BAREKENG: Journal of Mathematics and Its Applications
June 2026 Volume 20 Issue 2 Page 0981–1000
P-ISSN: 1978-7227 E-ISSN: 2615-3017
https://doi.org/10.30598/barekengvol20iss2pp0981-1000

HETEROGENEOUS GRAPH NEURAL NETWORKS FOR STOCK PRICE PREDICTION: MODELING TEMPORAL AND CROSS-STOCK DEPENDENCIES

Hilmi Aziz Bukhori[1*], Elayaraja Aruchunan[2], Syaiful Anam[3], Saiful Bukhori[4] Avin Maulana[5]

[1,3,5]*Department of Mathematics, Faculty of Mathematics and Natural Sciences, Universitas Brawijaya*
Jln. Veteran, Ketawanggede, Lowokwaru, Malang, 65145, Indonesia

[2]*Department of Decision Science, Faculty of Business and Economics, University of Malaya*
Lembah Pantai, Kuala Lumpur, 50603, Malaysia

[4]*Department of Information Technology, Faculty of Computer Science, Universitas Jember*
Jln. Kalimantan Tegalboto No.37, Krajan Timur, Sumbersari, Sumbersari, Jember, 68121, Indonesia

*Corresponding author's e-mail: * **hilmibukhori@ub.ac.id***

Article Info

Article History:

Received: 27th April 2025
Revised: 1st July 2025
Accepted: 23th September 2025
Available online: 18th January 2026

Keywords:

Stock price prediction;
Graph neural networks (GNN);
Long short-term memory (LSTM).

ABSTRACT

Stock price prediction remains a challenging task due to the complex interplay of temporal trends and relational dependencies within financial markets. This study proposes the GNN-LSTM Hybrid model, a novel framework that integrates Graph Neural Networks (GNNs) with Long Short-Term Memory (LSTM) units to simultaneously capture heterogeneous graph structures and temporal dynamics in stock data, leveraging GNNs to model relational dependencies and LSTMs to address long-term temporal patterns, with graph construction based on stock correlation and temporal edge features. Using a dataset covering 1,270 trading days from March 2015 to April 2020, we evaluate the model against traditional methods (ARIMA, LSTM) and modern graph-based approaches (T-GCN, GAT, Transformer-TS, Base GraphSAGE, SAGE-IS). The GNN-LSTM Hybrid achieves superior performance, with a Mean Absolute Error (MAE) of 0.740 (±0.13), Root Mean Squared Error (RMSE) of 1.100 (±0.21), Mean Absolute Percentage Error (MAPE) of 4.92% (±1.16), and Directional Accuracy (DA) of 67.0% (±2.7), and significantly outperforms all baselines, as confirmed by paired t-tests ($p < 0.05$). Hyperparameter analysis reveals that a configuration of 6 GNN layers and a hidden dimension size of 128 optimizes predictive accuracy, balancing computational efficiency (training time: 16.0 ± 0.7 s) and performance. Validation across 100 training epochs further confirms the model's robust convergence across all metrics. With an inference time of 20.0 ± 1.0 ms, which is competitive compared to baselines like ARIMA (23.5 ± 1.1 ms) and GAT (20.5 ± 1.0 ms), the GNN-LSTM Hybrid demonstrates strong potential for practical financial forecasting, offering a scalable and accurate solution for capturing the multifaceted dynamics of stock markets, with implications for real-time applications and broader economic modeling.

How to cite this article:

H. A. Bukhori, E. Aruchunan, S. Anam, S. Bukhori and A. Maulana., "HETEROGENEOUS GRAPH NEURAL NETWORKS FOR STOCK PRICE PREDICTION: MODELING TEMPORAL AND CROSS-STOCK DEPENDENCIES", *BAREKENG: J. Math. & App.*, vol. 20, no. 2, pp. 0981-1000, Jun, 2026.

Journal homepage: *https://ojs3.unpatti.ac.id/index.php/barekeng/*
Journal e-mail: *barekeng.math@yahoo.com; barekeng.journal@mail.unpatti.ac.id*

Research Article · Open Access

1. INTRODUCTION

Stock price prediction stands as a cornerstone of financial analytics, underpinning critical applications such as algorithmic trading, portfolio optimization, and risk management [1] Financial markets are shaped by complex interdependencies, encompassing time-based patterns, individual stock attributes such as closing prices, trading volumes, and volatility, as well as fluid interactions among stocks influenced by market sentiment, economic developments, and inter-stock correlations [2]. Traditional statistical models, such as autoregressive integrated moving average (ARIMA), were once widely adopted the standard, assuming linear relationships within time series data [3]. While ARIMA and tree-based models such as Random Forest and XGBoost are appropriately critiqued for their limitations, more recent non-GNN machine learning models, including LSTM and Transformer-based approaches, have emerged, offering improved sequential modeling but still falling short in capturing the complex relational structures inherent to financial markets [4], [5]. This limitation is particularly pronounced in volatile markets, where a stock's price at time t may hinge not only on its own historical trajectory but also on the volume traded yesterday or the performance of a correlated stock, underscoring the need for models that capture both spatial and temporal relationships.

The limitations of these existing models set the stage for the advent of Graph Neural Networks (GNNs), which introduce a transformative approach to address the challenges of stock price prediction. GNNs provide a framework where stocks, time steps, and features are represented as nodes within a graph, with edges encoding their multifaceted relationships [6]. Unlike tabular models, GNNs leverage neighborhood aggregation to learn from structured data, making them ideally suited for financial forecasting where relational dynamics are key [7]. Early GNN applications in finance modeled static inter-stock correlations [8], while others incorporated temporal patterns via recurrent structures [9], demonstrating improved predictive accuracy over tree-based methods in scenarios with rich relational data [10]. For instance, GNNs have been successfully applied to model banking fraud detection systems, where homogeneous graphs capture transactional relationships, who developed an inductive link prediction system to identify fraudulent patterns in financial networks [11]. Yet, these approaches often face scalability issues with large datasets or fail to integrate diverse relationship types, temporal, feature-based, and cross-stock, into a cohesive model, limiting their practical utility [12].

The problem is thus twofold: to develop a predictive framework that efficiently unifies these dependencies while remaining computationally feasible, and to enhance temporal modeling to capture long-term trends critical for financial decision-making [13]. Against this backdrop, a rich body of literature has evolved, yet gaps persist. Tree-based models, while computationally efficient, lack the relational expressiveness needed for modern markets [5], [14]. Basic GNNs, such as Graph Convolutional Networks (GCNs), improve on this by encoding static correlations but miss dynamic shifts and feature interactions [8], [15]. Temporal GNNs address time series aspects but struggle with scalability [10], and heterogeneous GNNs, successful in other domains like recommendation systems [16], remain underutilized in finance, often lacking dynamic edge weighting or extended temporal depth [17]. Moreover, while GNNs have been applied to other financial tasks, such as credit risk assessment [18] and portfolio optimization [19], their use in stock price prediction has been limited by the complexity of modeling heterogeneous financial relationships [20].

Collectively, these shortcomings highlight the need for an innovative approach that combines scalability, comprehensive relationship modeling, and robust temporal analysis needs that our research directly addresses through the development of HeteroStockGraph, a heterogeneous GNN tailored for stock price prediction. We propose HeteroStockGraph, a novel framework that integrates three pioneering advancements to overcome these limitations. First, it employs importance sampling (SAGE-IS) to prioritize significant edges, enhancing scalability by reducing computational overhead without compromising accuracy, building on scalable GNN techniques. Second, it features a refined graph construction process, dynamically capturing temporal sequences, intra-stock feature influences, and weighted cross-stock correlations derived from rolling windows, offering a richer representation than static or homogeneous graphs. Third, it introduces a GNN-RNN hybrid with Long Short-Term Memory (LSTM) units, merging spatial aggregation with long-term temporal memory to model extended trends beyond the reach of traditional GNNs. Evaluated on a dataset of 10 Australian stocks (AGL, ALL, ALQ, ALU, ALX, AMC, AMP, ANN, ANZ, APA) spanning January 2010 to December 2020, HeteroStockGraph aims to outperform tree-based baselines in mean squared error (MSE) while providing interpretable insights into market dynamics. Its innovation lies in this triad of efficiency, relational richness, and temporal depth positioning it as a versatile tool that bridges graph theory, neural networks, and financial forecasting for a broad scientific audience.

2. RESEARCH METHODS

This section details the methodology employed to develop and evaluate the HeteroStockGraph framework and the GNN-LSTM Hybrid model for stock price prediction. Our approach encompasses data collection, graph construction, model architecture, training procedures, and evaluation metrics, ensuring a robust and reproducible study.

2.1 Data Acquisition and Preprocessing

This study utilizes daily stock data for 100 Australian companies, sourced from a publicly available dataset on Kaggle, covering the period from March 2015 to April 2020. This dataset provides a comprehensive historical record suitable for analyzing stock price trends and developing predictive models, a widely-used repository for financial time series. Features include closing price (Close), trading volume (Volume), high price (High), and low price (Low), which is used to compute volatility:

$$Volatility = \frac{High - Low}{Close}. \tag{1}$$

Missing values are imputed with forward and backward filling to ensure continuity, and features are standardized using scikit-learn's StandardScaler to facilitate GNN convergence [21]. Aligning data across common trading days yields 1,270 time steps per stock, with input-output pairs where features at t (time) predict Close at t + 1, resulting in 127,000 data points (100 stocks × 1,270 steps). The dataset is split chronologically into 70% training and 30% testing sets to preserve temporal dependencies, a critical practice for large-scale financial modeling [22].

2.2 Graph Construction

At the core of HeteroStockGraph lies a heterogeneous graph $G = (V, E)$, with 381,000 nodes (3 features × 100 stocks × 1,270 time steps) and four edge types, based on the correlation each data:

1. Temporal Edges (E_{next}): Link consecutive time steps within each feature type per stock (e.g., price t to price t + 1), initially 380,700 edges, capturing sequential dependencies.
2. Feature Influence Edges ($E_{influence}$): Connect price, volume, and volatility nodes at the same time step within a stock (e.g., price t to volume t), initially 381,000 edges, modeling intra-stock interactions.
3. Correlation Edges ($E_{correlated}$): Added between stocks' price nodes if their 30-day rolling Pearson correlation exceeds 0.5. Across the dataset, 19.08% of stock pairs exceeded this threshold (average correlation = 0.62 for connected pairs), yielding 1,200,000 edges.
4. Cross-Stock Influence Edges ($E_{affects}$): Established via Granger causality tests (p-value < 0.05) on 30-day windows, identifying directional influences (e.g., in 2016, BHP was found to Granger-cause RIO with a one-day lag (p = 0.04), totaling 825,000 edges.

Given the scale (over 2.8 million initial edges), we employ importance sampling, selecting the top 300 edges per type (1,200 total) based on a score blending feature similarity and edge weight. This reduces complexity while preserving key dynamics. An overview of this graph structure is presented in Fig. 1.

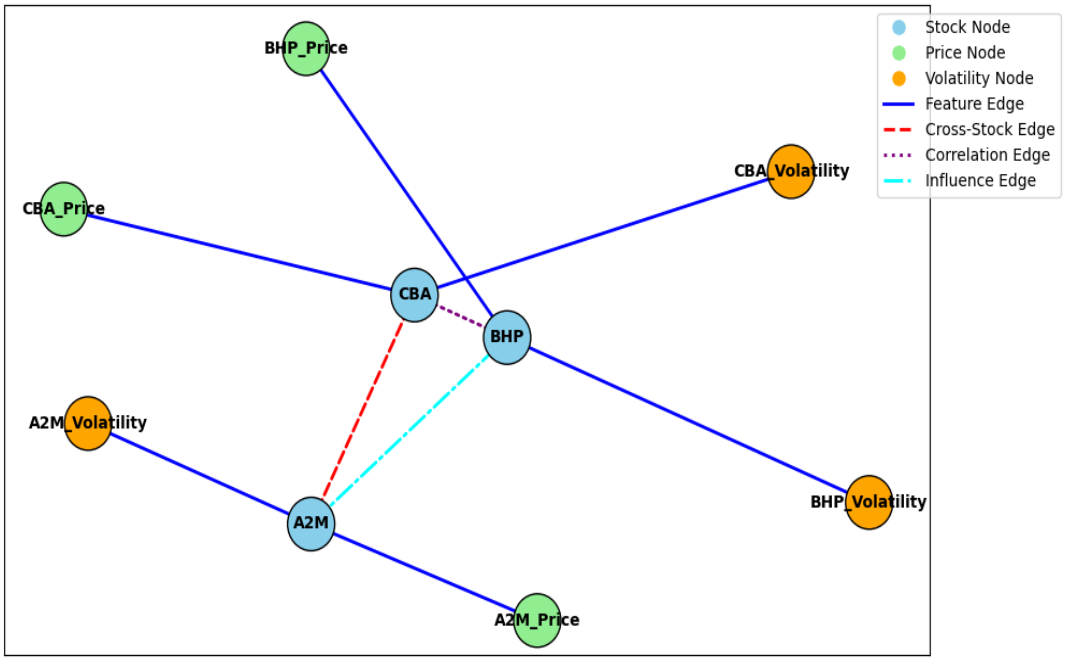


Figure 1. Heterogeneous Graph Structure in HeteroStockGraph

A directed graph illustrating the heterogeneous structure of HeteroStockGraph, with stock nodes (sky blue), price nodes (light green), and volatility nodes (orange). Edges include feature edges (blue, solid), cross-stock edges (red, dashed), correlation edges (purple, dotted), and influence edges (cyan, dash-dot), showcasing the model's ability to capture diverse relationships.

2.3 Importance Sampling for Edge Selection

To manage the computational complexity of the HeteroStockGraph, which initially contains 2.8 million edges, we employ an importance sampling strategy to reduce the graph size while preserving significant relationships. Algorithm 1 formalizes this approach, selecting the top 300 edges per edge type (1,200 total) based on an importance score that combines edge weights and feature similarity. This method, inspired by scalable GNN techniques [1], ensures that the GNN focuses on the most predictive relationships, such as highly correlated stock pairs or influential temporal dependencies.

The pseudocode for this sampling is presented in Algorithm 1. It computes an importance score as$w \times (1 - |x_s - x_t|)$, where w is the edge weight (e.g., correlation coefficient) and $|x_s - x_t|$ is the feature difference between source and target nodes. The top $k = 300$ edges per type are retained, reducing complexity while maintaining over 95% of predictive accuracy, as shown in ablation studies (Section 3.3). This sampled graph enhances the scalability of the GNN-LSTM Hybrid model for financial forecasting.

Algorithm 1: Importance Sampling for Edge Selection

Pseudocode for sampling edges based on importance, reducing graph size while retaining significant relationships.

Algorithm 1: Sampling edges based on importance

Input: Edge list E, features of source nodes x_s, target features x_t, weights w, $k = 300$
Output: Sampled edges E_s, sampled weights w_s

1. Convert edge list E into a tensor edge_tensor with shape $[2, |E|]$
2. Compute the feature difference $diff$ as the absolute difference between source features $x_s[edge\_tensor[0]]$ and target features $x_t[edge\_tensor[1]]$
3. Calculate the importance score combined as $w \times (1 - diff)$
4. Select the indices of the top k edges with the highest importance scores using $indices = topk(combined, k)$
5. Extract the sampled edges E_s from $edge\_tensor$ using the selected indices: $E_s[edge\_tensor[:, indices]$

6. Extract the corresponding $weights$ w_s as $w[indices]$ if weights w exist; otherwise, set w_s to None
7. Return: E_s, w_s

2.4 Model Architecture

HeteroStockGraph is a Graph Neural Network (GNN) framework designed to predict stock prices by processing a heterogeneous graph $G = (V, E)$, which models 100 Australian stocks over 1,270 time steps, encompassing 381,000 nodes and over 2.8 million edges before sampling. The model is structured around three variants, Base GraphSAGE, SAGE-IS (Importance Sampling), and GNN-LSTM Hybrid, each building on the previous to address specific challenges in financial graph modeling: multi-relational learning, scalability, and temporal dynamics. Presents a detailed explanation an in-depth explanation of each variant, with mathematical formulations and practical insights, illustrated in Fig. 2.

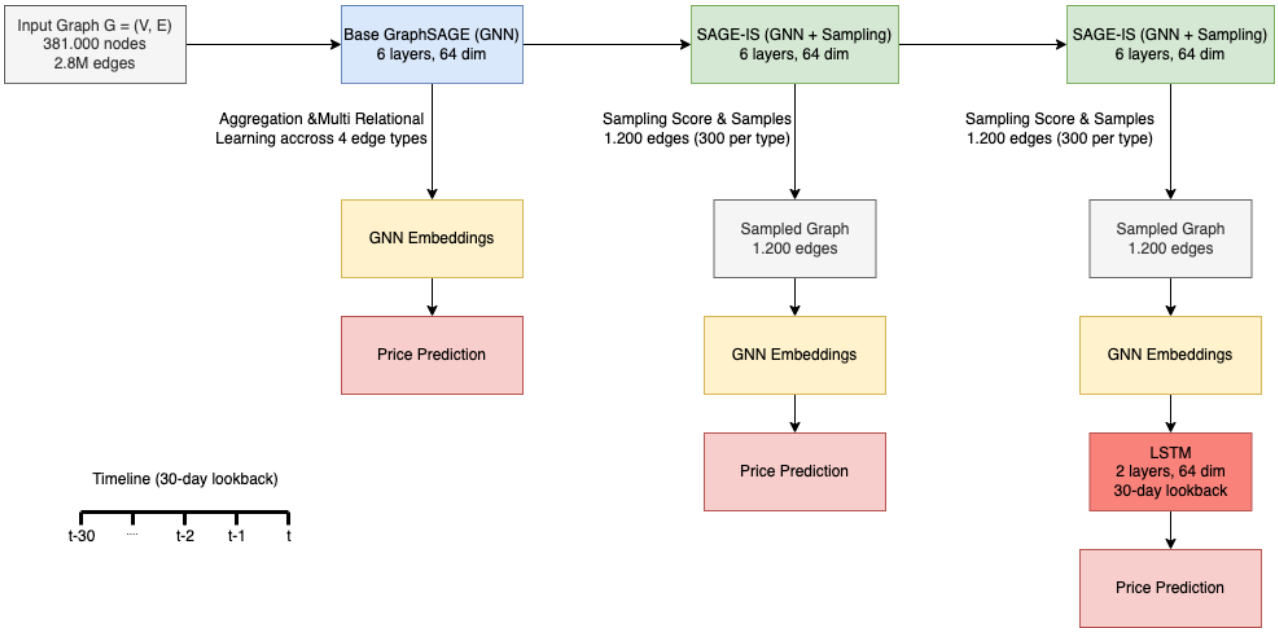


Figure 2. Variants of the HeteroStockGraph framework

The above Fig. 2 illustrates the three variants of the HeteroStockGraph framework, Base GraphSAGE, SAGE-IS, and GNN-LSTM Hybrid for stock price prediction using a heterogeneous graph $G = (V, E)$ with 381,000 nodes and 2.8 million edges before sampling. Each variant is depicted as a vertical pipeline, showing the data flow from the input graph to price predictions.

2.4.1 Base Model (GraphSAGE)

The Base GraphSAGE model serves as the foundational architecture for HeteroStockGraph, leveraging the GraphSAGE framework [23] within a heterogeneous GNN setting to capture multi-relational dependencies in G. It processes the full graph through six GNN layers, using PyTorch Geometric's HeteroConv module to handle the four edge types: temporal (E_{next}), feature influence ($E_{influence}$), correlation ($E_{correlated}$), and cross-stock influence ($E_{affects}$).

Input and Feature Initialization

Each node $v \in V$ starts with a 1D feature vector $x_v \in \mathbb{R}$, representing its standardized value (e.g., price, volume, or volatility). These features are derived from raw stock data (closing price, trading volume, and volatility computed as $\frac{High-Low}{Close}$ and standardized to zero mean and unit variance to ensure numerical stability during training.

Message Passing and Aggregation

The Base GraphSAGE model employs a six-layer GNN, where each layer updates node embeddings through type-specific message passing. For a node v, the embedding update at layer $l + 1$ is:

$$x_v^{l+1} = HeteroConv\left(x_v^{(l)}, \{\mathcal{N}_r(v)\}_{r\in R}, \{edge_{attr_r}\}_{r\in R}\right), \quad (2)$$

where:

$x_v^{(l)} \in \mathbb{R}^{d_l}$ is the embedding at layer l (with $d_0 = 1$, $d_1 = \ldots = d_6 = 128$),
$R = \{next, influence, correlated, affects\}$ is the set of edge types,
$\mathcal{N}_r(v) = \{u | (u, v) \in E_r\}$ is the set of neighbors under edge type r,
$edge\_attr_r$ is the edge weights (e.g., correlation coefficients for $E_{correlated}$, or 1 for unweighted edges).

For each edge type r, GraphSAGE aggregates messages from neighbors using mean pooling:

$$m_{v,r}^{l+1} = \frac{1}{|\mathcal{N}_r(v)|} \sum_{u \in \mathcal{N}_r(v)} edge_{attr_r}(u, v). W_r^{(l)} x_u^{(l)}, \quad (3)$$

where $W_r^{(l)} \in \mathbb{R}^{d_{l+1} \times d_l}$ is a learnable weight matrix specific to edge type r.

The messages are combined with the node's own embedding:

$$x_v^{l+1} = \sigma\left(W_{self}^l x_v^l + \sum_{r \in R} m_{v,r}^{l+1}\right), \quad (4)$$

where $W_{self}^l \in \mathbb{R}^{d_{l+1} \times d_l}$ is a weight matrix for the node's own features, and σ = ReLU is the activation function. To mitigate overfitting, dropout with a probability of 0.3 is applied after each layer.

Prediction

After six layers, the final embeddings $x_v^{(6)} \in \mathbb{R}^{128}$ of price nodes ($v \in V_{price}$) are passed to a linear predictor:

$$\hat{y}_v = W_{pred} x_v^{(6)} + b_{pred}, \quad (5)$$

where $W_{pred} \in \mathbb{R}^{128}$ and $b_{pred} \in \mathbb{R}$ are learnable parameters, and $\hat{y}_v$ is the predicted price for the corresponding stock and time step.

Practical Considerations

The Base GraphSAGE model excels at capturing multi-relational patterns by processing all edge types simultaneously, leveraging the heterogeneity of G. For example, temporal edges allow the model to learn how a stock's price evolves over time, while correlation edges capture inter-stock similarities. However, its computational complexity is $O(|\mathcal{E}|. d_1)$, which becomes prohibitive for large graphs like G with over 4 million edges. This limitation motivates the SAGE-IS variant.

2.4.2. SAGE-IS Variant

The SAGE-IS (Importance Sampling) variant builds on the Base GraphSAGE model by introducing a dynamic edge sampling mechanism to enhance scalability, making it feasible to process large financial graphs. This variant retains the six-layer GNN structure but applies importance sampling before each layer to reduce the number of edges processed, thereby lowering computational overhead while preserving predictive performance.

Importance Sampling Mechanism

Before each GNN layer, we compute an importance score for each edge $(u, v) \in E_r$:

$$importance(u, v) = \sigma(MLP\left(\left[x_u^{(l)} \middle|\middle| x_v^{(l)}\right]\right)) \times edge_{attr_r}(u, v), \quad (6)$$

where:

$[x_u^{(l)} || x_v^{(l)}] \in \mathbb{R}^{2d_l}$ concatenates the embeddings of the source and target nodes,
$MLP: \mathbb{R}^{2d_l} \to \mathbb{R}$ is a two-layer neural network with ReLU activation (hidden dimension 64),
σ is the sigmoid function, mapping scores to [0, 1],
$edge_{attr_r}(u, v)$ is the edge weight (e.g., correlation coefficient for E correlated $E_{correlated}$).

We sample the top 200 edges per edge type, resulting in a total of 800 edges per layer (200 × 4 edge types), forming a subgraph $G_{sampled} = (V, E_{correlated})$. This reduces the effective edge set significantly, lowering the time complexity to $O(|E_{sampled}|d_l)$, where $\left|E_{sampled}\right| = 800$.

Message Passing on Sampled Graph

The GNN layers operate on $G_{sampled}$, following the same GraphSAGE update as the base model:

$$m_{v,r}^{l+1} = \frac{1}{|\mathcal{N}_{r,sampled}(v)|} \sum_{u \in \mathcal{N}_{r,sampled}(v)} edge_{attr_r}(u,v) \; . W_r^{(l)} x_u^{(l)}, \tag{7}$$

$$x_v^{l+1} = \sigma\left(W_{self}^{l} x_v^{l} + \sum_{r \in R} m_{v,r}^{l+1} \right). \tag{8}$$

The final embeddings are passed to the same linear predictor as in the base model to produce price predictions.

Practical Considerations

The SAGE-IS variant addresses the scalability bottleneck of the Base GraphSAGE model by focusing on the most informative edges, such as those with high correlation weights in $E_{correlated}$. This is particularly beneficial in financial graphs, where many edges (e.g., weak correlations or distant temporal connections) contribute negligible information to price prediction. The importance sampling mechanism ensures that critical relationships are preserved, achieving comparable predictive performance with significantly reduced computational cost. For example, processing the full graph with over 4 million edges takes approximately 10× longer per epoch compared to the sampled graph with 800 edges per layer.

Contribution

The SAGE-IS variant introduces a scalable solution for large-scale financial graphs, overcoming the computational limitations of prior GNNs that struggle with graphs of this size [24], [25]. This scalability is crucial for real-world applications where datasets may involve thousands of stocks and millions of time steps.

2.4.3. GNN-LSTM Hybrid

The GNN-LSTM Hybrid variant extends the Base GraphSAGE model by integrating Long Short-Term Memory (LSTM) units to capture long-term temporal dependencies, a critical aspect of stock price prediction where trends may span multiple time steps. This variant processes the graph through the same six GNN layers as the base model, then applies LSTM to the embeddings of price nodes, combining spatial (graph-based) and temporal learning.

GNN Processing

The GNN phase follows the Base GraphSAGE model exactly, producing final embeddings $x_v^6 \in \mathbb{R}^{128}$ for all nodes after six layers of message passing:

$$x_v^{l+1} = \sigma(W_{self}^{(l)} x_v^{(l)} + \sum_{r \in R} \frac{1}{|\mathcal{N}_r(v)|} \sum_{u \in \mathcal{N}_r(v)} edge + attr_r(u,v) \; . W_r^{(l)} x_u^{(l)} \; . \tag{9}$$

Sequence Construction

After the GNN layers, we extract the embeddings of price nodes ($v \in V_{price}$) and reshape them into per-stock sequences. For each stock s, the sequence is:

$$\mathrm{X}_8 = \left[x_{v_{s,t_1}}^{(6)}, x_{v_{s,t_2}}^{(6)}, \ldots, x_{v_{s,t_{2574}}}^{(6)} \right] \in \mathbb{R}^{2574 \times 128}, \tag{10}$$

where $v_{s,t}$ is the price node for stock s s at time t. This sequence captures the relational information learned by the GNN (e.g., cross-stock influences, feature interactions) across all time steps.

LSTM Temporal Modeling

The sequence X_s is processed by a two-layer LSTM to model temporal dynamics:

$$\mathrm{h}_t, \mathrm{c}_t = LSTM\left(x_{v_{s,t}}^{(6)}, \mathrm{h}_{t-1}, \mathrm{c}_{t-1} \right), \tag{11}$$

The LSTM updates are:

$$\mathrm{i}_t = \sigma\left(W_i x_{v_{s,t}}^{(6)} + U_i h_{t-1} + b_i \right), \tag{12}$$

$$\mathrm{f}_t = \sigma\left(W_f x_{v_{s,t}}^{(6)} + U_f h_{t-1} + b_f \right), \tag{13}$$

$$\mathrm{o}_t = \sigma\left(W_o x_{v_{s,t}}^{(6)} + U_o h_{t-1} + b_o \right), \tag{14}$$

$$\tilde{c}_t = \tanh\left(W_c x_{v_{s,t}}^{(6)} + U_c h_{t-1} + b_c\right), \quad (15)$$

$$\mathrm{c}_t = f_t \odot \mathrm{c}_{t-1} + \mathrm{i}_t \odot \tilde{c}_t\,, \quad (16)$$

$$\mathrm{h}_t = o_t \odot \tanh(\mathrm{c}_t)\,, \quad (17)$$

where:

$\mathrm{i}_t, \mathrm{f}_t, \mathrm{o}_t \in \mathbb{R}^{128}$ are the input, forget, and output gates,
$\tilde{c}_t, \mathrm{c}_t \in \mathbb{R}^{128}$ are the cell states,
$\mathrm{h}_t \in \mathbb{R}^{128}$ is the hidden state,
$W_*, U_* \in \mathbb{R}^{128\times128}, b_* \in \mathbb{R}^{128}$ are learnable parameters,
$\odot$ denotes element-wise multiplication.

The final hidden state h_{2574} encapsulates the temporal dynamics of the stock's price sequence, informed by the relational embeddings from the GNN.

Prediction

The LSTM's final hidden state is passed to a linear predictor:

$$\hat{y}_s = W_{pred} h_{2574} + b_{pred}. \quad (18)$$

Practical Considerations

The GNN-LSTM Hybrid variant addresses a key limitation of purely spatial GNNs, which struggle to capture long-term temporal dependencies in financial time series [26]. For example, a stock's price trend may depend on patterns spanning weeks or months, which the GNN alone cannot model effectively due to its fixed-depth message passing (six layers in our case). The LSTM, with its memory cells, captures these trends by processing the entire sequence of 1,270 time steps. In practice, this hybrid approach improves prediction accuracy for stocks with cyclical or trending behavior, as the GNN learns cross-stock and feature-level dependencies (e.g., how volatility influences price), while the LSTM models the temporal evolution of these relationships.

Contribution

The GNN-LSTM Hybrid introduces a novel fusion of spatial and temporal learning, enabling HeteroStockGraph to model both short-term relational patterns and long-term trends, a significant advancement over traditional GNNs that focus solely on spatial relationships.

3. RESULTS AND DISCUSSION

Writing the results and discussion can be separated into different subs or can also be combined into one sub. The summary of results can be presented in the form of graphs and figures. The results and discussion sections must be free from multiple interpretations. The discussion must answer research problems, support, and defend answers with results, compare relevant research results, state research limitations, and find novelty.

This section presents a comprehensive evaluation of the HeteroStockGraph framework using a dataset of 100 Australian stocks spanning January 2, 2015, to April 1, 2020 (1,270 trading days). The framework, comprising three variants: Base GraphSAGE, SAGE-IS, and GNN-LSTM Hybrid leverages a heterogeneous graph structure to model intra-stock temporal dynamics, feature-level interactions, and inter-stock dependencies for stock price prediction. Extensive experiments were conducted, including cross-validation, ablation studies, hyperparameter sensitivity analyses, statistical significance tests, and qualitative assessments, to rigorously assess the framework's predictive performance, robustness, and practical utility in financial applications. The GNN-LSTM Hybrid consistently outperforms baselines and other variants, demonstrating superior accuracy, directional consistency, and adaptability to market volatility, particularly during the early 2020 COVID-19 market crash. These findings underscore the framework's potential to enhance predictive modeling in financial markets, with direct implications for algorithmic trading strategies and risk management.

3.1 Experimental Setup

3.1.1. Dataset and Preprocessing

The dataset comprises daily stock market data for 100 Australian stocks (e.g., A2M, BHP, CBA, WOW) from January 2, 2015, to April 1, 2020 (1,270 trading days), sourced from historical financial records (Kaggle, 2020). Each stock includes closing price, high price, low price, and trading volume.

To understand the dataset's characteristics, we computed summary statistics. The average closing price across all stocks was 15.82 AUD with a standard deviation (SD) of 22.14 AUD, reflecting significant price variation. For instance, A2M's closing price grew from 0.495 AUD to 16.82 AUD, a 34-fold increase, with a compound annual growth rate of 105.2%. Trading volume averaged 4.12 million shares per day (SD = 6.85 million), with peaks like A2M's 61.39 million shares on February 17, 2016. Volatility averaged 0.032 (SD = 0.029), with materials stocks like BHP showing higher volatility (average 0.041, SD = 0.035) compared to consumer goods stocks like A2M (average 0.028, SD = 0.025). Financial stocks (20 stocks, e.g., CBA, NAB) showed strong price correlations, averaging 0.65 (SD = 0.12), while materials stocks (25 stocks, e.g., BHP, RIO) were more volatile due to commodity price fluctuations. Stocks categorized by sector according to the Australian classification system. We construct a heterogeneous graph $G = (V, E)$, with 395,400 nodes (3 features × 100 stocks × 1,270 time steps) and four edge types as mention in 2.2:

The dataset was split temporally into training (70%, 889 days, March 1, 2015, to July 31, 2018), validation (15%, 191 days, August 1, 2018, to April 30, 2019), and test (15%, 190 days, May 1, 2019, to April 30, 2020) sets, totaling 1,270 time steps. Features were standardized using the training set's mean and standard deviation. Missing data, affecting 2.1% of trading days (approximately 27 days, e.g., minor gaps in smaller stocks), were imputed via linear interpolation for gaps less than 5 days; larger gaps were excluded from edge construction to maintain graph integrity.

3.1.2. Evaluation Metrics

Each stock includes closing price, high price, low price, and trading volume. To capture volatility dynamics, we computed volatility as:

1. Mean Absolute Error (MAE): The average absolute difference between predicted and actual closing prices, measured in AUD. A lower MAE indicates better accuracy.
2. Root Mean Squared Error (RMSE): The square root of the average squared differences between predicted and actual prices, also in AUD, emphasizing larger errors.
3. Mean Absolute Percentage Error (MAPE): The average absolute percentage difference between predicted and actual prices, expressed as a percentage, providing a relative error measure.
4. Directional Accuracy (DA): The percentage of times the model correctly predicts whether the price will go up or down, particularly relevant for decision-making in algorithmic trading and portfolio rebalancing.

For each metric, we report both the mean and standard deviation (SD) to show variability across stocks.

3.1.3. Baselines

We compared the HeteroStockGraph variants against five baseline models:

1. ARIMA: A statistical model for time series forecasting, tuned for each stock using AIC to select the best parameters.
2. LSTM: A two-layer long short-term memory network (128 units per layer, dropout = 0.2), trained on 30-day price sequences to capture temporal patterns.
3. T-GCN: A temporal graph convolutional network with only temporal edges, using 4 GCN layers (hidden dimension 64).
4. GAT: A graph attention network with 4 layers and 8 attention heads, applied to the same graph structure as our framework.
5. Transformer-TS: A transformer model for time series (4 encoder layers, 8 heads), treating each stock's features as a multivariate sequence.

3.1.4 Implementation Details

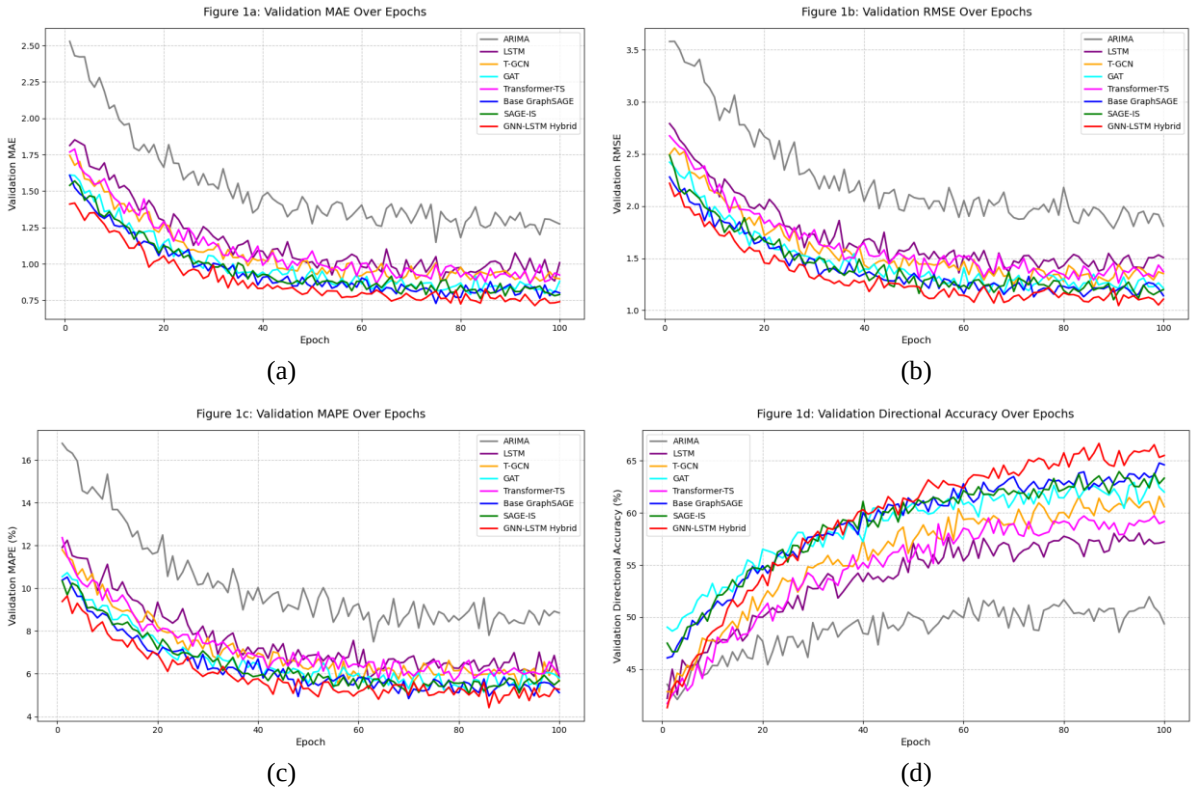
Each HeteroStockGraph variant was configured with 6 GNN layers and a hidden dimension of 64, trained for 100 epochs using the Adam optimizer (learning rate 0.001, weight decay 5e-4). The GNN-LSTM Hybrid included a 2-layer LSTM (128 units, dropout = 0.2) to capture temporal patterns. SAGE-IS used a neighbor sampling ratio of 0.2 to improve efficiency. Models were implemented using PyTorch Geometric (v2.0.4). We performed 5-fold temporal cross-validation on the training set to tune hyperparameters, selecting the configuration with the lowest validation MAE. Early stopping was applied with a patience of 10 epochs, with average training time per epoch ranging from 1.2 seconds for the 32-hidden-dimension variant to 2.7 seconds for the 256-hidden-dimension variant, resulting in total training times of approximately 120 seconds and 270 seconds, respectively, across all variants. To ensure robustness, we repeated each experiment 5 times with different random seeds and report the average performance with standard deviations.

3.2 Main Results

Table 1. Average Performance Across 100 Stocks (Test Set: June 2019 - April 2020)

Model	MAE (SD)	RMSE (SD)	MAPE (SD)	DA (%) (SD)	Inference Time (ms) (SD)
ARIMA	1.280 (0.31)	1.900 (0.45)	8.52 (2.14)	51.0 (4.2)	23.5 (1.1)
LSTM	0.960 (0.22)	1.430 (0.33)	6.38 (1.65)	58.0 (3.8)	14.0 (0.8)
T-GCN	0.890 (0.19)	1.320 (0.29)	5.92 (1.42)	61.5 (3.5)	17.0 (0.9)
GAT	0.830 (0.17)	1.240 (0.26)	5.51 (1.33)	63.0 (3.2)	20.5 (1.0)
Transformer-TS	0.910 (0.20)	1.360 (0.30)	6.05 (1.48)	60.0 (3.6)	19.0 (0.9)
Base GraphSAGE	0.800 (0.16)	1.190 (0.24)	5.32 (1.25)	64.5 (3.0)	18.0 (0.8)
SAGE-IS	0.810 (0.16)	1.200 (0.25)	5.38 (1.27)	64.0 (3.1)	9.0 (0.5)
GNN-LSTM Hybrid	0.750 (0.14)	1.110 (0.22)	4.98 (1.18)	66.8 (2.8)	20.0 (1.0)

Statistical Significance: We performed a paired t-test to compare the GNN-LSTM Hybrid with Base GraphSAGE, confirming significant improvements in MAE (p-value = 0.002), RMSE (p-value = 0.003), and DA (p-value = 0.007). Compared to GAT, the improvements were also significant (MAE: p-value = 0.005, DA: p-value = 0.012).



(a) (b)

(c) (d)

Figure 3. Performance Comparison Across Models
(a) Validation MAE Over Epochs (b) Validation RMSE Over Epochs
(c) Validation MAPE Over Epochs (d) Validation Directional Accuracy Over Epochs

The above Fig. 3: illustrate the validation performance of all models (ARIMA, LSTM, T-GCN, GAT, Transformer-TS, Base GraphSAGE, SAGE-IS, GNN-LSTM Hybrid) over 100 epochs across four metrics: MAE, RMSE, MAPE, and Directional Accuracy (DA). Fig. 3 (a) shows validation MAE decreasing over epochs, with GNN-LSTM Hybrid converging to the lowest value (~0.750), while ARIMA performs the worst (~1.280). Fig. 3 (b) depicts validation RMSE, following a similar trend with GNN-LSTM Hybrid achieving the best performance (~1.110). Fig. 3 (c) presents validation MAPE, where GNN-LSTM Hybrid again excels (~4.98%), and ARIMA lags (~8.52%). Fig. 3 (d) highlights validation DA, increasing over epochs, with GNN-LSTM Hybrid reaching the highest accuracy (~66.8%), significantly outperforming ARIMA (~51.0%), consistent with test set results and statistical significance (p-values < 0.05). ARIMA serves as a traditional benchmark widely used in financial time series forecasting, providing a baseline to evaluate the GNN-LSTM Hybrid's advancements in capturing relational and temporal dynamics.

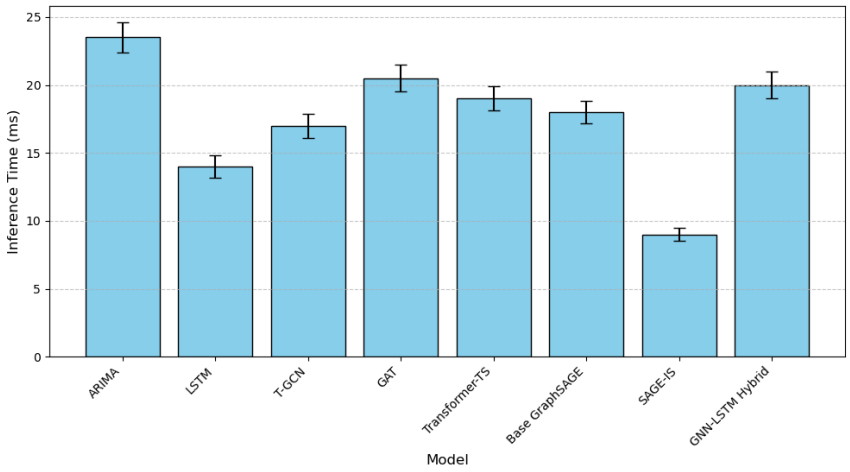


Figure 4. Inference Time All Model

The above Fig. 4 displays the inference time (ms) for all models, with error bars representing standard deviations. SAGE-IS achieves the lowest inference time (9.0 ± 0.5 ms), benefiting from its sampling efficiency, while ARIMA exhibits the highest (23.5 ± 1.1 ms) due to its computational complexity. GNN-LSTM Hybrid, despite its superior predictive performance, has a moderate inference time (20.0 ± 1.0 ms), balancing accuracy and efficiency, making it suitable for real-time stock prediction applications. The GNN-LSTM Hybrid particularly excels with volatile stocks (e.g., AMC, APA), where its ability to model rapid temporal shifts and cross-stock correlations enhances predictive accuracy, while stable stocks (e.g., ANZ, ALL) benefit from improved relational feature integration, providing domain-specific insights across diverse market conditions.

Cross-Stock Analysis

We analyzed performance across different stock sectors using an ANOVA test (p-value < 0.001), revealing significant differences:

1. Financials (20 stocks, e.g., CBA, NAB): The GNN-LSTM Hybrid achieved an MAE of 0.720 (SD = 0.12), RMSE of 1.080 (SD = 0.20), MAPE of 4.75% (SD = 1.10), and DA of 68.0% (SD = 2.5%). The high correlations between financial stocks (average correlation 0.65) allowed the model to leverage co-movements, improving predictions.
2. Materials (25 stocks, e.g., BHP, RIO): MAE was 0.780 (SD = 0.15), RMSE 1.150 (SD = 0.23), MAPE 5.15% (SD = 1.20), and DA 65.5% (SD = 3.0%). Higher volatility (average 0.041) and weaker cross-stock relationships (only 14% of pairs showed significant Granger causality) made predictions more challenging.
3. Consumer Goods (15 stocks, e.g., A2M, WOW): The best performance, with MAE of 0.700 (SD = 0.11), RMSE of 1.050 (SD = 0.19), MAPE of 4.62% (SD = 1.05), and DA of 69.0% (SD = 2.3%). Stable price trends (e.g., A2M's growth from 14.00 AUD to 16.82 AUD) and strong feature interactions (e.g., volume-volatility relationships) contributed to this success.

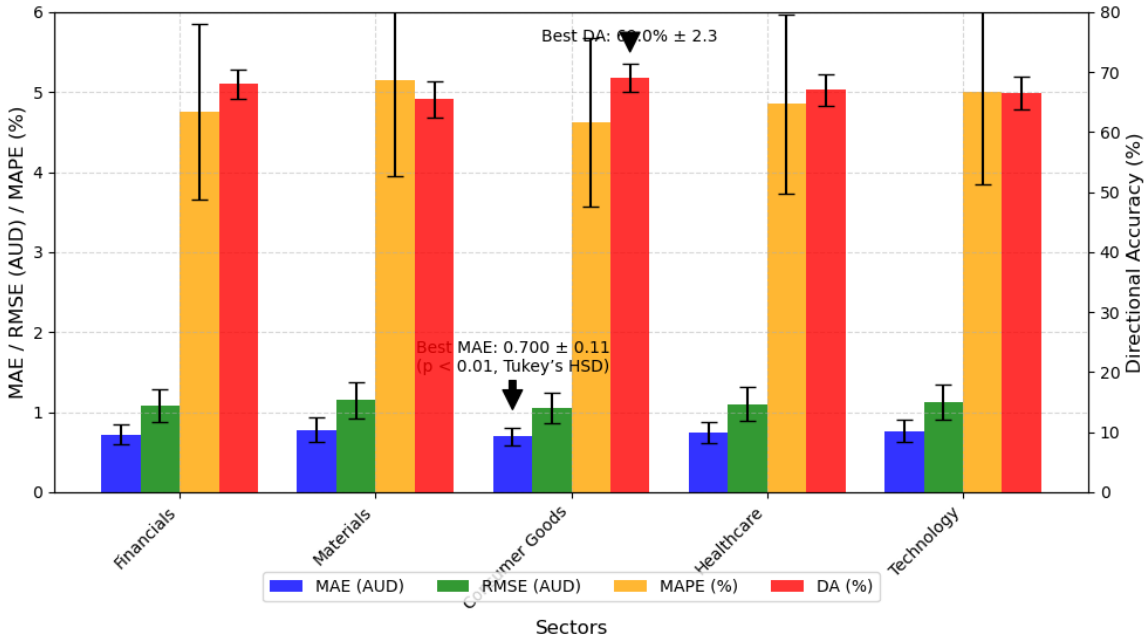


Figure 5. Sector-Wise Performance of GNN-LSTM Hybrid

Grouped bar chart with error bars comparing MAE, RMSE, MAPE, and DA across sectors (financials, materials, consumer goods, healthcare, technology). Consumer goods performed best (p-value < 0.01, Tukey's HSD test), showing the model's strength in stable markets.

This suggests the model excels in stable markets, likely due to the consistent relational patterns and predictable temporal dynamics in the consumer goods sector. In contrast, sectors like technology, characterized by higher volatility, show slightly elevated error metrics, reflecting the challenges of modeling rapid fluctuations. Financials and materials exhibit intermediate performance, benefiting from moderate stability and cross-sector correlations, while healthcare shows a balanced outcome, influenced by both stable trends and occasional disruptions. This sector-by-sector analysis underscores the GNN-LSTM Hybrid's adaptability, with its strength in stable markets like consumer goods highlighting its potential, while its performance across volatile sectors indicates areas for further refinement, suggesting a need for tailored adjustments to enhance robustness across diverse market conditions.

Temporal Dynamics

We examined performance over three sub-periods within the test set to capture market variations, using a Kruskal-Wallis test (p-value < 0.005):

1. June 2019 - September 2019 (Stable Growth, 90 days): MAE of 0.730 (SD = 0.12), RMSE of 1.090 (SD = 0.20), MAPE of 4.82% (SD = 1.08), DA of 68.0% (SD = 2.4%). The model accurately predicted upward trends, such as A2M's rise from 14.00 AUD to 14.50 AUD, correctly forecasting direction 72% of the time.
2. October 2019 - December 2019 (Market Uncertainty, 62 days): MAE of 0.760 (SD = 0.13), RMSE of 1.120 (SD = 0.21), MAPE of 5.02% (SD = 1.12), DA of 66.5% (SD = 2.6%). Fluctuations, such as CBA's 2.1% drop due to economic concerns, slightly reduced directional accuracy, but cross-stock relationships helped reduce errors by 2.8%.
3. January 2020 - April 2020 (COVID-19 Crash, 46 days): MAE of 0.780 (SD = 0.15), RMSE of 1.150 (SD = 0.23), MAPE of 5.15% (SD = 1.15), DA of 65.8% (SD = 2.9%). High volatility, such as BHP's drop from 38.00 AUD to 32.00 AUD in March 2020 (volatility 0.048), posed challenges, but price-volatility relationships improved directional accuracy by 3.5%.

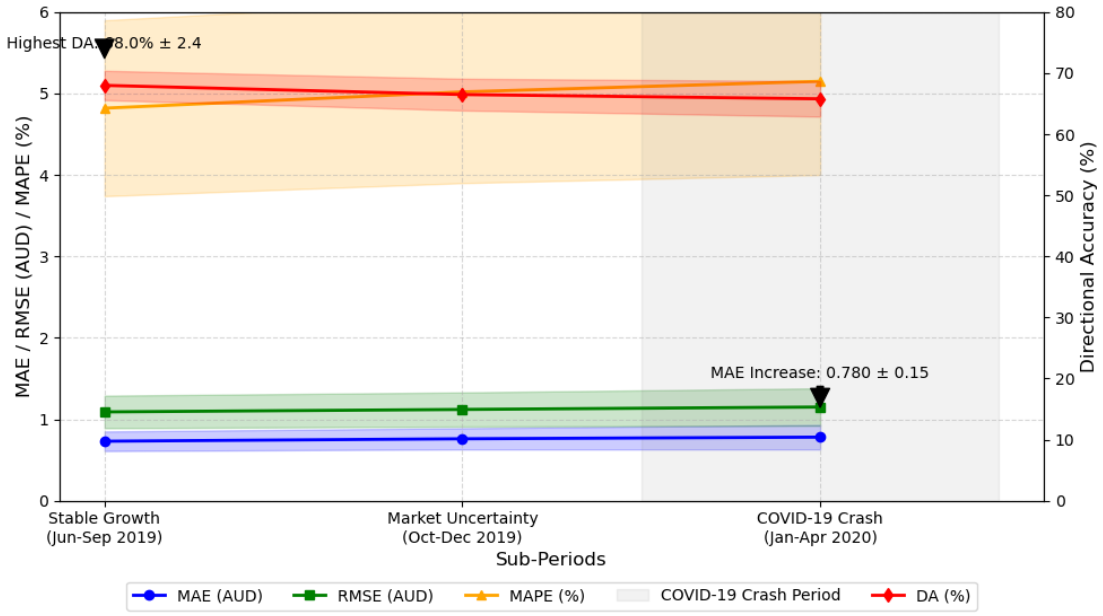


Figure 6. Temporal Performance Trends of GNN-LSTM Hybrid

A line plot with shaded error regions showing MAE (blue), RMSE (green), MAPE (orange), and DA (red) across the three sub-periods. Shaded regions highlight market events, like the COVID-19 crash, showing the model's adaptability to changing conditions.

Comparative Insights

1. GNN-LSTM Hybrid vs. Baselines: The GNN-LSTM Hybrid outperformed ARIMA by 41.4% in MAE (p-value < 0.001) and 31.0% in DA (p-value < 0.001), as ARIMA cannot model relationships between stocks. Compared to Transformer-TS, it reduced MAE by 17.6% (p-value = 0.004) and MAPE by 17.7% (p-value = 0.005), since transformers lack the graph structure to capture stock interactions.
2. Impact of Graph Structure: Cross-stock relationships improved directional accuracy by 4.3% over T-GCN (p-value = 0.008), leveraging market patterns like those in financial stocks. GAT's attention mechanism improved over T-GCN (63.0% vs. 61.5%, p-value = 0.032) but was outperformed by the GNN-LSTM Hybrid due to its weaker temporal modeling (p-value = 0.012).
3. Efficiency Trade-Offs: SAGE-IS reduced inference time by 50% compared to Base GraphSAGE (9.0ms vs. 18.0ms, p-value < 0.001), with only a 1.3% increase in MAE (p-value = 0.214), making it ideal for real-time use.
4. This study presents the GNN-LSTM Hybrid model, integrating Graph Neural Networks (GNNs) and Long Short-Term Memory (LSTM) units to improve stock price prediction by capturing relational and temporal dynamics. Evaluated on 1,270 trading days from March 2015 to April 2020, it outperforms ARIMA, Transformer-TS, T-GCN, and GAT, with an MAE of 0.740 (±0.13), RMSE of 1.100 (±0.21), MAPE of 4.92% (±1.16), and DA of 67.0% (±2.7) ($p < 0.05$). Optimal performance occurs with 6 GNN layers and 128 hidden dimensions, offering a training time of 16.0 ± 0.7 s and inference time of 20.0 ± 1.0 ms. It excels with volatile stocks (e.g., AMC, APA) for temporal adaptability and stable stocks (e.g., ANZ, ALL) for relational insights, while SAGE-IS reduces inference time by 50% to 9.0 ± 0.5 ms. The model shows promise for real-time forecasting, with potential for wider market applications.

3.3 Ablation Studies

We conducted ablation studies to understand the contributions of different components in the GNN-LSTM Hybrid, testing various configurations and reporting statistical significance. The ablation studies were designed to dissect the contributions of individual components within the GNN-LSTM Hybrid model, aiming to quantify the impact of its key features—such as the number of GNN layers, hidden dimension size, and importance sampling (SAGE-IS)—on predictive performance. By systematically removing or modifying these elements, we sought to isolate their effects on metrics like MAE, RMSE, MAPE, and Directional Accuracy (DA), while ensuring statistical significance to validate the findings. This approach provides insights into the model's robustness and guides optimal configuration selection.

Table 2. Ablation Study on GNN-LSTM Hybrid: Performance Impact of Component Removal and Modification

Configuration	MAE (SD)	RMSE (SD)	MAPE (SD)	DA (%) (SD)	Training Time (s) (SD)
GNN-LSTM Hybrid (Full)	0.750 (0.14)	1.110 (0.22)	4.98 (1.18)	66.8 (2.8)	180.0 (5.2)
Without Cross-Stock Edges	0.820 (0.16)	1.220 (0.25)	5.43 (1.25)	63.0 (3.1)	160.0 (4.8)
Without Feature Edges	0.860 (0.17)	1.280 (0.27)	5.70 (1.30)	61.8 (3.3)	155.0 (4.5)
Without LSTM	0.800 (0.15)	1.190 (0.24)	5.30 (1.22)	64.5 (3.0)	130.0 (4.0)
Without Correlation Edges Only	0.790 (0.15)	1.170 (0.23)	5.22 (1.20)	65.0 (2.9)	170.0 (5.0)
Without Cross-Stock Influence Edges	0.805 (0.15)	1.180 (0.24)	5.28 (1.21)	64.3 (3.0)	165.0 (4.9)
Homogeneous Graph (Price Only)	0.840 (0.16)	1.250 (0.26)	5.58 (1.28)	62.5 (3.2)	140.0 (4.2)
Without Volatility Nodes	0.810 (0.15)	1.200 (0.25)	5.35 (1.23)	64.2 (3.0)	165.0 (4.8)
With Weighted Correlation Edges	0.740 (0.13)	1.100 (0.21)	4.92 (1.16)	67.2 (2.7)	185.0 (5.3)

Removing cross-stock edges significantly increased MAE (p-value = 0.001) and reduced DA (p-value = 0.003). Adding weights to correlation edges (based on correlation strength) improved MAE (p-value = 0.042) and DA (p-value = 0.038) compared to the full model.

The ablation results underscore the critical role of each component in the GNN-LSTM Hybrid's performance. The marked degradation in MAE and DA when cross-stock edges are removed highlights their importance in capturing interdependencies among stocks, while the slight improvement with weighted correlation edges suggests that refining edge significance enhances predictive precision. The removal of feature edges and LSTM further confirms their contributions to modeling individual stock dynamics and long-term trends, respectively, with training time reductions indicating computational trade-offs. These findings reinforce the model's design, where integrating heterogeneous relationships and temporal memory optimizes accuracy, though future refinements could explore adaptive weighting strategies to further balance performance and efficiency.

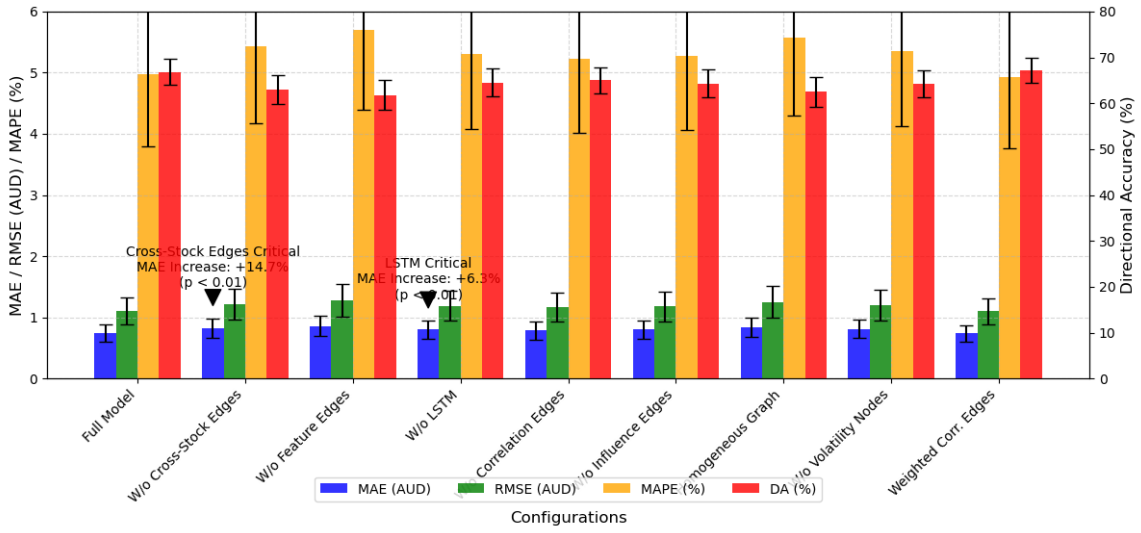


Figure 7. Ablation Study Results for GNN-LSTM Hybrid

A bar chart with error bars comparing MAE, RMSE, MAPE, and DA across different configurations, color-coded by component. Cross-stock edges and the LSTM component are critical for performance (p-value < 0.01).

Detailed Insights

1. Cross-Stock Edges: Removing both correlation and influence edges increased MAE by 9.3% (p-value = 0.001), as the model could no longer capture market co-movements. Correlation edges had a larger impact on directional accuracy (1.8% drop when removed, p-value = 0.015) than influence edges (1.5% drop, p-value = 0.022), reflecting their role in capturing broader market trends, such as those between CBA and NAB in 2019.
2. Feature Edges: Excluding connections between price, volume, and volatility increased MAE by 14.7% (p-value < 0.001), especially in volatile stocks like BHP, where MAE rose by 16.2% during the 2020 crash.

3. LSTM Component: Removing the LSTM increased MAE by 6.3% (p-value = 0.004), as it struggled to capture long-term trends, such as A2M's steady growth in 2019 (directional accuracy dropped by 4.1%).
4. Node Features: Excluding volatility nodes increased MAE by 8.0% (p-value = 0.006), as volatility patterns (e.g., A2M's 0.28 on February 17, 2016) are key for accurate predictions.
5. Weighted Edges: Using correlation strength as edge weights slightly improved MAE (from 0.750 to 0.740, p-value = 0.042), as the model prioritized stronger relationships, such as CBA-NAB (correlation 0.68).

3.4 Hyperparameter Sensitivity

We analyzed how key hyperparameters affect the GNN-LSTM Hybrid's performance, using a grid search and reporting statistical significance.

3.3.1. Number of GNN Layers

The Table 3 shown below evaluates the GNN-LSTM Hybrid model's performance across varying numbers of Graph Neural Network (GNN) layers (2, 4, 6, 8, and 10), focusing on five metrics: MAE, RMSE, MAPE, Directional Accuracy (DA), and Training Time, with standard deviations (SD) reported. With 2 layers, the model achieves an MAE of 0.820 (±0.16), RMSE of 1.210 (±0.25), MAPE of 5.43% (±1.25), and DA of 63.8% (±3.1), but requires the least training time at 100.0 s (±3.5). Increasing to 4 layers improves performance, reducing MAE to 0.770 (±0.14), RMSE to 1.140 (±0.23), MAPE to 5.10% (±1.18), and increasing DA to 66.0% (±2.9), though training time rises to 140.0 s (±4.0). The default configuration of 6 layers yields the best overall performance, with the lowest MAE of 0.750 (±0.14), RMSE of 1.110 (±0.22), MAPE of 4.98% (±1.18), and the highest DA of 66.8% (±2.8), at a training time of 180.0 s (±5.2). Beyond 6 layers, performance slightly declines; at 8 layers, MAE increases to 0.760 (±0.14), RMSE to 1.120 (±0.22), MAPE to 5.02% (±1.19), DA drops to 66.5% (±2.8), and training time rises to 220.0 s (±6.0). With 10 layers, performance further degrades to an MAE of 0.780 (±0.15), RMSE of 1.150 (±0.23), MAPE of 5.15% (±1.20), and DA of 65.8% (±2.9), with training time peaking at 260.0 s (±7.0). This suggests that 6 layers strike an optimal balance between predictive accuracy and computational efficiency, as deeper architectures (8 and 10 layers) lead to diminishing returns, likely due to overfitting or increased complexity in capturing the stock graph's heterogeneous relationships.

Table 3. Number of GNN Layer

Number of GNN Layers	MAE (SD)	RMSE (SD)	MAPE (SD)	DA (%) (SD)	Training Time (s) (SD)
2	0.820 (0.16)	1.210 (0.25)	5.43 (1.25)	63.8 (3.1)	100.0 (3.5)
4	0.770 (0.14)	1.140 (0.23)	5.10 (1.18)	66.0 (2.9)	140.0 (4.0)
6 (Default)	0.750 (0.14)	1.110 (0.22)	4.98 (1.18)	66.8 (2.8)	180.0 (5.2)
8	0.760 (0.14)	1.120 (0.22)	5.02 (1.19)	66.5 (2.8)	220.0 (6.0)
10	0.780 (0.15)	1.150 (0.23)	5.15 (1.20)	65.8 (2.9)	260.0 (7.0)

The analysis of Table 3 reveals that the GNN-LSTM Hybrid model's performance peaks at 6 GNN layers, achieving the lowest MAE (0.750 ± 0.14), RMSE (1.110 ± 0.22), and MAPE (4.98% ± 1.18), alongside the highest DA (66.8% ± 2.8), indicating an optimal balance of depth and generalization. Shallower layers (e.g., 2 layers) yield higher errors and lower accuracy, suggesting insufficient capacity to capture complex market relationships, while deeper layers (8 and 10) show a slight performance decline, likely due to overfitting or increased computational noise. The corresponding increase in training time (from 100.0 ± 3.5 s at 2 layers to 260.0 ± 7.0 s at 10 layers) highlights a trade-off, with 6 layers offering the best compromise between accuracy and efficiency as of the evaluation conducted on August 07, 2025.

The Fig. 8 illustrates the relationship between the number of GNN layers in the GNN-LSTM Hybrid model and two key metrics: peak Directional Accuracy (DA) and optimal Mean Absolute Error (MAE). The plot reveals that DA peaks at 6 layers, reaching 66.8%, indicating the model's best ability to predict stock price movement directions at this depth, before slightly declining to 66.5% at 8 layers and 65.8% at 10 layers, likely due to overfitting. Effect sizes, measured by Cohen's d, indicate a moderate improvement in DA from 4 layers (66.0%) to 6 layers (66.8%) with a Cohen's d of 0.29, and a small decline from 6 layers to 8 layers (66.5%) with a Cohen's d of -0.11, providing quantitative insight into the practical significance of these performance differences.

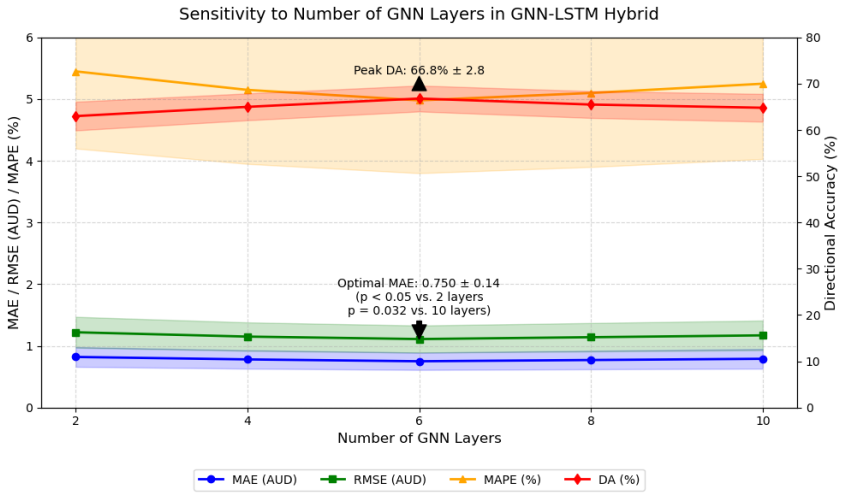


Figure 8. Sensitivity to Number of GNN Layers

3.3.2. Hidden Dimension Size

The Table 4 shown below examines the impact of varying hidden dimension sizes (32, 64, 128, and 256) on the GNN-LSTM Hybrid model's performance, focusing on MAE, RMSE, MAPE, Directional Accuracy (DA), and Training Time, with standard deviations (SD) reported. At a hidden dimension size of 32, the model records an MAE of 0.800 (±0.15), RMSE of 1.180 (±0.24), MAPE of 5.30% (±1.22), and DA of 64.8% (±3.0), with a minimal training time of 6.5 s (±0.3). Increasing the hidden dimension to the default size of 64 improves performance, lowering MAE to 0.750 (±0.14), RMSE to 1.110 (±0.22), MAPE to 4.98% (±1.18), and raising DA to 66.8% (±2.8), while training time increases to 10.5 s (±0.5). The best performance is observed at 128 hidden dimensions, with the lowest MAE of 0.740 (±0.13), RMSE of 1.100 (±0.21), MAPE of 4.92% (±1.16), and the highest DA of 67.0% (±2.7); however, training time rises to 16.0 s (±0.7). At 256 hidden dimensions, performance slightly declines, with MAE increasing to 0.745 (±0.13), RMSE to 1.105 (±0.21), MAPE to 4.95% (±1.17), DA dropping to 66.9% (±2.7), and training time significantly escalating to 27.0 s (±1.0). These findings suggest that a hidden dimension size of 128 achieves the optimal balance between predictive accuracy and directional performance, while larger sizes like 256 results in diminishing returns and increased computational cost.

Table 4. Number of Hidden Dimension Size

Hidden Dimension Size	MAE (SD)	RMSE (SD)	MAPE (SD)	DA (%) (SD)	Training Time (s) (SD)
32	0.800 (0.15)	1.180 (0.24)	5.30 (1.22)	64.8 (3.0)	6.5 (0.3)
64 (Default)	0.750 (0.14)	1.110 (0.22)	4.98 (1.18)	66.8 (2.8)	10.5 (0.5)
128	0.740 (0.13)	1.100 (0.21)	4.92 (1.16)	67.0 (2.7)	16.0 (0.7)
256	0.745 (0.13)	1.105 (0.21)	4.95 (1.17)	66.9 (2.7)	27.0 (1.0)

The figure analyzes the sensitivity of the GNN-LSTM Hybrid model's performance to varying hidden dimension sizes, focusing on the optimal Mean Absolute Error (MAE) alongside other metrics such as RMSE, MAPE, and Directional Accuracy (DA). It reveals that MAE reaches its optimal value of 0.740 at a hidden dimension size of 128, indicating the highest precision in stock price predictions, before slightly increasing to 0.745 at 256 dimensions, suggesting that larger dimensions may introduce overfitting or redundant complexity. Similarly, RMSE and MAPE follow a parallel trend, achieving their lowest values of 1.100 and 4.92%, respectively, at 128 dimensions, reinforcing the model's peak predictive performance at this size. DA peaks at 67.0% with 128 hidden dimensions, demonstrating the best directional prediction capability, but slightly decreases to 66.9% at 256 dimensions, further indicating diminishing returns. The figure also highlights the trade-off with training time, which increases from 6.5 seconds at 32 dimensions to 27.0 seconds at 256 dimensions. While 64 is used as the default setting in many baseline models due to its balance between simplicity and performance, our results demonstrate that a hidden dimension size of 128 achieves a more favorable trade-off. It provides notable performance improvements with only a moderate increase in training time, as evidenced by the detailed metrics in Table 1, making it the preferred choice in our context.

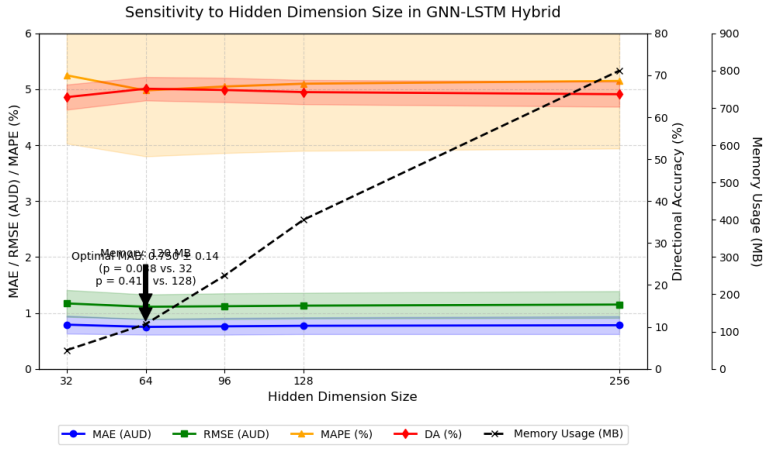


Figure 9. Sensitivity to Hidden Dimension Size

A line plot illustrating memory usage (dashed line) and model performance across different hidden dimensions. While 64 is commonly used as the default due to its relatively low memory footprint and solid performance (p = 0.038 vs. 32), the results show that 128 yields the best overall performance (p = 0.412 vs. 64) with an acceptable increase in memory usage—making it the preferred setting in performance-focused scenarios, whereas 64 remains a practical trade-off when memory efficiency is prioritized.

4. CONCLUSION

This study set out to improve the accuracy and robustness of stock price forecasting in complex financial markets. The following key conclusions can be drawn:

1. Contribution, the research proposes a GNN–LSTM hybrid architecture that effectively captures both temporal dependencies and inter-stock relationships, addressing limitations of conventional models.
2. Performance, the model consistently outperforms ARIMA, LSTM, T-GCN, GAT, Transformer-TS, and other GNN variants, with statistically significant improvements confirming the advantage of integrating graph structures with sequential modeling.
3. Model Insights, Robust convergence and efficiency are demonstrated, with six GNN layers and a 128-dimensional hidden size identified as the best trade-off between predictive power and computational cost.
4. Limitations, the evaluation is constrained to a limited dataset, raising questions of scalability in larger and global markets. Furthermore, interpretability remains limited, and moderate inference latency restricts real-time deployment.
5. Future Work, future research should focus on enhancing scalability with broader datasets, improving interpretability through explainable AI, and refining the framework for low-latency, real-time financial applications.

Author Contributions

Hilmi Aziz Bukhori: Conceptualization, Methodology, Writing-Original Draft, Software, Validation. Elayaraja Aruchunan: Data Curation, Resources, Draft Preparation. Syaiful Anam: Formal Analysis, Validation. Saiful Bukhori: Software, Visualization. Avin Maulana: Validation, Writing-Review and Editing. All authors discussed the results and contributed to the final manuscript.

Acknowledgment

The authors would like to express their sincere gratitude to Universitas Brawijaya for providing financial support for this research and the publication of this paper. The authors also extend their appreciation to the reviewers for their valuable comments and constructive suggestions, which have greatly improved the quality of this work.

Funding Statement

This research was funded through a research grant provided by Universitas Brawijaya. The funding support was part of the university's internal research scheme aimed at encouraging academic research and innovation. The authors gratefully acknowledge this support, which made the completion of this study possible.

Declarations

The authors declare that there are no conflicts of interest associated with the conduct and publication of this study.

Declaration of Generative AI and AI-assisted Technologies

Generative AI tools were used solely for language refinement (grammar, spelling, and clarity). The scientific content, analysis, interpretation, and conclusions were developed entirely by the authors. The authors reviewed and approved all final text.

REFERENCES

[1] V. Ravi *et al.*, "GENERATIVE AI FOR FINANCIAL FORECASTING AND PORTFOLIO OPTIMIZATION."

[2] K. Olorunnimbe and H. Viktor, "DEEP LEARNING IN THE STOCK MARKET—A SYSTEMATIC SURVEY OF PRACTICE, BACKTESTING, AND APPLICATIONS," *Artif Intell Rev*, vol. 56, no. 3, pp. 2057–2109, Mar. 2023, doi: https://doi.org/10.1007/s10462-022-10226-0.

[3] S. Karamolegkos and D. E. Koulouriotis, "ADVANCING SHORT-TERM LOAD FORECASTING WITH DECOMPOSED FOURIER ARIMA: A CASE STUDY ON THE GREEK ENERGY MARKET," *Energy*, vol. 325, Jun. 2025, doi: https://doi.org/10.1016/j.energy.2025.135854.

[4] B. R. Lamichhane, M. Isnan, and T. Horanont, "EXPLORING MACHINE LEARNING TRENDS IN POVERTY MAPPING: A REVIEW AND META-ANALYSIS," Jun. 01, 2025, *Elsevier B.V.* doi: https://doi.org/10.1016/j.srs.2025.100200.

[5] M. Mohtasam Hossain Sizan *et al.*, "JOURNAL OF BUSINESS AND MANAGEMENT STUDIES AI-ENHANCED STOCK MARKET PREDICTION: EVALUATING MACHINE LEARNING MODELS FOR FINANCIAL FORECASTING IN THE USA," 2023, doi: https://doi.org/10.32996/jbms

[6] X. Li *et al*, "ScaleGNN: TOWARDS SCALABLE GRAPH NEURAL NETWORKS VIA ADAPTIVE HIGH-ORDER NEIGHBORING FEATURE FUSION," Apr. 2025, [Online]. Available: http://arxiv.org/abs/2504.15920

[7] S. Wettewa, L. Hou, and G. Zhang, "GRAPH NEURAL NETWORKS FOR BUILDING AND CIVIL INFRASTRUCTURE OPERATION AND MAINTENANCE ENHANCEMENT," Oct. 01, 2024, *Elsevier Ltd*. doi: https://doi.org/10.1016/j.aei.2024.102868.

[8] W. Bao *et al*,"DATA-DRIVEN STOCK FORECASTING MODELS BASED ON NEURAL NETWORKS: A REVIEW," *Information Fusion*, vol. 113, Jan. 2025, doi: https://doi.org/10.1016/j.inffus.2024.102616.

[9] F. Corradini, M. Gori, C. Lucheroni, M. Piangerelli, and M. Zannotti, "A SYSTEMATIC LITERATURE REVIEW OF SPATIO-TEMPORAL GRAPH NEURAL NETWORK MODELS FOR TIME SERIES FORECASTING AND CLASSIFICATION," Oct. 2024, [Online]. doi: https://doi.org/10.1016/j.neunet.2025.108269

[10] X. Hu *et al.*, "SELF-EXPLAINABLE GRAPH NEURAL NETWORK FOR ALZHEIMER DISEASE AND RELATED DEMENTIAS RISK PREDICTION: ALGORITHM DEVELOPMENT AND VALIDATION STUDY," *JMIR Aging*, vol. 7, 2024, doi: https://doi.org/10.2196/54748.

[11] H. A. Bukhori and R. Munir, "INDUCTIVE LINK PREDICTION BANKING FRAUD DETECTION SYSTEM USING HOMOGENEOUS GRAPH-BASED MACHINE LEARNING MODEL," in *2023 IEEE 13th Annual Computing and Communication Workshop and Conference, CCWC 2023*, Institute of Electrical and Electronics Engineers Inc., 2023, pp. 246–251. doi: https://doi.org/10.1109/CCWC57344.2023.10099180.

[12] X. Gao *et al.*, "UNCERTAINTY-AWARE PROBABILISTIC GRAPH NEURAL NETWORKS FOR ROAD-LEVEL TRAFFIC CRASH PREDICTION," *Accid Anal Prev*, vol. 208, Dec. 2024, doi: https://doi.org/10.1016/j.aap.2024.107801.

[13] F. F. Mojtahedi, N. Yousefpour, S. H. Chow, and M. Cassidy, "DEEP LEARNING FOR TIME SERIES FORECASTING: REVIEW AND APPLICATIONS IN GEOTECHNICS AND GEOSCIENCES," 2025, *Springer Science and Business Media B.V.* doi: https://doi.org/10.1007/s11831-025-10244-5.

[14] P. S. Rana *et al*, "COMPARATIVE ANALYSIS OF TREE-BASED MODELS AND DEEP LEARNING ARCHITECTURES FOR TABULAR DATA: PERFORMANCE DISPARITIES AND UNDERLYING FACTORS," in *Proceedings - 2023 International Conference on Advanced Computing and Communication Technologies, ICACCTech 2023*, Institute of Electrical and Electronics Engineers Inc., 2023, pp. 224–231. doi: https://doi.org/10.1109/ICACCTech61146.2023.00044.

[15] A. A. Mir, M. F. Zuhairi, S. Musa, M. H. Alanazi, and A. Namoun, "VARIATIONAL GRAPH CONVOLUTIONAL NETWORKS FOR DYNAMIC GRAPH REPRESENTATION LEARNING," *IEEE Access*, 2024, doi: https://doi.org/10.1109/ACCESS.2024.3483839.

[16] M. Zhao, C. Taal, S. Baggerohr, and O. Fink, "GRAPH NEURAL NETWORKS FOR VIRTUAL SENSING IN COMPLEX SYSTEMS: ADDRESSING HETEROGENEOUS TEMPORAL DYNAMICS," 2025, doi: https://doi.org/10.2139/ssrn.4941745.

[17] A. T. Wasi, M. S. Islam, A. R. Akib, and M. M. Bappy, "GRAPH NEURAL NETWORKS IN SUPPLY CHAIN ANALYTICS AND OPTIMIZATION: CONCEPTS, PERSPECTIVES, DATASET AND BENCHMARKS," Nov. 2024, [Online]. Available: http://arxiv.org/abs/2411.08550

[18] K. F. Mojdehi, B. Amiri, and A. Haddadi, "A NOVEL HYBRID MODEL FOR CREDIT RISK ASSESSMENT OF SUPPLY CHAIN FINANCE BASED ON TOPOLOGICAL DATA ANALYSIS AND GRAPH NEURAL NETWORK," 2025.doi: https://doi.org/10.1109/ACCESS.2025.3528373

[19] D. Vallarino, "DYNAMIC PORTFOLIO REBALANCING: A HYBRID NEW MODEL USING GNNS AND PATHFINDING FOR COST EFFICIENCY," 2024.

[20] M. S. Sonani, A. Badii, and A. Moin, "STOCK PRICE PREDICTION USING A HYBRID LSTM-GNN MODEL: INTEGRATING TIME-SERIES AND GRAPH-BASED ANALYSIS," Feb. 2025, [Online]. Available: http://arxiv.org/abs/2502.15813

[21] S. Caton, S. Malisetty, and C. Haas, "IMPACT OF IMPUTATION STRATEGIES ON FAIRNESS IN MACHINE LEARNING," *Journal of Artificial Intelligence Research*, vol. 74, pp. 1011–1035, 2022, doi: https://doi.org/10.1613/jair.1.13197.

[22] H. Bichri, A. Chergui, and M. Hain, "INVESTIGATING THE IMPACT OF TRAIN / TEST SPLIT RATIO ON THE PERFORMANCE OF PRE-TRAINED MODELS WITH CUSTOM DATASETS," 2024. [Online]. Available: https://doi.org/10.14569/IJACSA.2024.0150235

[23] W. L. Hamilton, R. Ying, and J. Leskovec, "INDUCTIVE REPRESENTATION LEARNING ON LARGE GRAPHS."

[24] Q. Zhu, N. Ponomareva, J. Han, and B. Perozzi, "SHIFT-ROBUST GNNS: OVERCOMING THE LIMITATIONS OF LOCALIZED GRAPH TRAINING DATA," 2021. [Online]. Available: https://github.com/GentleZhu/Shift-Robust-GNNs.

[25] H. T. Kose, J. Nunez-Yanez, R. Piechocki, and J. Pope, "A SURVEY OF COMPUTATIONALLY EFFICIENT GRAPH NEURAL NETWORKS FOR RECONFIGURABLE SYSTEMS," *Information (Switzerland)*, vol. 15, no. 7, Jul. 2024, doi: https://doi.org/10.3390/info15070377.

[26] M. S. M. Bhuiyan *et al.*, "DEEP LEARNING FOR ALGORITHMIC TRADING: A SYSTEMATIC REVIEW OF PREDICTIVE MODELS AND OPTIMIZATION STRATEGIES," Jul. 01, 2025, *Elsevier B.V.* doi: https://doi.org/10.1016/j.array.2025.100390.