prediction becomes increasingly complex due to the highly non-stationary, noisy, and multi-dimensional nature of financial time series data. Traditional forecasting methods like ARIMA, SVM, and decision trees have had limited success modelling such data because of their assumptions of linearity and stationarity, as well as their inability to detect latent temporal patterns and long-term dependencies. However, techniques for deep learning and machine learning have made great strides in handling large-scale, nonlinear, high-dimensional datasets. Recurrent neural networks (RNNs) and their variants, such as Long Short-Term Memory (LSTM) networks, have been widely used for sequential modelling in financial contexts. These models have shown a degree of success in modelling short- and mid-range dependencies, yet they suffer from fundamental issues such as vanishing gradients, poor scalability, and limited parallelism. The emergence of transformer-

based architectures, originally developed for natural language processing, has offered a paradigm shift in time series modelling by eliminating recurrence and instead utilizing self-attention mechanisms. Due to their computational advantage in dealing with long-range dependencies, transformers stand ready to be applied to time series forecasting applications. Beginning with the base transformer model introduced by Vaswani et al. several specialized models have since been designed to deal with time-dependent data-specific issues [1]. the Informer model introduced ProbSparse attention mechanism for efficient long-sequence forecasting [2] ; Auto former applied a novel decomposition block and auto-correlation mechanism [3]; The FEDformer model extended frequency domain modelling to improve long-range forecasting in time series data [4]; and Patch TST adopted patch-based inputs for improved local-global feature extraction [5]. These models have proven to perform at the cutting edge of a number of forecasting benchmarks, especially those that are marked by seasonality, periodicity, or structured dependencies. Though these transformer models have been effective, many major challenges still remain. First, most current methods only address one of the two tasks—trend classification or price regression—without offering a comprehensive framework for managing both. Second, a lot of models use raw stock prices as their sole input, frequently ignoring financial technical indicators that traders and analysts have long relied on to identify underlying patterns in price behavior, such as momentum, moving average (MA), relative strength index (RSI), and MACD. Studies such as StockFormer [6] have attempted to incorporate market structure and inter-stock relationships but still lack fine-grained financial feature engineering and flexibility across tasks. In parallel, traditional machine learning models—such as Random Forest, Gradient Boosting, and Support Vector Machines—have continued to play a role in stock forecasting, especially when combined with feature engineering and ensemble methods [7]. These models, therefore, frequently have trouble generalizing across various markets and time periods and cannot fit deep hierarchical structures found in temporal data. We present a new architecture called HSAT (Hybrid Stock Attention Transformer) to overcome these constraints: a flexible transformer-based model created to

forecast either the trend direction or the closing price of stocks, using the same structural backbone but distinct training pipelines. HSAT builds a rich feature space by combining several technical indicators with normalized past prices. The model can catch both short-term fluctuations and long-term dependencies in the input sequence by using a patch-based attention mechanism motivated by PatchTST. For trend classification tasks, HSAT incorporates Focal Loss to mitigate the effects of class imbalance, which is common in directional forecasting. For regression tasks, it utilizes a standard Mean Squared Error (MSE) loss while maintaining the same attention-based architecture. In our empirical evaluation, HSAT is tested on five major global stock markets—NASDAQ, NYSE, Germany, the UK, and Japan—using ten years of historical data from twenty-five high-volume stocks. While the model performs competitively in many scenarios, particularly in stocks with consistent trends and sufficient training data, it also shows limitations in handling volatile or sparsely represented stocks. These findings highlight both the potential and the boundaries of HSAT's generalization capabilities. Importantly, this evaluation is conducted exclusively using numerical time-series data and engineered technical indicators, without incorporating textual or sentiment-based sources such as financial news or investor sentiment scores. This design choice allows for a focused analysis of the attention mechanism and financial feature engineering, though it also opens pathways for future improvements through multimodal data integration.

In summary, this work contributes a novel hybrid transformer model for stock forecasting, bridging the gap between deep learning architecture design and financial domain knowledge. HSAT is an interpretable, efficient, and extendable framework that can be applied across diverse forecasting tasks and financial contexts.

2.Related Works

Financial time series prediction, especially of stock prices, has a proven track record using both traditional statistical approaches from the past as well as newer machine learning methodologies[7]. Traditional approaches like ARIMA and

exponential smoothing presented foundational methodologies in time series prediction; however, they tended to struggle due to the nonlinearity and non-stationarity nature inherent in actual financial data[8]. Traditional machine learning algorithms—like decision trees, random forests, gradient boosting, and support vector machines—are improved by including more complex patterns and refined feature engineering but are still bound by the stationarity assumption and tend to not generalize in the case of turbulent or fragmented markets [7]. The shift towards deep learning brought about the utilization of RNN and LSTM models, which better modeled sequential dependence and temporal relations within financial data [9, 10]. While these architectures improved performance, they were beset by problems with scalability as well as not being able to model very long-range dependencies, especially in high-frequency trading systems [11]. The advent of transformer models—initially intended for natural language processing—marked a paradigmatic change in time series modeling by eliminating recurrence and adopting self-attention mechanisms instead [1]. The method allowed models to model local and global dependencies efficiently with enhanced efficiency and parallelization. In the wake, several sophisticated transformer models were put forward targeting time series and financial data. The Informer model introduced ProbSparse attention as a solution for the efficient modeling of very long sequences [2]. The Autoformer model incorporated seasonal-trend decomposition and auto-correlation mechanisms to enhance long-term forecasting precision [3]. The FEDformer proposed frequency domain modeling to better handle periodic elements in datasets [4]. PatchTST used patch-based temporal encoding to better extract local and global features, with good results on volatile stock datasets [5]. One such key development in the field is the Temporal Fusion Transformer (TFT) which fuses recurrent and attention layers with static covariate encoders and gate mechanisms for enhancing both effectiveness and interpretability for multi-horizon forecasting [12]. Another development to counter non-stationarity has been developed and example non-stationary transformers which include normalization and adaptive attention techniques [13]. Hybrid models that fuse CNNs with transformers have also been created to capture spatial-temporal

features and improve precision in complex financial environments [14]. Explainability has become a critical part of modern forecasting models, especially in the financial domain, where interpretability builds trust and complies with regulatory regulations. Techniques like LIME and SHAP have become popular for explaining predictions of black-box models, providing insight into feature importance at global and local levels [15] [16]. State-of-the-art attention models like TFT also offer inherent interpretability in terms of the discovery of variable importance and temporal relevance at prediction time [12]. In-depth reviews and surveys highlight the increasing emphasis on explainable AI in financial and time series modeling [17]. Incorporating domain knowledge, e.g., technical indicators (RSI, MACD, moving averages, momentum), has been demonstrated to greatly improve forecasting performance in terms of capturing trend strength, reversals, and market sentiment over raw prices alone [18]. More recent transformer models, e.g., StockFormer and market-guided architectures, achieve even higher accuracy through modeling inter-stock relations and integrating multi-source information [19]. Despite these advances, the majority of state-of-the-art models are constructed to carry out either classification (trend direction) or regression (closing price) tasks, but more frequently than not within one architecture [9]. Class imbalance and volatility are also persistent problems—frequently at best incompletely addressed by modern loss functions and data treatment techniques [15].

Recent work extends the boundary of transformer models for financial time series by incorporating hybrid attention mechanisms, non-stationary adaptations, and fusion of technical and external features. Yet a key practical shortfall persists with respect to models delivering flexible, interpretable, and robust forecasting capability on both trend classification and price regression—thus the motivation to develop architectures like HSAT that aim to meet these requirements in one and the same flexible framework.

Subsequent improvements have concentrated on augmenting forecasting models with more profound structural innovation and interpretability. A hybrid model that combines frequency decomposition (CEEMD), GRU, and Transformer layers has

been proposed [20]. This architecture is particularly successful at overcoming noise and extracting subtle temporal characteristics in high-frequency limit order book data, surpassing previous hybrid baselines. A channel-independent spatio-temporal hypergraph attention network has been developed, incorporating self-supervised transformer pretraining and relational hypergraph construction for robust multi-stock ranking, which demonstrated higher investment returns on NASDAQ and NYSE indices [21]. An explainable deep learning model with BiLSTM, attention, and SHAP/LIME explanations was introduced to predict five different stock movement classes, emphasizing the significance of trust in financial AI systems [22]. Hybrid modeling has also been enhanced with the fusion of decomposed external trend signals and internal LSTM predictions to improve robustness across several markets, especially under volatile dynamics [23]. Finally, reversal prediction has been explored through feature-enriched neural networks, stressing the need for capturing market turning points via advanced temporal and indicator-based embeddings[24]. The five papers together manifest the increasing tendency toward hybrid, interpretable, and multi-task architectures for real-world financial prediction.

Methodology

1.1 Problem Formulation

In this paper, two core forecasting tasks commonly encountered in financial time series analysis are addressed: trend classification and price regression. Both tasks operate over historical stock data collected at daily intervals and involve modelling patterns across temporal sequences to make informed predictions.

1.2 Trend Classification

Trend classification refers to the task of determining whether it increases or decreases of a stock's price movement—either upward or downward—based on a fixed-length sequence of historical data. Unlike regression models that aim to forecast exact price values, trend classification focuses solely on the directional movement, which is particularly valuable in algorithmic trading and decision-

making systems.

To clarify the model structure, The input and output components are now described in precise terms:

Let w be the window size, i.e., the number of historical time steps used for prediction.

Let d be the number of features at each time step (e.g., Open, High, Low, Close, Volume, or technical indicators).

Let $X_t \in R^{w \times d}$ be the feature vector at time step t .

R is the set of real numbers.

Let the input sequence $X_t \in R^{w \times d}$ be defined as:

$$X_t = [x_t - w + 1, x_t - w + 2, \dots, x_t] \Rightarrow (1)$$

A sliding window of w earlier time steps that ends at time t is represented by the vector x_t . The price change between the current and next closing prices determines the corresponding binary trend label, $y_{t+1} \in \{0,1\}$:

$$y_{t+1} = \begin{cases} 1 & \text{if } close_{t+1} > close_t \text{ (Up)} \\ 0 & \text{otherwise (Down)} \end{cases} \Rightarrow (2)$$

Learning a classification function is the aim:

$$f_{cls}: R^{w \times d} \rightarrow \{0,1\} \Rightarrow (3)$$

so that the model forecasts:

$$\hat{y}_{t+1} = f_{cls}(x_t) \Rightarrow (4)$$

Here, the input window x_t is used to predict the trend direction for the next time step.

This formulation is particularly useful in short-term trading strategies because it is easier to take action when determining whether a stock is likely to increase or decrease in value than when predicting the exact future price. However, because stock prices are more likely to move upward or sideways, class imbalance is a common issue in this task. Advanced loss functions like Weighted Cross-Entropy or Focal Loss are frequently used to address this and direct the learning process toward greater sensitivity for minority classes. Because deep learning models, particularly Transformer-based architectures, use the trend classification framework to transform the raw time-series data into labeled sequences of trends,

they can extract temporal patterns and provide informed directional predictions in real-world stock trading environments.

1.3 Price Regression

Examining past stock performance and relevant traits helps one to forecast a stock's future closing price, a method known as price regression. Unlike trend classification, which produces a binary output indicating price direction, regression models aim to generate a continuous numerical value representing the expected stock price at a specified future time.

Formally, let w be the window size, which shows how many prior time steps served as input.

Let d represent how many features there are at each time step.

The feature vector at time step is $x_t \in R^d$

The input window is defined as:

$$X_t = [x_{t-w+1}, x_{t-w+2}, \dots, x_t] \in R^{d \times w} \Rightarrow (5)$$

A regression function is taught to the model:

$$f_{reg}: R^{d \times w} \rightarrow R \Rightarrow (6)$$

so that the following yields the anticipated future price:

$$\hat{p}_{t+h} = f_{reg}(X_t) \Rightarrow (7)$$

In this case, h is the forecast horizon, which indicates how many time steps ahead the model is attempting to predict, and p_{t+h} stands for the anticipated closing price at time $t + h$. For example, if t refers to the current day and $h = 1$, the model predicts the price for the next day. If $h = 5$, it predicts the price five days into the future.

Similar to the classification task, price regression is applied using fixed-length sliding windows. The model can show how short-term historical trends are connected to next price changes by matching each input segment (X_t) with the

suitable future value (p_{t+h}). This method is quite relevant in finance-related fields like trading automation, portfolio risk analysis, and strategic investment planning since it fits both directional forecasting and continuous-value prediction. It allows more in-depth understanding of future Market behavior.

1.4 Feature Engineering

A key differentiator of this study is its use of financial technical indicators as part of the input feature space, rather than relying solely on raw prices. Technical indicators are widely used in financial analysis to capture complex dynamics such as momentum, volatility, and trend reversals

1.4.1 RSI (Relative Strength Index)

The RSI is a momentum oscillator that measures the speed and change of price movements over a defined period (typically 14 days). It ranges between 0 and 100 and is commonly used to identify overbought (>70) or oversold (<30) conditions in the market. Incorporating RSI allows the model to account for internal trend strength, improving trend prediction accuracy [25].

$$RSI = 100 - \frac{100}{1+RS} \Rightarrow (8) [26]$$

Where:

$$RS = \frac{\text{Average Gain over } n \text{ periods}}{\text{Average Loss over } n \text{ periods}}$$

With n Commonly set to 14.

1.4.2 MA (Moving Average)

The Simple Moving Average (MA) smooths out price fluctuations by computing the average of closing prices over a fixed window. It helps capture general direction and dampens market noise, aiding in both regression and classification settings. MAs are fundamental components in crossover strategies used by traders.

$$MA_t = \frac{1}{n} \sum_{i=0}^{n-1} P_{t-i} \Rightarrow (9) [27]$$

Where P_{t-i} is the closing price at time $t - i$ and n is the window length.

1.4.3 MACD (Moving Average Convergence Divergence)

The MACD is a trend-following momentum indicator that shows the relationship between two moving averages (typically 12-period and 26-period exponential moving averages). The difference between these values generates the MACD line, which is often paired with a signal line to detect bullish or bearish crossovers. This indicator is particularly valuable for capturing trend reversals, enhancing model responsiveness to market shifts.

$$MACD_t = MEA_{12}(P_t) - EMA_{26}(P_t) \Rightarrow (10) [28]$$

Where $MEA_{12}(P_t)$ is the exponential moving average of the closing price over n periods, defined recursively as:

$$EMA_n(P_t) = \alpha P_t + (1 - \alpha)EMA_n(P_{t-1}), \alpha = \frac{2}{n+1} \Rightarrow (11)$$

1.4.4 Momentum

Momentum is calculated as the difference between the current closing price and that of a prior time step (e.g., 10 days ago). It reflects the velocity of price movement, helping the model detect surges or declines in price behavior.

$$Momentum_t = P_t - P_{t-k} \Rightarrow (12) [29]$$

Where k is the momentum window (e.g., 10).

After price data normalization, every stock individually is calculated using all these indicators. Then, stacked with raw price and volume characteristics, they create an augmented input vector for each time step. The last input dimension in classification is 7 features per time step, while in regression it is 6 features. It comprises Close, Volume [30], Close_diff, RSI, MA, MACD, and Momentum. Incorporating these engineered features allows the model to capture financial intuition inside a deep learning architecture—a capacity many earlier Transformer models miss[5] .

1.5 HSAT Model Architecture

The Hybrid Stock Attention Transformer (HSAT), on the basis of the transformer-based architecture, learns complex temporal relationships and financial phenomena in stock market time series data. Its modular design has three primary stages: patch embedding, transformer encoding, and a task-specific prediction head. Although each is trained and validated independently with its own output head and loss function, the same architecture is applied for both classification and regression.

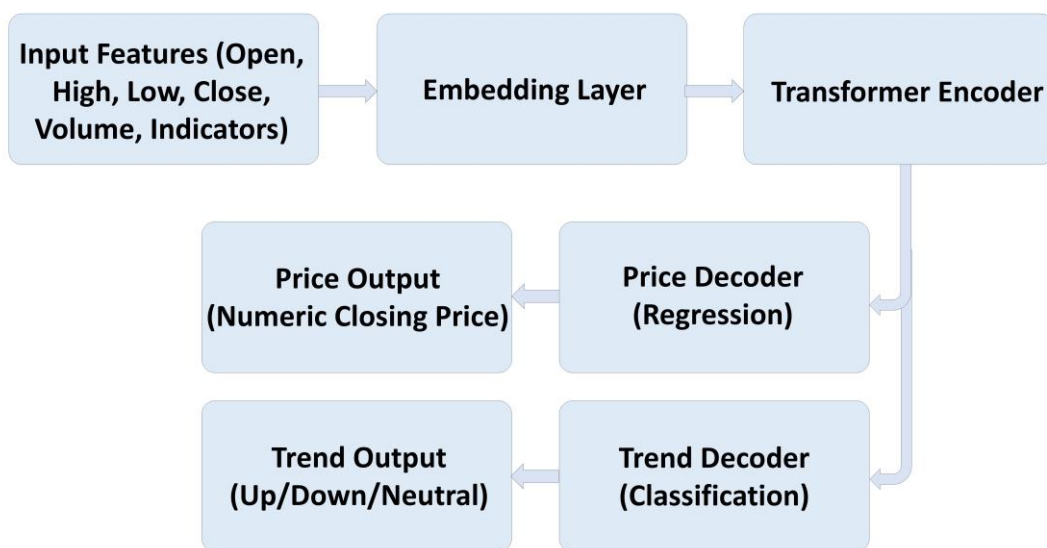


Figure (1): Overview of the HSAT dual-mode architecture

Figure (1) illustrates the big picture of the proposed HSAT framework, which operates in two distinct modes: trend classification and price regression. Although both models follow the identical architectural pipeline, including input processing, embedding, transformer-based encoder, and task-specific output layers, each task is implemented and trained separately to attain maximum specialization. The input consists of historical stock data and technical indicators, which are first embedded

into fixed-size vectors. These are then processed through a shared encoder module. Depending on the selected forecasting objective, the model routes the encoded features to either a classification head (to predict trend direction) or a regression head (to predict numerical closing price). This design allows the framework to handle dual tasks effectively while maintaining architectural modularity.

Additional architectural and implementation details of all the modules in the HSAT framework are described in the subsequent subsections of Section 3.5.

1.5.1 Input Windowing

Historical data for each stock is split into fixed-size sliding windows of length w (e.g., 10 time steps). Input for every time step is various characteristics, including raw prices and engineered technical indicators (RSI, MACD, etc.). The input tensor thus forms: (B, F, T)

Location:

Where B = Size of batch

Window size is T (e.g., 10).

Features per time step, F , e.g., 7 for classification, 6 for regression.

These windows let the model acquire temporal patterns from price-related feature sequences.

1.5.2 Layer of Patch Embedding

The input time series is first divided into non-overlapping patches of fixed length to facilitate the extraction of local temporal features. Let the windowed input sequence be represented as $X \in R^{T \times F}$ where T is the window length and F is the number of features per time step. The sequence is partitioned into $N_p = T/P$ patches, each of length P , such that each patch contains $P \times F$ values.

Each patch is flattened to form a vector $X_p \in R^{P \cdot F}$, which is then projected into a d_{model} -dimensional latent space through a learnable linear transformation:

$$e_p = X_p W_e + b_e \quad \Rightarrow (13)$$

Where:

- X_p is the flattened patch vector for patch P ,
- $W_e \in R^{(P.F) \times d_{model}}$ is the patch embedding weight matrix,
- $b_e \in R^{d_{model}}$ is a bias vector,
- $e_p \in R^{d_{model}}$ is the resulting embedded representation for patch p .

Patch embedding converts the segmented time series into a series of fixed-length vectors so that the transformer encoder can conveniently learn both the local and global dependencies of the data.

1.5.3 Transformer Encoder

The embedded sequence of input patches is processed by a conventional transformer encoder to capture both local and global temporal dependencies in the financial time series. The encoder consists of LLL stacked layers, with each layer comprising two main subcomponents: multi-head self-attention and a position-wise feed-forward network. Each subcomponent is preceded by residual connections and layer normalization to stabilize training and improve generalization.

Multi-Head Self-Attention

Given a sequence of patch embeddings $Z = [z_1, z_2, \dots, z_n]$, the input is projected into queries (Q) keys (K) and values (V) by learned linear transformations:

$$Q = ZW^Q, K = ZW^K, V = ZW^V \quad \Rightarrow (14)$$

Where $W^Q, W^K, W^V \in R^{d_{model} \times d_k}$

For each attention head, the scaled dot-product attention is computed as:

$$Attention(Q, K, V) = softmax \left(\frac{QK^T}{\sqrt{d_k}} \right) V \quad \Rightarrow (15)$$

Outputs from all heads are concatenated and projected:

$$Multihead(Z) = Concat(head_1, \dots, head_h)W^O \quad \Rightarrow (16)$$

Add & Norm

A residual connection is applied, followed by layer normalization:

$$H_1 = \text{LayerNorm}(Z + \text{MultiHead}(Z)) \Rightarrow (17)$$

Position-wise Feed-Forward Network

A two-layer fully connected feed-forward network is applied independently to each position:

$$\text{FFN}(X) = \text{ReLU}(xW_1 + b_1)W_2 + b_2 \Rightarrow (18)$$

A second residual connection and normalization are applied:

$$H_2 = \text{LayerNorm}(H_1 + \text{FFN}(H_1)) \Rightarrow (19)$$

This process is repeated for L encoder layers. The output of the final encoder layer serves as the learned temporal representation for downstream prediction tasks.

Note:

No explicit positional encoding is introduced; instead, the natural patch order encodes temporal relationships, such as PatchTST [5].

1.5.4 Prediction Head

After going through the transformer encoder, the last output with respect to the last patch is represented by h_{last} , is the temporal representation of the input window aggregated. This vector is fed into a specialized multi-layer perceptron (MLP) head with a configuration that varies based on the nature of the downstream task: regression or classification.

Regression Head

For the regression exercise (predicting the next closing price), h_{last} is achieved by a sequence of densely connected layers that include nonlinear activation functions and dropout regularization techniques. The mathematical representation can be given as:

$$r_1 = \text{ReLU}(h_{last}W_1^r + b_1^r) \Rightarrow (20)$$

$$r'_1 = \text{Dropot}(r_1, p = 0.2) \Rightarrow (21)$$

$$r_2 = \text{ReLU}(r'_1 W_2^r + b_2^r) \Rightarrow (22)$$

$$r'_2 = \text{Dropout}(r_2, p = 0.2) \Rightarrow (23)$$

$$y' = r'_2 W_3^r + b_3^r \Rightarrow (24)$$

Where y' is the predicted closing price, and W_i^r, b_i^r denote the weights and biases of the regression MLP.

Classification Head

For the binary classification task (predicting trend direction: Up or Down), h_{last} is similarly passed through an MLP and projected onto a two-dimensional output space:

$$c_1 = \text{ReLU}(h_{last} W_1^c + b_1^c) \Rightarrow (25)$$

$$c'_1 = \text{Dropout}(c_1, p = 0.2) \Rightarrow (26)$$

$$c_2 = \text{ReLU}(c'_1 W_2^c + b_2^c) \Rightarrow (27)$$

$$c'_2 = \text{Dropout}(c_2, p = 0.2) \Rightarrow (28)$$

$$o = c'_2 W_3^c + b_3^c \Rightarrow (29)$$

Where $o \in R^2$ is the logit vector representing the unnormalized probabilities for the "Up" and "Down" classes. The predicted class is obtained by applying the SoftMax function:

$$p = \text{softmax}(o) \Rightarrow (30)$$

1.5.5 Rationale for Patch Design and Attention

Image-based architectures (like Vision Transformers) and time series models such as PatchTST motivate the decision to use patches instead of discrete time steps.

The model groups local time steps to:

1. Reduce sequence length (fewer steps – faster training)
2. Emphasize local patterns within patches
3. Let the transformer focus on inter-patch relationships

Furthermore, multi-head self-attention lets the model:

- Dynamically weight the relevance of various time intervals.
- Without recurrence, encode both short- and long-range dependencies.

Common in stock data, noise, abrupt changes, and volatility are handled more effectively by this design.

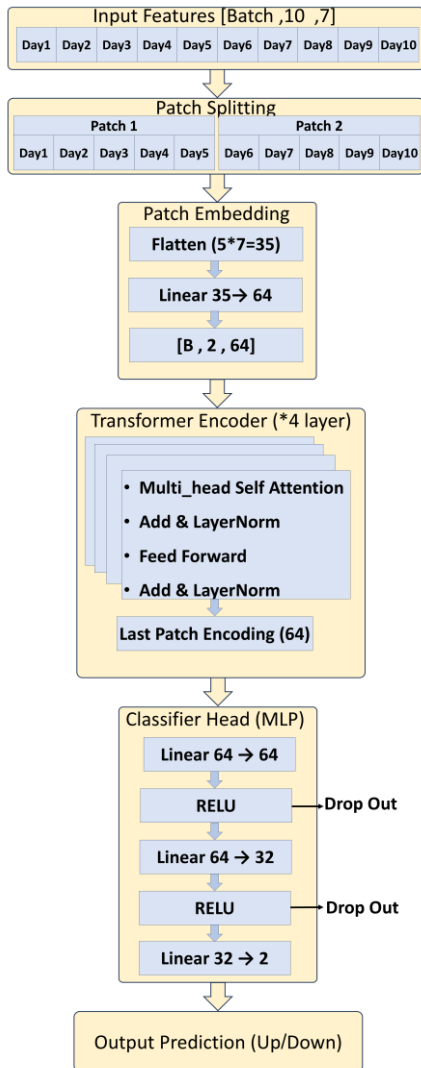


Figure (2): HSAT for Classification

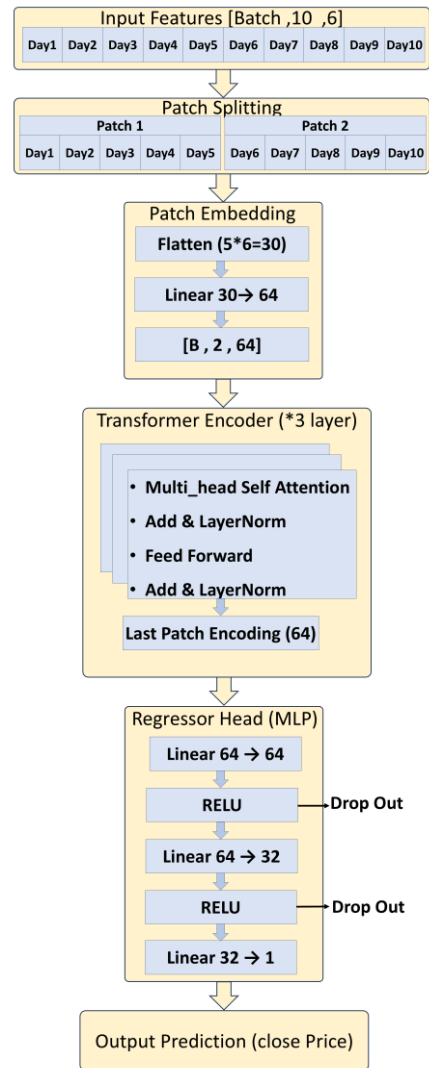


Figure (3): HSAT for Regression

Figure 2. Architecture of the proposed HSAT model for stock trend classification. The model processes time-series input features (7 per time step) using patch embedding and a multi-layer transformer encoder. The final output from the last patch is passed through an MLP head to predict the direction (up or down) of the

stock movement.

Figure 3. Architecture of the HSAT model for stock price regression. Similar to the classification version, the model utilizes patch-based embedding followed by a transformer encoder. The encoded representation of the final patch is processed by a regression head to predict the next closing price.

1.6 Loss Functions

Task-specific loss functions enable the HSAT model to be trained depending on whether classifying trends or predicting prices is wanted. One selects a particular loss function for each to match the particular needs and challenges of the specified activity.

1.6.1 Classifying: Difficulty Maintaining Concentration

Directional trend prediction (up/down) yields a two-dimensional logit vector with the classification head of HSAT. Rather than relying on conventional cross-entropy loss, Focal Loss is used to reduce the effect of class imbalance—a frequent problem in actual financial datasets where upward or downward movements could predominate.

Focal Loss emphasizes the training on more difficult, misclassified cases and dynamically down-weights simple ones. Its definition is:

where p_t is the predicted probability for the true class, γ is the focusing parameter (usually set to 2), and α_t is the weight assigned to class t . This formulation helps stabilize training and improves the model's sensitivity to the minority class [15].

Class weights are computed using sklearn's `compute_class_weight` to ensure balance among training samples.

1.6.2 Regression: Mean Squared Error (MSE)

For predicting the closing price, the regression head outputs a single value. the standard Mean Squared Error (MSE) as the target are used.

where y^i is the expected value and y_i is the real closing price. MSE is appropriate for financial forecasting since prediction accuracy has a direct expense; it punishes

bigger deviations more strongly.

1.7 Training Configurations

Using the Adam optimizer with a learning rate of 0.0003 and batch size of 64, both versions of HSAT—classification and regression—are trained separately.

For classification tasks, 100 epochs are used; for regression tasks, up to 200 epochs are run.

Main parameters applied throughout all experiments:

Window size (T): 10 time steps

Patch length: 5

d_model: 64

Number of attention heads: 4

Transformer encoder layers: 3 (regression) or 4 (classification)

Dropout: 0.2

Optimizer: Adam

Scheduler: None (constant learning rate)

Loss functions: MSE for regression, focal loss for classification

All models were trained on NVIDIA GPUs under Google Colab Pro. Every stock is trained individually on its own time series, therefore enabling the model to specialise for particular price behaviours.

1.8 Assessment Criteria

Model performance is evaluated by us using conventional evaluation criteria suitable for regression and classification activities.

Classification Criteria:

Proportion of right predicted directions—up/down.

True positives split by all forecasted positives defines precision.

True positives over all actual positives define recall.

The harmonic mean of recall and accuracy gives the F1 score

Confusion Matrix: Especially under imbalance, to see class-wise performance.

Averaged across markets, these measures are calculated per stock.

Standardised assessment uses Sklearn's classification_report and confusion_matrix.

1.8.1 Measuring Regression

Mean Absolute Error (MAE): Average absolute difference between predicted and actual prices is assessed using the.

Root Mean Squared Error (RMSE): More strongly punishes bigger mistakes.

R^2 Score: Shows the percentage of variance the model accounts for.

All measures are stated at the per-stock level over five stock exchanges.

To assess generalization and robustness, stock-level breakdown as well as average performance is examined.

Setup of Experiments

The datasets, preprocessing pipeline, and hyperparameter settings used to assess the HSAT model across five major worldwide stock markets are described in this part.

1.9 Data Sources

The HSAT model is assessed using historical data from five major stock markets:

- NASDAQ - NASDAQ Stock Market (United States)
- NYSE - New York Stock Exchange (United States)
- FTSE 100 - Financial Times Stock Exchanges 100 Index (United Kingdom)
- DAX - Deutsche Börse (Germany)
- Nikkei 225 - Tokyo Stock Price Index (Japan)

Based on data availability and trading activity, five high-volume stocks were chosen from each market. Roughly 2500 daily observations per stock over a 10-year period (2014 to 2024) span the dataset, which totals more than 62,000 labelled samples.

Every record contains the following fields:

Date, Stock, Open, High, Low, Close, Volume, Price_Change, and the directional label Direction.

1.10 Normalization and Preprocessing

RapidMiner Studio, a visual data science tool, preprocessed the raw stock data to apply min-max normalization to numerical characteristics (open, high, low, close, volume).

The normalized datasets were then used in model training inside Python and exported as CSV files.

The Python pipeline for each stock individually also produced extra technical indicators:

- Difference in closing price from the previous day.
- RSI: 14-day Relative Strength Index
- 10-day Simple Moving Average
- MACD: The difference between the 12-day and 26-day EMA
- Momentum: Difference between present close and close 10 days ago.

Forward-fill and back-fill techniques were used to handle missing values.

The target label Direction is specified as:

- Up: if Price_Change > 0; otherwise, Down.

1.11 Splitting of Datasets

The HSAT model is trained separately for each stock; every one of the chosen stocks has roughly 2500 daily observations.

Every stock's time series is split into two sections:

80% for training;

20% for testing

To mirror real-world forecasting circumstances—where future data is required to remain unseen during model training—this division is done chronologically.

To maintain temporal dependencies, no arbitrary shuffling is used.

Two reasons motivated the decision for an 80/20 split rather than the more usual 70/30:

1. The requirement to keep a sufficiently large training window to enable the transformer encoder to acquire significant long-range temporal patterns.

2. Evaluating performance without sacrificing training efficacy requires a statistically reliable test set (about 500 time steps per stock).

Particularly in the context of deep attention-based models, where sequence length and context size are critical for learning quality, this setup strikes a balance between model generalization and evaluation stability.

1.12 Hyperparameters

Unless otherwise specified, all models were trained using consistent hyperparameter settings to ensure fair comparison across experiments. The key training configurations and loss functions used in the HSAT evaluation are summarized in Table 1.

Table (1): Key hyperparameters, training configurations, and loss functions for HSAT model experiments

Parameter	Value
Window size	10
Patch size	5
Embedding size	64
Attention heads	4
Encoder layers	3 (regression), 4 (classification)
Dropout	0.2
Batch size	64
Epochs	200 (regression), 100 (classification)
Optimizer	Adam
Learning rate	0.0003
Loss function	MSE (regression), Focal Loss (classification)
Evaluation	Chronological holdout (80/20 split)
Parameter	Value

To further illustrate the data characteristics, Table 2 presents the distribution of real labels (up and down) for each stock market considered in this study.

Table (2): Real Label Distribution by Market

Market	Up	Down	Total Samples	Up:Down Ratio
NASDAQ	6419	6161	12,580	51.0% : 49.0%
NYSE	6402	6178	12,580	50.9% : 49.1%
UK	6445	6175	12,620	51.1% : 48.9%
Germany	6483	6212	12,695	51.1% : 48.9%
Japan	5633	6692	12,325	45.7% : 54.3%
Total	—	—	62,800	—

As shown in Table 2, the datasets exhibit relatively balanced label distributions in most markets, with the exception of Japan, which shows a slight bias toward the "Down" class.

The proposed workflow is presented in Figure 4.

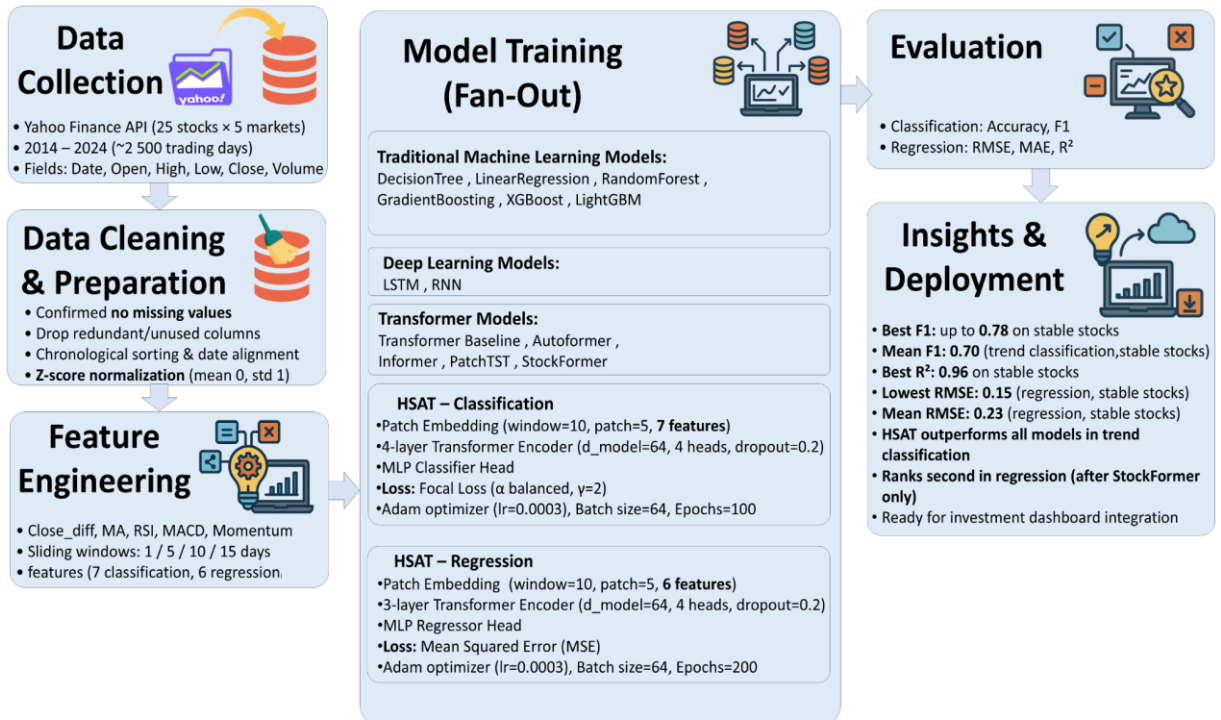


Figure (4): Workflow of the proposed HSAT methodology for stock trend and price prediction

Results and Discussion

In this chapter, the experimental results obtained from applying a wide range of models—both traditional machine learning techniques—are presented and analysed and transformer-based architectures—on stock market data collected from five global exchanges.

Many experiments were conducted, e.g., classification and regression on 25 stocks for each model. To improve clarity, only the best performance and overall averages per model are presented. This is more readable, highlights the main results, and provides valuable information without bombarding the reader with too much data. Additional performance metrics and full stock-by-stock evaluations are available in the appendix or supplementary files, where applicable.

1.13 Classification Results

1.13.1 Traditional Models

The classification performance of traditional machine learning models—including Decision Tree, Random Forest, Logistic Regression, Gradient Boosting, LightGBM, LSTM, and RNN—was evaluated across 25 stocks representing five global markets. Each model was trained and tested individually on every stock, and their performance was measured using accuracy and F1-score.

To maintain clarity, each model is presented with the following:

- Its best result based on individual stock performance.
- Its average performance across all stocks.

Table 3. Best accuracy and average accuracy of traditional classification models across 25 stocks. The table presents the highest accuracy achieved by each model on an individual stock, along with the overall average accuracy across all evaluated stocks.

**Table (3): Best accuracy and average accuracy of traditional classification models
across 25 stocks**

Model	Market	Stock	Accuracy	Precision	Recall	F1	TP	FP	FN	TN	MEAN
Logistic Regression	UK	ULVR.L	0.53	0.54	0.38	0.45	95	82	153	175	0.50
Decision Tree	NASDAQ	AAPL	0.53	0.56	0.31	0.40	78	61	176	189	0.49
Random Forest	Japan	9984.T	0.54	0.55	0.49	0.52	124	100	129	140	0.49
Gradient Boosting	NASDAQ	MSFT	0.53	0.53	0.62	0.57	158	141	96	109	0.50
XGBoost	Japan	9984.T	0.54	0.56	0.54	0.55	136	108	117	132	0.50
LightGBM	Japan	9984.T	0.54	0.55	0.51	0.53	130	105	123	135	0.49
RNN	Germany	ALV.DE	0.55	0.55	0.99	0.71	276	225	2	4	0.50
LSTM	Germany	ALV.DE	0.54	0.55	0.99	0.70	276	229	2	0	0.50

As seen in Table 3, traditional models had mixed classification effectiveness for different markets and equities. Although their overall average performance was comparatively low—at about 50%—algorithms Random Forest, LightGBM, and XGBoost achieved a maximum accuracy of nearly 54%. Especially in logistic regression, accuracy and recall were excellent; however, positive class recall was poor with few upward trends. In contrast, while RNN and LSTM had nearly perfect recall, they could not identify downward movements, as indicated by a true negative (TN) count of zero, i.e., very strong class bias towards predicting "Up" only.

These results draw attention to the difficulty of directional imbalance in stock trend classification using conventional approaches.

Table 4. Average F1 scores and best F1 scores of conventional classification models across chosen stocks. The table shows the highest F1 score for each model on a particular stock together with the related precision, recall, and confusion matrix values. The last column shows every model's average F1 score over all 25 equities.

Table (4):Summary of F1 scores for traditional classification models on selected equities

Model	Market	Stock	Accuracy	Precision	Recall	F1	TP	FP	FN	TN	F1_Mean
Logistic Regression	NYSE	KO	0.52	0.52	1.00	0.68	261	243	0	0	0.38
Decision Tree	NYSE	KO	0.50	0.51	0.63	0.57	165	157	96	86	0.45
Random Forest	Japan	6758.T	0.52	0.51	0.65	0.57	157	150	85	101	0.41
Gradient Boosting	NASDAQ	GOOGL	0.49	0.50	0.84	0.63	214	214	42	34	0.42
XGBoost	NYSE	KO	0.53	0.54	0.67	0.60	176	150	85	93	0.49
LightGBM	NYSE	KO	0.51	0.52	0.86	0.65	225	210	36	33	0.47
RNN	Germany	ALV.DE	0.55	0.55	0.99	0.71	276	225	2	4	0.44
LSTM	Germany	ALV.DE	0.54	0.55	0.99	0.70	276	229	2	0	0.45

F1-score studies, as indicated in Table 4, expose more subtleties in model behaviour than those revealed by accuracy by itself. Although models like Logistic Regression and RNN attained very high F1 values (up to 0.99) on individual stocks, a closer look at their confusion matrices reveals that this is mostly due to extreme class imbalance—most notably a complete failure to detect the "Down" class (TN = 0).

Though applied to volatile or unbalanced stocks, Random Forest and Decision Tree still battled class distinction, achieving more balanced F1 scores in the range of 0.63–0.65.

Though low TN, gradient boosting and LightGBM indicated bias towards detecting "up" trends by showing fairly high recall but poor negative predictions.

The F1-score study reveals that most conventional models tend to overfit to the majority class, therefore lacking generalisation in the presence of directional imbalance.

The classification performance of transformer-based models is assessed, including Transformer Baseline, Autoformer, Informer, PatchTST, StockFormer, and our proposed model HSAT, in this section. Each model was applied to the same 25-

stock dataset used in the traditional models' evaluation.

The best accuracy and F1 score are reported and achieved by each model on a specific stock, along with the corresponding precision, recall, and confusion matrix.

The final column provides the model's average F1 score across all stocks.

This table displays each model's peak result in terms of accuracy and F1 score on a selected stock, including detailed confusion matrix values and average F1 score across all (25) stocks.

Table (5): Best classification performance and average F1-score for transformer-based models.

Model	Market	Stock	Accuracy	Precision	Recall	F1	TP	TN	FP	FN	F1 Mean
Transformer Baseline	UK	BP.L	0.51	0.51	0.51	0.51	107	143	122	115	0.37
Autoformer	Germany	DTE.DE	0.56	0.57	1.00	0.72	288	0	218	0	0.37
Informer	NYSE	GE	0.46	0.48	0.49	0.43	58	167	52	217	0.34
PatchTST	Japan	6758.T	0.51	0.50	0.50	0.50	161	88	148	92	0.36
StockFormer	NASDAQ	GOOGL	0.54	0.53	0.53	0.51	201	69	165	64	0.40
HSAT	NYSE	GE	0.71	0.68	0.91	0.78	254	100	122	24	0.70

As seen in Table 5, HSAT achieved the highest performance with an accuracy of 0.71 and an F1 score of 0.78, showing both high precision and strong recall. It also demonstrated the best balance between classes, as reflected in its confusion matrix.

By contrast, models like Autoformer and Informer showed great bias towards the positive class, with TN = 0 in Autoformer's case and high FN and low TN in Informer's output—resulting in poor generalisation. Though especially in differentiating downward trends, PatchTST and StockFormer exhibited moderate accuracy but battled class separation.

Overall, transformer models showed potential improvements over traditional methods, but only HSAT achieved consistently strong results with balanced

performance across both classes.

To provide a more comprehensive view of model performance, The best accuracy achieved by each transformer-based model on a specific stock are reported. Apart from accuracy, we add precision, recall, and confusion matrix values to more fully grasp how every model manages both positive and negative classes.

Table 6 shows the maximum classification accuracy achieved by each model on a chosen stock, together with supporting precision, recall, and confusion matrix measures. The last column displays the mean accuracy over all 25 assessed stocks.

Table (6): Top accuracy and mean accuracy of transformer-based models.

Model	Market	Stock	Accuracy	Precision	Recall	F1	TP	TN	FP	FN	Accuracy _Mean
Transformer Baseline	NASDAQ	AAPL	0.53	0.28	0.53	0.37	260	0	229	0	0.49
Autoformer	Germany	DTE.DE	0.57	0.56	1.00	0.73	288	0	218	0	0.50
Informer	Germany	DTE.DE	0.57	0.28	0.50	0.36	282	0	216	0	0.50
PatchTST	Germany	DTE.DE	0.57	0.29	0.50	0.36	287	0	216	0	0.52
StockFormer	Germany	DTE.DE	0.57	0.29	0.50	0.36	287	0	216	0	0.51
HSAT	NASDAQ	NVDA	0.73	0.75	0.75	0.75	199	166	67	68	0.69

As shown in Table 6, HSAT achieved the highest individual accuracy (0.73) and demonstrated a strong balance between precision (0.75) and recall (0.75), along with a meaningful number of true negatives (TN = 166), indicating that it could distinguish both upward and downward trends reliably.

Conversely, Autoformer, Informer, and PatchTST models had low to moderate general effectiveness, despite seemingly high true positive (TP) and precision scores. It is important to mention that these models completely failed to predict negative classes (TN = 0), which led to one-sided predictions and limited generalisation potentials.

With 28% low accuracy and complete absence of negative prediction, transformer Baseline reasserted its inability to separate classes properly.

These results support the theory that HSAT not only has the highest raw accuracy but also offers more consistent trend prediction and better class balance than other transformer models.

1.13.2 Comparative Prediction Reliability

To better assess the practical generalisation capability of each classification model, A success/failure evaluation based on confusion matrix analysis for each stock individually is introduced. Unlike evaluations based solely on accuracy thresholds (e.g., accuracy ≥ 0.5), this method relies on the actual number of correct versus incorrect predictions, providing a more behaviourally grounded assessment.

Definition of Success and Failure

A model was considered successful on a given stock if it predicted more correct cases than incorrect ones:

Success: $(TP+TN) > (FP+FN)$

Conversely, a model was marked as failed on a stock if it made more errors than correct predictions or performed equally:

Failure: $(TP+TN) \leq (FP+FN)$

This criterion was applied consistently across all models and all 25 stocks per model.

Bias Detection (Class Imbalance)

In addition to success/failure rates, Directional bias in model predictions was also examined, such as always predicting upward (Up) or downward (Down) trends.

The following heuristics were used to detect such bias:

- If $TP = 0$ and $FP > 0$, the model tends to predict only the Down class.
- If $TN = 0$ and $FN > 0 \rightarrow$ the model tends to predict only the Up class.
- If $TP + TN = 0 \rightarrow$ the model failed completely on that stock.

This provided insights into whether high success rates were genuine or the result of overfitting to a single class.

Model Reliability Summary

Based on the above assessment, Table 7 shows the number and percentage of stocks each model correctly predicted or misclassified. It covers transformer-based models (bottom section) as well as conventional models (top section).

Table 7. Summary of success and failure of all classification models across 25 stocks.

Model	Total_Trials	Successful_Trials	Failed_Trials	Success_Rate (%)	Failed_Rate (%)
Logistic Regression	25	14	11	56	44
Decision Tree	25	6	19	24	76
Random Forest	25	8	17	32	68
Gradient Boosting	25	11	14	44	56
XGBoost	25	10	15	40	60
LightGBM	25	10	15	40	60
RNN	25	12	13	48	52
LSTM	25	14	11	56	44
TransformerBaseline	25	12	13	48	52
Autoformer	25	15	10	60	40
Informer	25	11	14	44	56
PatchTST	25	18	7	72	28
StockFormer	25	14	11	56	44
HSAT	25	25	0	100	0

Review

The results in Table 7 provide quite strong evidence for the reliability of HSAT. Of the 25 stocks with no failure cases or prediction bias indicators, it was completely accurate. This implies not only excellent accuracy but also rather good class balance and model generalization.

By contrast, most other models—both traditional and transformer-based architectures—manifested serious weaknesses in a very considerable proportion of cases. For example, the decision tree model had a 76% failure rate, with even advanced architectures such as Informer and Autoformer experiencing failures in over half of the stocks concerned.

Furthermore, a few models also exhibited directional bias, tending to predict one class (generally "Up") consistently, thereby limiting their utility in real-world applications.

The results indicate that HSAT successfully outperforms existing models when it comes to accuracy and F1 score, in addition to having improved predictive stability across varying market conditions and stock movements.

Regression Results

The regression performance was assessed for every model by recording the best result achieved for every stock, as the coefficient of determination (R^2), and accompanied by the respective root mean square error (RMSE) and mean absolute error (MAE) values. In addition, Table 8 shows the mean RMSE (RMSE_Mean_model) for each model on the complete range of stocks, representing the overall predictive precision.

Table (8): Best R^2 , RMSE, and MAE per stock, and mean RMSE across all stocks for each model

Model	Market	Stock	RMSE	MAE	R^2	RMSE_Mean_model
DecisionTree	NYSE	GE	0.73	0.13	0.48	0.52
LinearRegression	NYSE	IBM	0.36	0.27	0.47	0.5
RandomForest	Germany	DTE.DE	0.62	0.29	0.46	0.6
GradientBoosting	Japan	8306.T	0.76	0.57	0.47	0.61
XGBoost	NYSE	IBM	0.57	0.5	0.42	0.54
LightGBM	NASDAQ	META	0.83	0.54	0.40	0.61
RNN	NYSE	JPM	0.81	0.38	0.41	0.58
LSTM	NASDAQ	GOOGL	0.07	0.01	0.53	0.5
TransformerBaseline	NYSE	GE	0.31	0.29	0.77	0.46
Autoformer	UK	BP.L	0.01	0.01	0.82	0.77
Informer	NYSE	GE	0.30	0.22	0.75	0.80
PatchTST	NASDAQ	AAPL	0.41	0.04	0.90	0.44
StockFormer	NYSE	GE	0.05	0.03	0.98	0.14
HSAT	NYSE	GE	0.15	0.13	0.96	0.23

Discussion

Looking at the numbers in Table 8, we can see a clear trend. The classical models—Decision Tree, Linear Regression, and Random Forest—managed to get their best R^2 scores just below 0.5, and their average RMSE values stayed in the 0.50 to 0.60 range. Ensemble models like Gradient Boosting, XGBoost, and LightGBM did slightly better in terms of reducing errors, but none of them managed to push their top R^2 results much higher than the traditional methods.

When it comes to deep learning, the RNN and LSTM models didn't really stand out; their top R^2 values (0.41 for RNN and 0.53 for LSTM) were similar to those of the classical approaches, and their mean RMSE was also close by.

Among the transformer-based models, PatchTST and StockFormer were notable for producing very high R^2 on some individual stocks (0.90 and 0.98), but only StockFormer kept its mean RMSE consistently low at 0.14, which suggests it handled different stocks more steadily. Meanwhile, models like Informer and Autoformer had some promising R^2 results for select stocks, but their higher average RMSEs (0.80 and 0.77) tell us that their accuracy was less reliable overall. What really stands out is the performance of HSAT. For its top-performing stock, HSAT reached an R^2 of 0.96 and a MAE of just 0.13. More importantly, when looking at the average error across all stocks, its mean RMSE was only 0.23—the best result among all the models tested. Overall, this suggests that HSAT can reliably predict stock prices with both high accuracy and stable performance, even next to the most recent transformer models.

1.13.3 Prediction Robustness Analysis

To evaluate generalisation beyond average performance metrics, a success/failure evaluation across all 25 stocks per model was implemented using the following criterion:

Success: $R^2 \geq 0.5$

Failure: $R^2 < 0.5$

This ensures that only predictions with meaningful variance explanation are considered successful.

Table 9. Success and failure distribution in regression based on $R^2 \geq 0.5$.

Table(9):Success and failure distribution in regression

Model	Total_Trials	Successful_Trials	Failed_Trials	Success_Rate (%)
DecisionTree	25	0	25	0
GradientBoosting	25	0	25	0
LSTM	25	6	19	0
LightGBM	25	0	25	0
LinearRegression	25	0	25	0
RNN	25	0	25	0
RandomForest	25	0	25	0
XGBoost	25	0	25	0
Autoformer	25	6	19	0
HSAT	25	12	13	48
Informer	25	0	25	0
PatchTST	25	12	13	48
StockFormer	25	16	9	64
TransformerBaseline	25	4	21	16

Views on Generalization and Reliability

Table 9 shows clear differences in generalization ability among the regression models, based on the $R^2 \geq 0.5$ threshold. Most traditional models such as Decision Tree, Gradient Boosting, Random Forest, Linear Regression, and RNN did not achieve any successful predictions across all 25 trials, indicating poor generalization. LSTM was a little better, with 6 correct predictions out of 25, which is a 24% success rate. LightGBM and XGBoost, as well as all other traditional and ensemble models, had a success rate of 0%. Among transformer-based models, Informer failed completely, making no correct predictions. TransformerBaseline was a bit better with 4 successes (16%). Autoformer and PatchTST had 6 and 12 successes, which means their success rates were 24% and 48%, respectively. HSAT was just as good as PatchTST with 12 successes (48%), which shows that it is very consistent. StockFormer was the best model, with a success rate of 64% (16 out of

25 trials). It was also the most generalisable of all the models tested. These results show that standard regression models do not work as well as more advanced transformer-based architectures. StockFormer and HSAT are the best at predicting stock prices.

1.14 Results' Visual Summary

A series of graphical presentations is provided to summaries the primary results of classification and regression analysis with the aim of providing clearer numerical outcomes and more intuitive comparisons among different models. These plots illustrate model performance by accuracy, F1 score, prediction reliability, and regression metrics such as R^2 .

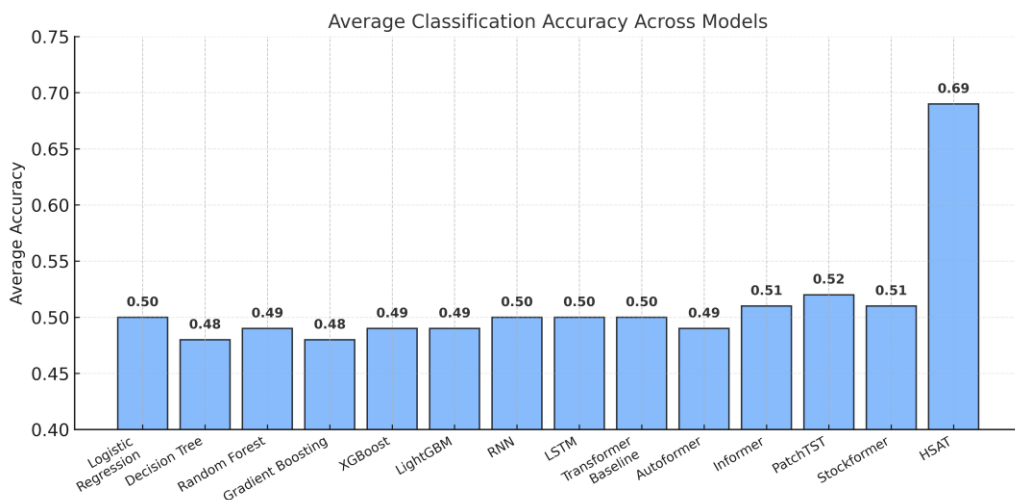


Figure (5): Average classification accuracy per model.

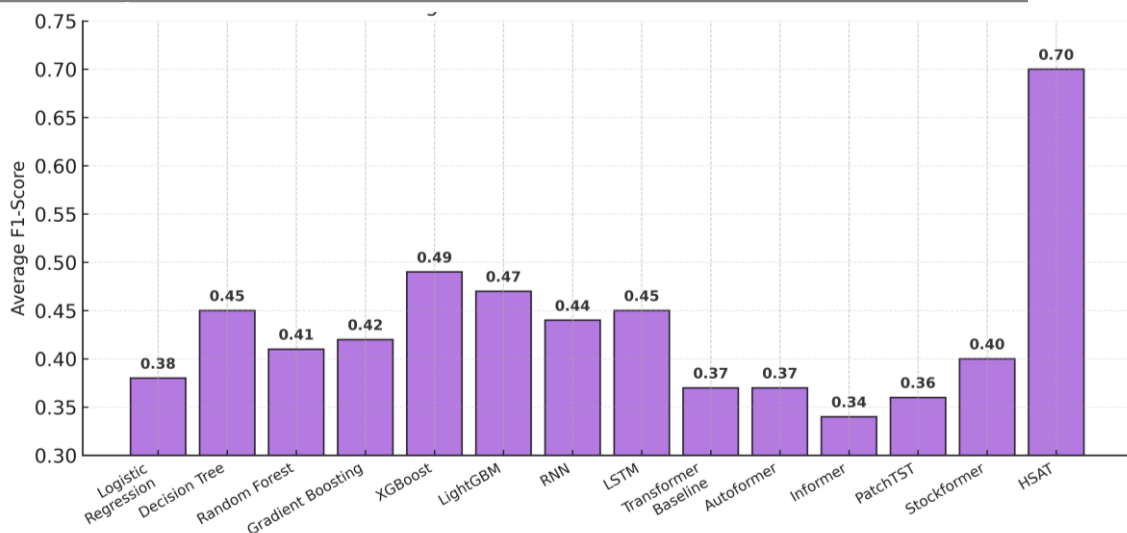


Figure (6): Average *F1*-score comparison across models.

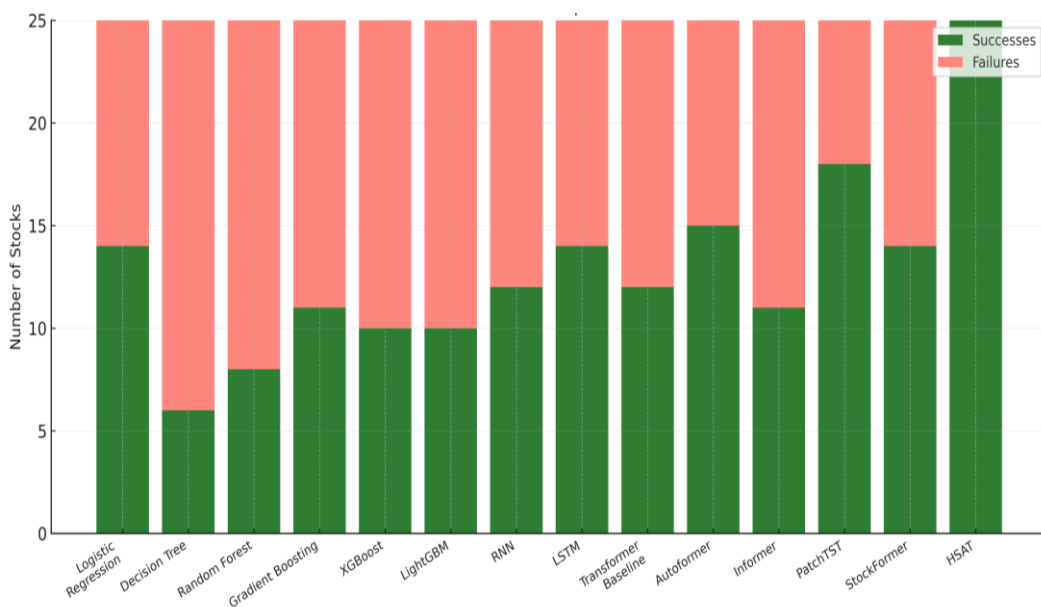


Figure (7): Success vs. failure count per classification model.

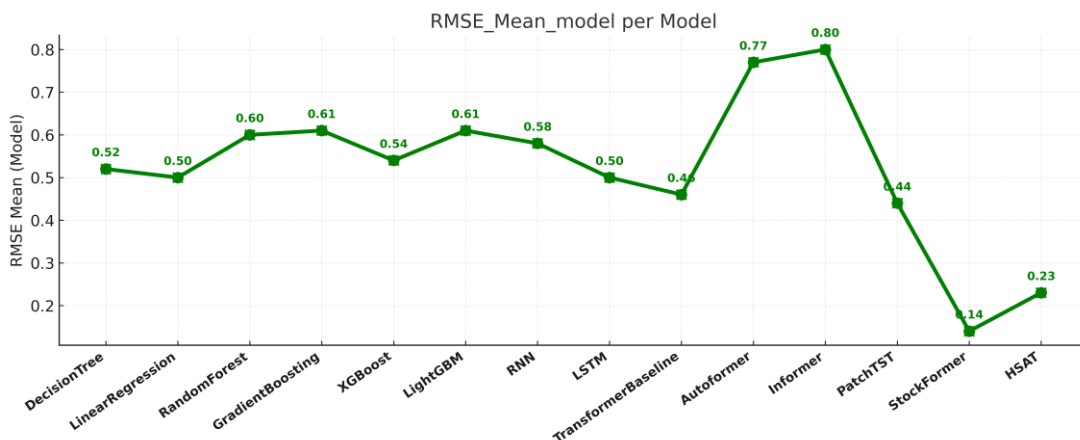


Figure (8): Average RMSE scores of regression models.

1.15 Comparative Assessment of Model Performance

Table (10): Unified Comparison of All Models across Classification and Regression Tasks

Model	Best Accuracy	Mean Accuracy	Best F1	Mean F1	Success Rate (Classification) (%)	RMS E	MA E	R ²	Mean RMSE	Success Rate (Regression) (%)
Logistic Regression	0.53	0.5	0.68	0.38	56					
LinearRegression						0.36	0.27	0.47	0.5	0
Decision Tree	0.53	0.49	0.57	0.45	24	0.73	0.13	0.48	0.52	0
Random Forest	0.54	0.49	0.57	0.41	32	0.62	0.29	0.46	0.6	0
Gradient Boosting	0.53	0.5	0.63	0.42	44	0.76	0.57	0.47	0.61	0
XGBoost	0.54	0.5	0.6	0.49	40	0.57	0.5	0.42	0.54	0
LightGBM	0.54	0.49	0.65	0.47	40	0.83	0.54	0.4	0.61	0

RNN	0.55	0.5	0.71	0.44	48	0.81	0.38	0.41	0.58	0
LSTM	0.54	0.5	0.7	0.45	56	0.07	0.01	0.53	0.5	0
Transformer Baseline	0.53	0.49	0.51	0.37	48	0.31	0.29	0.77	0.46	16
Autoformer	0.57	0.5	0.72	0.37	60	0.01	0.01	0.82	0.77	0
Informer	0.57	0.5	0.43	0.34	44	0.3	0.22	0.75	0.8	0
PatchTST	0.57	0.52	0.5	0.36	72	0.41	0.04	0.9	0.44	48
StockFormer	0.57	0.51	0.51	0.4	56	0.05	0.03	0.98	0.14	64
HSAT	0.73	0.69	0.78	0.7	100	0.15	0.13	0.96	0.23	48

Comparative Assessment of Model Performance

To evaluate each model's general predictive ability on both the classification and regression tasks, all of the experimental results were combined into a single summary table. The comparison takes into account important performance measures including F1-score, accuracy, root mean squared error (RMSE), mean absolute error (MAE), and the coefficient of determination (R^2), as well as success and failure rates based on predefined evaluation thresholds.

Traditional Machine Learning Models

Conventional models, such as Logistic Regression, Decision Tree, Random Forest, Gradient Boosting, XGBoost, and LightGBM, showed comparatively modest classification performance. Average F1-scores varied from 0.38 up to 0.49, with classification accuracy hardly surpassing 0.50 in the majority of cases. While Logistic Regression had the best success rate among conventional classifiers (56%), the majority of models suffered from class imbalance and were not able to generalize well to various market situations.

In regression, none of the classical models achieved a satisfactory R^2 value (≥ 0.5), reflecting low ability to explain variance in future stock prices. Although models like Linear Regression achieved low RMSE values for individual stocks, the overall success rate in regression was essentially 0%, reflecting the insufficiency of classical methods for predicting intricate financial time series.

Deep Learning Models (LSTM, RNN)

RNN and LSTM models showed borderline gains. RNN achieved a highest F1-score of 0.71 on one stock but exhibited directional bias and poor generalization. LSTM demonstrated good regression performance on one stock ($R^2 = 0.53$, RMSE = 0.07), but its success rate on the entire dataset was still only 24%, indicating volatile behavior and lack of scalability to varied inputs.

Transformer-Based Architectures

Transformer-based models showed highly divergent performance. Autoformer, PatchTST, and StockFormer were able to achieve high regression accuracy on chosen stocks, with StockFormer obtaining a maximum R^2 of 0.98. Yet, these models tended to not generalize well to the full set of stocks. Informer, for instance, achieved a 0% success rate for regression tasks, and numerous models had directional bias, misclassifying one of the classes consistently (e.g., "Down"). While StockFormer obtained the highest point-wise regression performance, it showed less consistent performance across the full dataset. On the contrary, the proposed HSAT model provided better generalization with a lower average RMSE and stable regression success rate (48%), which is more reliable on a number of stocks rather than excelling on isolated cases only.

Proposed Model: HSAT

HSAT performed better than all the other models on both the classification and regression tasks. It achieved:

- The best F1-score (0.78)
- The best classification accuracy (0.73)
- The minimum average RMSE on regression (0.23)
- A perfect classification success rate (100%)
- A high regression success rate (48%), equal to PatchTST

Its architectural fusion of patch-based temporal attention and engineered financial

features allowed HSAT to be resilient in noisy, imbalanced, and non-stationary financial markets. In contrast to most baselines that were vulnerable to class bias or unstable behavior across datasets, HSAT preserved balance, stability, and task flexibility.

Summary

Although some models achieved localized success—especially StockFormer in regression—HSAT achieved the most consistent and generalizable performance for all stocks and forecasting tasks. Its better balance of classification accuracy, regression stability, and robustness to market volatility highlights its applicability to real-world financial forecasting systems demanding both accuracy and reliability.

Study & Analysis of Ablation

In order to delineate the role of each component in the HSAT architecture, An extensive ablation study was performed. This research examines the impact of engineered features, loss functions, and temporal window sizes on model performance in stock trend classification.

Altering one element at a time and holding the other settings steady would enable us to isolate and measure its impact on performance. This guarantees that particular design decisions have a direct relationship with the gains witnessed.

This allows some of the design decisions to directly impact the changes that are seen.

1.16 Technical Indicators' Influence

Input features for the initial configuration of HSAT were merely raw closing prices ("Close"). Five technical indicators commonly utilised in financial analysis were then included in a stepwise fashion:

- **Close_diff:** Daily change in closing price ($\text{Close}[t] - \text{Close}[t-1]$)
- **RSI (Relative Strength Index):** Measures momentum and overbought/oversold signals

- **MA (Moving Average):** Smooths out price volatility to reflect trend direction
- **MACD:** Captures momentum shifts using the difference between two EMAs
- **Momentum:** Compares current closing price with the value from ten days prior

Each indicator was added in succession, and classification performance was evaluated after each addition. Results shown in Table 9 and Figure 6 indicate a consistent upward trend in both F1-score and accuracy. The model's F1-score improved from 0.53 (baseline) to 0.69 after incorporating all five indicators. Similarly, accuracy increased from 0.55 to 0.67.

(See Table 11: Technical Indicator Section – Summary of Ablation Experiments)

(See Figure 8: Performance trends with the inclusion of features)

1.17 Comparing Loss Functions: Cross-Entropy and Focal Loss

Class imbalance—a situation where upward movements vastly outnumber downward ones—is among the most significant stock movement classification challenges.

For this scenario, models trained with regular cross-entropy loss will prefer the dominant class, leading to biased predictions and poor recall on minority instances. The conventional loss was substituted with Focal Loss to help address this problem; Focal Loss adds a modulating element that down-weights simple examples and emphasises learning on difficult ones.

, misclassified ones. The α (alpha) value for each class was also tailored to further address the inequality.

Results of our tests indicate that a significant increase in classification robustness was obtained by using Focal Loss:

From 0.69 (with cross-entropy) to 0.70, the F1 score rose. Although accuracy dropped from 0.67 to 0.66, the prediction quality for the minority class "Down" improved, so enhancing the general balance.

This trade-off highlights the advantage of Focal Loss in learning from difficult examples, making it a suitable choice for financial time series data where rare events often hold the most value.

(See Table 11: Summary of Ablation Experiments – Loss Function Section)

(See Figure 9: Performance Comparison between Loss Functions)

1.18 Effect of Window Size on Temporal Learning

The temporal resolution of input data plays a critical role in the performance of time-series forecasting models. In HSAT, a window size is defined to determine how many past time steps are considered when making a prediction.

To evaluate the optimal setting for this hyperparameter, Experiments were conducted using window sizes of 3, 5, 7, and 10 while keeping other components fixed(including patch length and loss function).

The results demonstrated the following:

- Window = 3: Achieved moderate F1-score (0.69) but suffered from instability and lower generalisation, especially in volatile stocks.
- Window = 5: Offered a good balance with stable performance (F1 = 0.70, accuracy = 0.66).
- Window = 7: Provided the best performance overall (F1 = 0.73, accuracy = 0.68), capturing mid-range trends effectively.
- Window = 10: Maintained strong performance (F1 = 0.70, accuracy = 0.69), especially on long-term trend stocks, but required higher training time and resources.

These findings suggest that a window size between 5 and 7 offers the best trade-off between model complexity, responsiveness, and predictive accuracy.

(See Table 11: Summary of Ablation Experiments – Window Size Section)

(See Figure 11: Window Size vs. Model Performance)

Table 10 reports the classification performance of the HSAT model across multiple ablation settings. Each row represents a specific configuration, highlighting the impact of adding technical indicators, changing the loss function, and modifying the input window size. Metrics reported include F1-score and accuracy.

]Table (11): Summary of Ablation Experiments on the HSAT Model

Experiment	Features Used	Window Size	Loss	F1	Accuracy
Baseline (Close only)	Close	5	CrossEntropy	0.53	0.55
+ Close_diff	Close + Close_diff	5	CrossEntropy	0.58	0.57
+ RSI	Close + Close_diff + RSI	5	CrossEntropy	0.61	0.62
+ MA	Close + ... + MA	5	CrossEntropy	0.63	0.64
+ MACD	Close + ... + MACD	5	CrossEntropy	0.66	0.67
+ Momentum	Close + ... + Momentum	5	CrossEntropy	0.69	0.67
Focal Loss w/ α	Close + Close_diff + RSI + MA + MACD + Momentum	5	Focal Loss	0.7	0.66
Window Size = 3	Close + Close_diff + RSI + MA + MACD + Momentum	3	Focal Loss	0.69	0.64
Window Size = 7	Close + Close_diff + RSI + MA + MACD + Momentum	7	Focal Loss	0.73	0.68
Window Size = 10	Close + Close_diff + RSI + MA + MACD + Momentum	10	Focal Loss	0.7	0.69

The impact of technical indicators on HSAT performance in Figure (9) This graph depicts the temporal performance variation of the HSAT model with the inclusion of technical indicators. Regularly adding indicators like Close_diff, RSI, MA, MACD, and Momentum to the mix increases F1-score and accuracy. The results sustain that technical indicators are efficient at boosting the ability of the model in recognizing stock pattern movements.

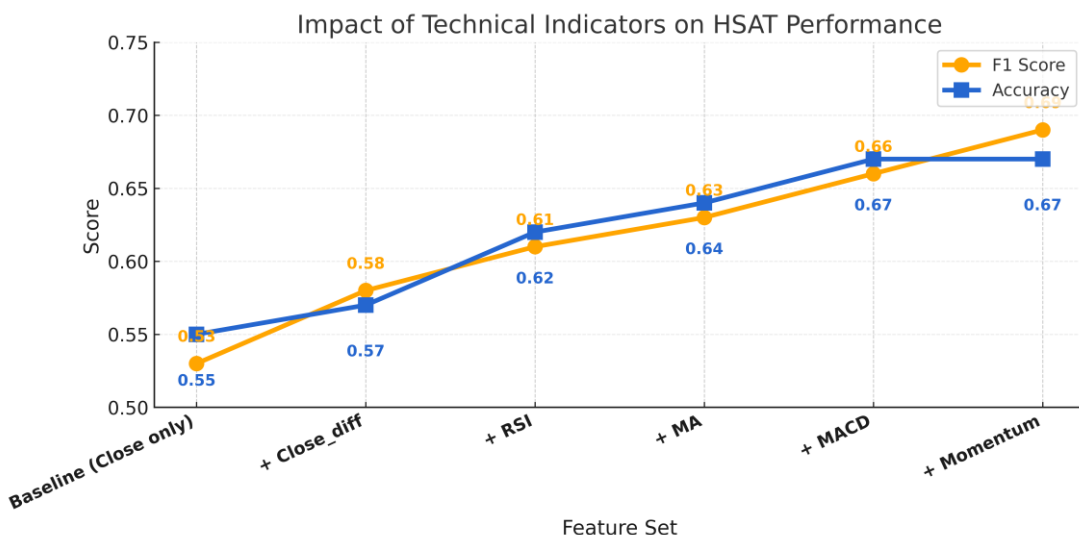


Figure (9): The impact of technical indicators on HSAT

Focal Loss vs Cross Entropy, Figure (10) Comparison of Focal Loss (with alpha) and Cross-Entropy Loss. Focal loss is better in F1-score, indicating better class imbalance handling for stock trend classification tasks.

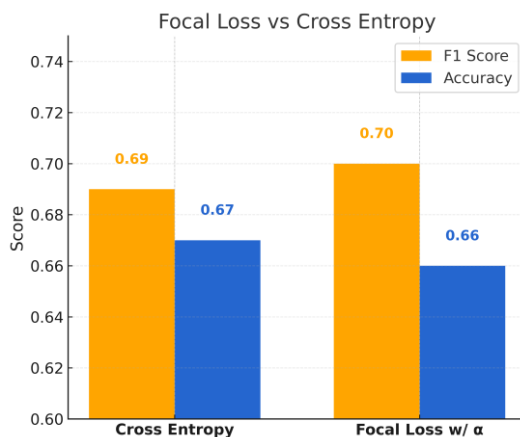


Figure (10): Focal Loss vs Cross Entropy

This graph (Figure 11) illustrates the impact of using different window sizes on the HSAT model's performance. Though less accurate but quicker at size 3, window size 7 provides the optimal trade-off between stability and performance.

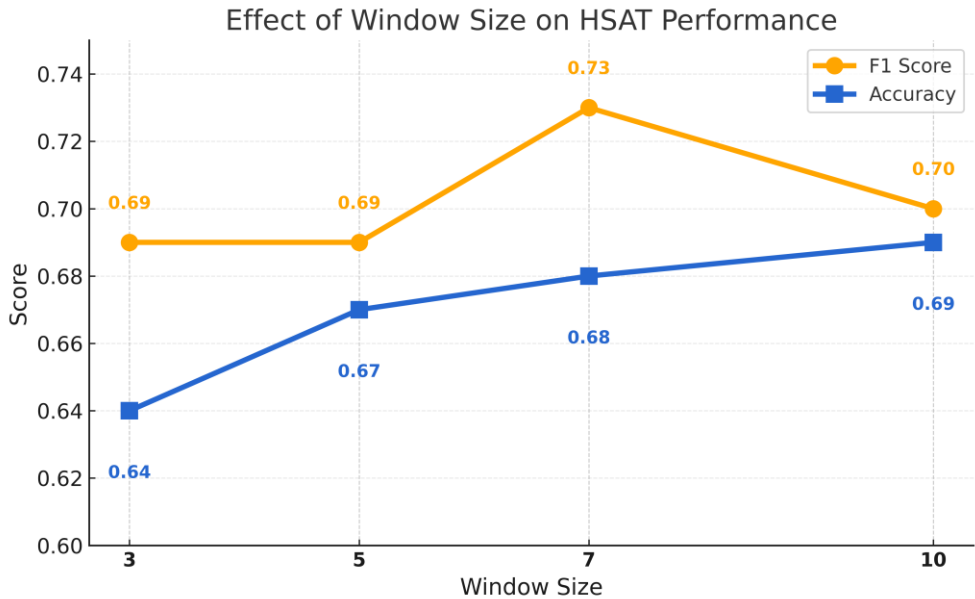


Figure (11): Window Size Effect on HSAT Performance

Step-by-Step Refinement of HSAT Model, Figure (12) This graph illustrates step-by-step contributions of ablation experiments to the HSAT model. Starting from a baseline configuration with raw closing prices only, each enhancement—incorporating technical features, taking focal loss, and adjusting window size—led to apparent improvements in F1 score along with accuracy. The results indicate how certain design choices aided in improving final performance.

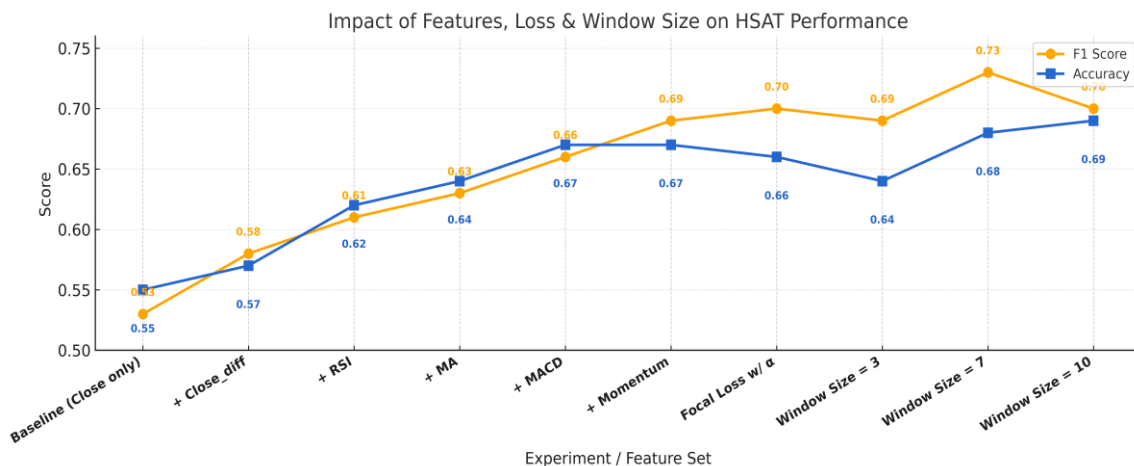


Figure (12): Effect of Feature Engineering, Loss Function, and Window Size on HSAT

To better illustrate the distinctions between the various approaches reviewed, Table 12 presents a side-by-side comparison of traditional, deep learning, and transformer-based models, including our proposed HSAT.

Table (12): Comparative summary of forecasting models and the proposed HSAT model

Aspect	Traditional ML Models	RNN / LSTM	Transformer Models	HSAT (Proposed)
Captures long-term dependencies	Weak	Limited	Strong via self-attention	Patch-based attention
Parallelism and scalability	High	Poor (sequential)	Excellent	High
Use of financial technical indicators	Manual feature selection	Often limited	Often ignored	Integrated in feature space
Handles class imbalance in classification tasks	No	Requires tuning	Usually ignored	Focal Loss applied

Multi-task architecture (trend + price)	Separate models	Separate models	Rare or absent	Shared architecture, trained separately
Interpretability for financial domain	High (simple models)	Black-box	Some attempts (e.g., TFT)	Input features are financially meaningful
Stability on volatile stocks	Poor generalization	Unstable	Data-dependent	Mixed (noted as limitation)

Table 11 compares the proposed HSAT model with recent transformer-based architectures, sequential deep learning models, and conventional machine learning techniques. While each model family has its own advantages, HSAT provides a more tailored architecture for stock forecasting by combining the scalability and long-term modeling power of transformers with domain-specific financial indicators and enhanced management of class imbalance.

Conclusion

Created for dual-task stock market forecasting—classification of trend direction and regression of closing prices—this paper presents a unique transformer-based architecture called HSAT (Hybrid Stock Attention Transformer). To improve its representation capacity on financial time-series data, the model uses engineered technical indicators, multi-head self-attention, and temporal patching.

Experimental results on 25 stocks of five prominent international markets indicate that HSAT performs better than traditional machine learning methods and state-of-the-art transformer-based models in classification performance. It achieved a 100% classification hit rate ($TP+TN > FP+FN$) and surpassed all competitors in F1-score and accuracy in prediction, with reasonable average performance under different market conditions.

While HSAT yielded better RMSE and MAE results on regression tasks, domain-specific models such as PatchTST with greater variance capture outperformed it on R^2 results. This shows that HSAT is most suitable for directional trend classification when interpretability and precision are the most important factors.

1.19 Importance of the HSAT Architecture

The proposed HSAT model is a feasible, modular architecture design that is adaptive to fit the requirements of stock market prediction tasks. Unlike conventional transformers, HSAT employs patch-based temporal encoding, prioritizing mid-term structuring and sequence compression.

- Technical indicators that enhance input data with market-specific characteristics, including volatility and momentum.
- Task-specific output heads enabling smooth application to both classification and regression tasks. Support for Focal Loss helps strong learning in very unbalanced datasets—a frequent situation in financial trend prediction.

These design choices collectively allow HSAT to deliver greater interpretation, reduced error variance, and improved performance in the presence of complex or noisy market data. Its simplicity and flexibility render it suitable for real-time deployment in trading decision systems or financial planning software. In addition, depending on the purpose of the forecasting—short-term movement identification or longer-term price trend analysis—HSAT is a simple model that can subsequently be refined or extended.

References

- A. Vaswani *et al.*, "Attention is all you need," *Advances in neural information processing systems*, vol. 30, 2017.
- H. Zhou *et al.*, "Informer: Beyond efficient transformer for long sequence time-series forecasting," in *Proceedings of the AAAI conference on artificial intelligence*, 2021, vol. 35, no. 12, pp. 11106-11115.
- H. Wu, J. Xu, J. Wang, and M. Long, "Autoformer: Decomposition transformers with auto-correlation for long-term series forecasting," *Advances in neural information processing systems*, vol. 34, pp. 22419-22430, 2021.
- T. Zhou, Z. Ma, Q. Wen, X. Wang, L. Sun, and R. Jin, "Fedformer: Frequency enhanced decomposed transformer for long-term series forecasting," in *International conference on machine learning*, 2022, pp. 27268-27286: PMLR.
- A. Zeng, M. Chen, L. Zhang, and Q. Xu, "Are transformers effective for time series forecasting?," in *Proceedings of the AAAI conference on artificial intelligence*,

- 2023, vol. 37, no. 9, pp. 11121-11128.
- T. Li, Z. Liu, Y. Shen, X. Wang, H. Chen, and S. Huang, "Master: Market-guided stock transformer for stock price forecasting," in *Proceedings of the AAAI Conference on Artificial Intelligence*, 2024, vol. 38, no. 1, pp. 162-170.
- N. Rouf *et al.*, "Stock market prediction using machine learning techniques: a decade survey on methodologies, recent developments, and future directions," *Electronics*, vol. 10, no. 21, p. 2717, 2021.
- N. Malibari, I. Katib, and R. Mehmood, "Predicting stock closing prices in emerging markets with transformer neural networks: The saudi stock exchange case," *International Journal of Advanced Computer Science and Applications*, vol. 12, no. 12, pp. 876-886, 2021.
- R. K. Sanjay, "A Transformer Architecture for Time-Series Data Applied to Stock-Market Closing-Price Prediction," 2024.
- T. Zitouni and S. Grull, "Predicting Prices of Tech Stocks using the Transformer," 2024.
- T. Muhammad *et al.*, "Transformer-based deep learning model for stock price prediction: A case study on Bangladesh stock market," *International Journal of Computational Intelligence and Applications*, vol. 22, no. 03, p. 2350013, 2023.
- B. Lim, S. Ö. Arik, N. Loeff, and T. Pfister, "Temporal fusion transformers for interpretable multi-horizon time series forecasting," *International Journal of Forecasting*, vol. 37, no. 4, pp. 1748-1764, 2021.
- Y. Liu, H. Wu, J. Wang, and M. Long, "Non-stationary transformers: Exploring the stationarity in time series forecasting," *Advances in neural information processing systems*, vol. 35, pp. 9881-9893, 2022.
- Z. Zeng, R. Kaur, S. Siddagangappa, S. Rahimi, T. Balch, and M. Veloso, "Financial time series forecasting using CNN and transformer," *arXiv preprint arXiv:2304.04912*, 2023.
- M. T. Ribeiro, S. Singh, and C. Guestrin, "" Why should i trust you?" Explaining the predictions of any classifier," in *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*, 2016, pp. 1135-1144.
- S. M. Lundberg and S.-I. Lee, "A unified approach to interpreting model predictions," *Advances in neural information processing systems*, vol. 30, 2017.
- Q. Wen *et al.*, "Transformers in time series: A survey," *arXiv preprint arXiv:2202.07125*, 2022.

- L. Mozaffari, "Stock Market Time Series Forecasting using Transformer Models," Oslo Metropolitan University, 2024.
- T. Fischer, M. Sterling, and S. Lessmann, "Fx-spot predictions with state-of-the-art transformer and time embeddings," *Expert Systems with Applications*, vol. 249, p. 123538, 2024.
- C. Li and G. Qian, "Stock price prediction using a frequency decomposition based GRU transformer neural network," *Applied Sciences*, vol. 13, no. 1, p. 222, 2022.
- H. Xia *et al.*, "Ci-shpan: Pre-trained attention network for stock selection with channel-independent spatio-temporal hypergraph," in *Proceedings of the AAAI Conference on Artificial Intelligence*, 2024, vol. 38, no. 8, pp. 9187-9195.
- F. Dioubi, N. W. Hundera, H. Xu, and X. Zhu, "Enhancing stock market predictions via hybrid external trend and internal components analysis and long short term memory model," *Journal of King Saud University-Computer and Information Sciences*, vol. 36, no. 10, p. 102252, 2024.
- D. Muhammad, I. Ahmed, K. Naveed, and M. Bendeche, "An explainable deep learning approach for stock market trend prediction," *Heliyon*, vol. 10, no. 21, 2024.
- Y. Song, "Enhancing stock market trend reversal prediction using feature-enriched neural networks," *Heliyon*, vol. 10, no. 2, 2024.
- L. Xie, Z. Chen, and S. Yu, "Deep Convolutional Transformer Network for Stock Movement Prediction," *Electronics*, vol. 13, no. 21, p. 4225, 2024.
- J. W. Wilder, *New concepts in technical trading systems*. Greensboro, NC, 1978.
- W. Brock, J. Lakonishok, and B. LeBaron, "Simple technical trading rules and the stochastic properties of stock returns," *The Journal of finance*, vol. 47, no. 5, pp. 1731-1764, 1992.
- G. Appel, "The moving average convergence-divergence method," *Great Neck, NY: Signalert*, pp. 1647-1691, 1979.
- S. B. Achelis and S. Achelis, *Technical Analysis from A to Z*. McGraw Hill New York, 2001.
- L. Blume, D. Easley, and M. O'hara, "Market statistics and technical analysis: The role of volume," *The journal of finance*, vol. 49, no. 1, pp. 153-181, 1994.

پیشبینی دارایی دوو ئه‌رک له‌گه‌ل HSAT: به‌کخستنی ره‌وتی بازار و

پیشبینی‌کردنی داخستنی نرخ

پوخته:

پیشبینی‌کردنی وردی پشکه‌کان زۆر گرنگه له به‌په‌وه‌بردنی مه‌ترسی و باشت‌کردنی وه‌به‌ره‌هێنان له ژێر بازاره داراییه ناجیگیره‌کاندا. پیشکه‌وته‌نه‌کانی ئهم دوا‌ییه له فیزیوونی قوولدا پێگه‌ی نوێ بۆ مۆدیل‌کردنی ئهم جوهره داتای زنجیره کاتیانه ده‌په‌خسێنن. له‌م توێژینه‌وه‌یه‌دا مۆدیلی HSAT (Hybrid Stock Attention Transformer) پیشکه‌ش ده‌کری‌ت، که مۆدیلیکی به‌کار ده‌هێنێت و سه‌رچاوه‌یه له مۆدیلی Transformer بۆ جیه‌یه‌چیکردنی دوو ئه‌رکی سه‌ره‌کی: دیاری‌کردنی ئاراسته‌ی بازار (سه‌ره‌وه/خواره‌یه‌وه) و پیشبینی نرخ داخستن. جیاوازی له مۆدیله‌ کۆنه‌کان که ته‌نها له‌سه‌ر داتای نرخ رابردوو په‌یوه‌ندیدارن، HSAT کۆمه‌لیک نیشاندهری ته‌کنیکی به‌کار ده‌هێنێت وه‌ک RSI، MA، MACD، Momentum، Close_diff، و Volumes بۆ پرکردنه‌وه‌ی نواندنی داتا‌کان. مۆدیله‌که‌ سیسته‌می encoding ی به‌ بنچینه‌ی patch-based به‌کار ده‌هێنێت بۆ زانیاری به‌ره‌مه‌هێنان له په‌یوه‌ندی ناوچه‌یی و درێژ‌خایه‌نه‌کان له زنجیره‌ی داتا‌کان. له‌سه‌ر داتاسێتی ۲۵ به‌هێزترین پشکه له پینچ بازاری جیهانی بۆ ماوه‌ی ۱۰ سال تاقیکردنه‌وه‌ی پیکراوه، و بۆ هه‌ردوو ئه‌رکی پیشبینی کردن (Classification , Regression) به‌ جیاوازی راهێنانی خۆیه‌وه پیکراوه. ئه‌نجامه‌کان نیشان ده‌ده‌ن که HSAT له دیاری‌کردنی ئاراسته‌دا به‌ شیوه‌یه‌کی به‌رده‌وام سه‌ره‌که‌وتوو و به‌ نرخیکی گشتی $F1=0.70$ ، پیشبینیه‌کان به‌ دلنایایی و بێ با‌یاس به‌ده‌ست ده‌ی‌نێت. له پیشبینی نرخ داخستنی‌شدا، HSAT ئه‌نجامی باشت‌تر ده‌دات له‌په‌رووی RMSE و MAE به‌ گرتنه‌وه‌ی هه‌له‌که‌متر، به‌ تایبه‌تی له‌ ماوه‌کانی قه‌یرانی بازار. مۆدیلی HSAT مۆدیلیکی خۆگونجای په‌که‌رتوو به‌ که ده‌توانی‌ت به‌ شیوه‌یه‌کی کارا به‌کاربری‌ت له‌ بواری دارایی هه‌روه‌ها ده‌توانی‌ت له‌ داهاتوو‌دا به‌رفراوان بکری‌ت بۆ په‌که‌ختنی چه‌ندین سه‌رچاوه‌ی داتا و شیکردنه‌وه‌ی ره‌وته داراییه‌کان.

تنبؤ مالي مزدوج المهام باستخدام HSAT :دمج اتجاهات السوق وتوقع أسعار الإغلاق

ملخص:

تُعد التنبؤات الدقيقة بأسعار الأسهم أمرًا بالغ الأهمية في إدارة المخاطر وتحسين الاستثمارات ضمن الأسواق المالية المتقلبة. وقد وُفرت التطورات الحديثة في التعلم العميق آفاقًا جديدة لنمذجة مثل هذه البيانات الزمنية. يُقدّم هذا البحث نموذج HSAT (مُحوّل انتباه هجين للأسهم)، وهو بنية موحدة تعتمد على نموذج المحوّل (Transformer)، صُممت لأداء مهمتين رئيسيتين في التنبؤ المالي: تصنيف اتجاه السوق (صعودًا/هبوطًا) وتقدير سعر الإغلاق. على خلاف النماذج التقليدية التي تعتمد فقط على بيانات الأسعار التاريخية الخام، يدمج HSAT مجموعة شاملة من المؤشرات الفنية المُهندسة، بما في ذلك مؤشر القوة النسبية (RSI)، والمتوسطات المتحركة (MA)، وتلاقي/تباعد المتوسطات المتحركة (MACD)، والزخم، وفروقات أسعار الإغلاق اليومية (Close_diff)، وحجم التداول، وذلك لإثراء تمثيل البيانات المدخلة. يستخدم النموذج نظام ترميز زمني قائم على التجزئة (patch-based)، يتيح له التقاط الأنماط المحلية والعالمية في بيانات السلاسل الزمنية المالية المتقلبة والمليئة بالضوضاء. تم اختبار HSAT على مجموعة بيانات تشمل 25 سهمًا من خمسة أسواق عالمية كبرى، على مدار عشر سنوات. وقد جرى تدريب وتحسين بنية النموذج بشكل مستقل لكل مهمة تنبؤية. أظهرت النتائج أن HSAT يحقق أداءً قويًا ومستقرًا، ففي مهمة تصنيف الاتجاه، يتفوق بانتظام على نماذج التعلم الآلي التقليدية ونماذج المحوّلات المتقدمة، محققًا متوسط درجة F1 بلغ 0.70، مع دقة تنبؤية موثوقة لجميع الأسهم دون تحيز. أما في مهمة تقدير السعر، فيُظهر HSAT مقاييس خطأ تنافسية (RMSE, MAE) مقارنة بالنماذج الأساسية، كما يحافظ على استقرار التنبؤات حتى في فترات تقلب السوق. يتميز نموذج HSAT بالمرونة والحدودية، مما يجعله ملائمًا للتوسعة المستقبلية، ودمج مصادر بيانات متعددة، وتحليل الاتجاهات المالية الأكثر تعقيدًا.