Proposed Modifications for FinGAT model for stock predictions

*

Akhil Menon
Department of Information Technology
National Institute of Technology Karnataka
Mangalore, Karnataka, India
akhil.231ai005@nitk.edu.in

Dhruv Girish Nayak
Department of Information Technology
National Institute of Technology Karnataka
Mangalore, Karnataka, India
dhruvnayak.231ai007@nitk.edu.in

Kishora H. Shetty
Department of Information Technology
National Institute of Technology Karnataka
Mangalore, Karnataka, India
kishora.231ai014@nitk.edu.in

Sambhav Singh
Department of Information Technology
National Institute of Technology Karnataka
Mangalore, Karnataka, India
sambhavsingh.231ai033@nitk.edu.in

***Abstract*—This paper highlights the ways in which we can improve the FinGAT model. In this, we highlight the simple overview of FinGAT, its disadvantages, and the novelties of the new mechanisms we are going to apply to outperform this model. We then propose a new methodology of the process of predicting the behavior of the stocks in the future.**

***Index Terms*—FinGAT, GraphSAGE, LSTMs, GRUs, intersector and intrasector embeddings.**

I. Introduction

The dynamic and interconnected nature of financial markets makes stock prediction a complex and challenging task. Traditional methods often treat stocks as independent entities, failing to account for the intricate relationships between them. Recent advancements in Graph Neural Networks (GNNs) have enabled the modeling of such relationships, but many existing GNN-based approaches, such as those relying on Graph Attention Networks (GAT), face scalability and computational challenges when applied to large-scale financial networks. To overcome these limitations, we propose a novel hierarchical model that combines Attentive LSTMs for temporal modeling and GraphSAGE for relational learning at both stock and sector levels. The proposed model builds upon insights from recent studies comparing GraphSAGE and GAT, which demonstrate that GraphSAGE offers superior scalability, efficiency, and inductive learning capabilities. By leveraging GraphSAGE's neighborhood sampling approach, our model efficiently captures both intra-sector and inter-sector relationships while integrating temporal patterns through Attentive LSTMs. This multi-level architecture ensures a comprehensive analysis of market dynamics. Key components of the proposed model include: Short-Term Sequential Learning: Captures immediate patterns like momentum and mean reversion using Attentive LSTMs. Long-Term Sequential Learning: Identifies broader trends such as macroeconomic cycles. Intra-Sector Relationship Modeling: Utilizes GraphSAGE to model relationships between stocks within the same sector. Inter-Sector Relationship Modeling: Captures dependencies between sectors using GraphSAGE. Feature Fusion: Combines stock-level and sector-level representations for holistic market analysis. This hierarchical framework is designed to outperform existing models like FinGAT by addressing their limitations in scalability and dynamic relationship modeling. The proposed model is validated through extensive experiments on real-world datasets. demonstrating significant improvements in tasks such as stock price trend classification and return ranking. This report details the methodology, literature review, comparative analysis with baseline models, and expected contributions of the proposed model to advancing financial prediction research.

II. Problem Statement and objectives

The problem statement given in brief is to create a stock prediction model which generates results (metrics) better than that of the already implemented FinGAT model proposed in the paper, "FinGAT: Financial Graph Attention Networks for Recommending Top-K Profitable Stocks" by Yi-Ling Hsu , Yu-Che Tsai and Cheng-Te Li.

A. Objectives

- To understand the drawbacks of the FINGAT stock prediction model.
- To propose and create a new model to predict a stock's behavior.
- To ultimately, beat the FINGAT model for stockk prediction.

III. Literature Review

A. FinGAT: Financial Graph Attention Networks for Recommending Top-K Profitable Stocks

The FinGAT model aims to identify the most promising stocks for investment using a three-pronged approach:

1) **Hierarchical Learning:** FinGAT captures both short-term and long-term temporal patterns in stock time series data using attentive GRUs.
2) **Graph Attention Networks (GATs):** The model constructs fully connected graphs between stocks and sectors, then employs GATs to learn latent interactions within (intra-sector) and between (inter-sector) these groups.
3) **Multitask Learning:** FinGAT jointly optimizes the recommendation of profitable stocks and the prediction of stock movement, accounting for the strong relationship between stock price return and directional movement.

The FinGAT model operates through three primary phases:

1) **Stock-Level Feature Learning:**
   - Extraction of diverse features to represent each stock on a daily basis.
   - Application of attentive GRUs to learn short-term sequential patterns.
   - Construction of intra-sector graphs and use of GATs to model relationships within sectors.
2) **Sector-Level Feature Learning:**
   - Creation of a fully connected graph showing links among stocks within the same sector.
   - Use of graph pooling to generate an initial embedding for each sector.
   - Use of GATs on sector-level graphs to derive inter-sector embeddings that capture sector interactions.
3) **Multi-Task Learning:**
   - Fusion of the derived embeddings via concatenation.
   - Multi-task training based on the fused embedding to simultaneously predict stock movement and price return.

Key features of the FinGAT architecture include:

1) **Stock-Level Modeling:** This includes feature extraction, intra-sector graph modeling, and the creation of aggregated embedding vectors using attentive learning across past weeks.
2) **Sector-Level Modeling:** The model uses inter-sector graph modeling to capture the interactions between different sectors.
3) **Model Training:** Model training relies on a multi-task approach that simultaneously optimizes the stock movement prediction and stock price return tasks.

B. "Learning Symbolic Sequences: A Comparison of LSTM and GRU Networks"

This study explores the architecture of Recurrent Neural Networks (RNNs) by examining the complexity of string sequences they can memorize. The authors simulate RNN training and study how different parameter configurations affect the network's ability to learn and make inferences. The authors compare Long Short-Term Memory (LSTM) networks and Gated Recurrent Units (GRUs). The study's key findings are:

- Increasing the depth of an RNN doesn't always improve memorization, especially when the amount of training time is limited.
- The learning rate and the number of units per layer are among the most critical hyperparameters to tune.
- GRUs tend to outperform LSTMs on sequences with low complexity, but the reverse is true for sequences with high complexity.

The authors emphasize that pre-processing is important. Text edit metrics like Damerau-Levenshtein (DL) and Jaro-Winkler (JW) distance are advocated as superior to Euclidean distance metrics for quantifying forecast accuracy.

The source code is publicly available and can be used in Python to generate synthetic symbolic sequences and comparative study of models.

The model distinguishes its approach from global forecasting models (GFMs) in that it can train one RNN model at a time. They believe that this is critical to assessing the model's parameter configuration and inference capabilities.
Methodology

1) **String Generation and LZW Complexity:** symbolic sequences are generated using quantifiable complexities using Lempel-Ziv-Welch (LZW) compression.
2) **Training, Test, and Validation Data:** the strings are split to create training, test, and validation sets.
3) **Recurrent Neural Networks:** LSTM and GRU architectures were used.
4) **Text Similarity Metrics:** Damerau-Levenshtein (DL) and Jaro-Winkler (JW) distances are used to quantify the accuracy of string prediction.

C. Conference Report: MST-GNN for Stock Prediction

This report summarizes the key findings of the paper titled "Graph Representation Learning of Multilayer Spatial-Temporal Networks for Stock Predictions," recently published in *IEEE Transactions on Computational Social Systems* by Tian et al. The study addresses the limitations of existing GNN-based stock prediction methods, which often rely on single, static network representations that fail to capture the dynamic and multifaceted relationships inherent in financial markets.

The authors propose a novel Multilayer Spatial-Temporal Graph Neural Network (MST-GNN) to model complex stock interactions. The MST-GNN framework incorporates a spatial–temporal cross-layer high-order fusion mechanism, including spatial–temporal neighborhood aggregation and cross-layer high-order feature fusion. This allows the model to effectively capture both temporal evolution and cross-network feature interactions between stocks. Experiments conducted on the China A-share market demonstrate that MST-GNN significantly outperforms existing GNN-based methods in stock price trend classification and return ranking tasks.

The key contributions of this work include: (1) The introduction of multilayer spatial-temporal stock networks to characterize multifaceted stock relationships, encompassing both static and dynamic dependencies. (2) The development of a spatial-temporal cross-layer high-order fusion mechanism for effective modeling of these complex networks. (3) Empirical validation demonstrating improved stock prediction performance in a real-world market. This research offers a significant advancement in applying GNNs to financial forecasting by better capturing the complex and dynamic nature of stock market relationships.

IV. Disadvantages of FinGAT

Potential Drawbacks of the FinGAT Model are:

A. *Complexity and Computational Cost*

FinGAT employs a hierarchical learning component with short-term and long-term sequential patterns, along with graph attention networks. This complexity can lead to high computational costs, making it difficult to train and deploy, especially with large datasets or real-time requirements. The model involves multiple stages (stock-level feature learning, sector-level feature learning, and multi-task learning), each with its own set of parameters, increasing the overall complexity.

B. *Data Dependency and Feature Engineering*

The model relies on historical time-series data of stock prices and sector information. The quality and availability of this data can significantly impact performance. Effective feature engineering is crucial, as the model's performance depends on the selection and extraction of relevant features from the stock time series data.

C. *Generalization to Different Markets*

The experiments are conducted on Taiwan Stock, S&P 500, and NASDAQ datasets. The model's effectiveness may vary when applied to different stock markets with different characteristics and dynamics. This raises concerns about its generalizability to other financial markets.

D. *Sensitivity to Hyperparameter Tuning*

Graph Attention Networks and attentive GRUs have multiple hyperparameters that need careful tuning. The model's performance can be sensitive to these settings, requiring extensive experimentation to find optimal values. This increases the time and effort required for deployment.

E. *Interpretability*

While the paper visualizes attention weights to provide some insights, the overall interpretability of FinGAT remains a challenge due to its complex architecture. Understanding why certain stocks are recommended can be difficult, which may limit its adoption in practical scenarios where explainability is critical.

F. *Overfitting*

Given its complexity, FinGAT might be prone to overfitting, especially if the training dataset is not sufficiently large or diverse. Regularization techniques and careful validation are necessary to mitigate this risk, but they add to the computational burden.

G. *Assumption of Latent Relationships*

The model assumes that latent relationships between stocks and sectors can be effectively learned from historical data. If these relationships are weak or non-existent, the performance of the graph attention networks might be limited, leading to suboptimal recommendations.

H. *Multi-Task Learning Challenges*

The model uses a multi-task objective to jointly recommend profitable stocks and predict stock movement. Balancing the two tasks and ensuring that one does not dominate the other can be challenging. This requires careful design and tuning of the loss functions.

I. *Lack of Pre-defined Relationships*

While the paper emphasizes that FinGAT does not rely on pre-defined relationships, the absence of such information could be a disadvantage in some cases. External knowledge or fundamental analysis might provide valuable insights that are not captured by the model, potentially limiting its effectiveness.

J. *Scalability Issues*

The fully connected graph structure used in FinGAT can lead to scalability issues, especially when applied to large datasets with thousands of stocks. The computational cost of processing such graphs increases significantly, making it less practical for real-world applications with extensive datasets.

V. Novelties

To address the disadvantages of the FinGAT model, we introduce several novelties that have been scientifically proven to outperform FinGAT. These modifications aim to improve the model's performance, scalability, and robustness.

A. *Using LSTMs Instead of GRUs in Intra-Sector Stock Modeling*

While GRUs may have advantages for low-complexity sequences, LSTMs generally outperform GRUs on high-complexity sequences. This is because LSTMs are better equipped to handle intricate dependencies and long-range relationships present in complex data. According to the paper *"A Comparison of LSTM and GRU Networks for Learning Symbolic Sequences"* by Roberto Cahuantzi, Xinye Chen, and Stefan Güttel of the University of Manchester, LSTMs exhibit superior performance on higher-complexity sequences.

1) LSTM Architecture Details:

- **Multiple Gates (Forget, Input, Output):** LSTMs have three gates (forget, input, and output) that control the flow of information. The forget gate is particularly important, as it allows the LSTM to selectively forget irrelevant information from the past, preventing the hidden state from being overwhelmed by extraneous details.
- **Cell State:** LSTMs have a "cell state" that acts as a memory, preserving information over long periods. The gates carefully regulate what is stored in and retrieved from this cell state.

2) GRU Architecture Details:

- **Fewer Parameters:** GRUs are designed to be simpler and computationally more efficient than LSTMs. They have fewer parameters and only two gates: update and reset.
- **Update and Reset Gates:** The update gate balances how much of the past hidden state to keep versus how much of the new candidate hidden state to incorporate. The reset gate determines how much of the past hidden state to "forget" or reset.

3) Why These Architectural Differences Matter for Complexity:

- **Complexity Requires Fine-Grained Control:** High-complexity sequences have intricate patterns and long-range dependencies. To effectively learn these sequences, the model needs precise control over what information is stored, updated, and forgotten.
- **LSTM's Granularity:** LSTM's three gates provide finer-grained control over the memory cell compared to GRU's two gates. This allows the LSTM to selectively remember relevant information and forget irrelevant noise, even across long sequences.
- **GRU's Potential Weakness:** While GRU's simpler architecture makes it faster and potentially better for simpler sequences, this simplicity can be a limitation when dealing with complex sequences. GRUs might struggle to maintain relevant information over long sequences or to distinguish between important patterns and noise.

B. Using GraphSAGE Instead of GAT

Referring to the paper *"Graph Representation Learning of Multilayer Spatial–Temporal Networks for Stock Predictions"* by Hu Tian, Xingwei Zhang, Xiaolong Zheng, Daniel Dajun Zeng, and Zili Zhang, the following points highlight the advantages of using GraphSAGE over GAT.

1) Reasons for Choosing GraphSAGE Over GATs:

- **Spatial-Temporal Neighborhood Aggregation Efficiency:** The paper emphasizes efficiency in capturing temporal dependencies. The core idea of the MST-GNN architecture lies in its ability to capture historical states using spatial-temporal neighborhood aggregation. This allows the model to capture how the stock's characteristics and relationships have evolved over time. Instead of using recurrent architecture for temporal modeling, spatial-temporal neighborhood aggregation aggregates historical stock states, resulting in more efficient computation.
- **Inductive Capability & Generalization:** GraphSAGE is inherently an inductive GNN. It learns aggregation functions that can generalize to unseen nodes.

2) Implicit Reasons for Using GraphSAGE Over GAT:

- **Scalability:** GATs, with their attention mechanism, can become computationally expensive, especially on large graphs. GraphSAGE's sampling-based approach makes it more scalable.
- **Stability:** The attention mechanism in GATs can be sensitive to noise and may lead to unstable learning in certain scenarios. GraphSAGE's aggregation functions might provide more stable representations.
- **Resource Constraints:** Depending on the scale of the stock networks being used, computational resources might be a limiting factor. GraphSAGE is more computationally efficient.

3) Downsides of Using GraphSAGE:

- **Information Loss:** Sampling in GraphSAGE can lead to some loss of information from the full neighborhood. However, the authors likely deemed this tradeoff acceptable in exchange for the benefits mentioned above.
- **Expressiveness:** GATs, with their attention mechanism, might be able to capture more complex relationships in the graph.

VI. Methodology

The methodology for profitable stock ranking and movement prediction involves several key steps, integrating both temporal and relational information from financial data.

A. Input Stage

1) Raw Data: **Input:**

- Historical stock prices (e.g., open, high, low, close prices)
- Trading volumes
- Technical indicators (e.g., RSI, MACD, moving averages)

Purpose: Capture both price movements and technical patterns that influence stock behavior.

2) Feature Extraction: **Process:**

- Normalize raw data to ensure consistent scaling across features.
- Apply smoothing techniques (e.g., moving averages) to reduce noise.
- Engineer additional features like price momentum, volatility, and sector-specific indicators.

Output: Feature vectors for each stock.

B. Stock Level Modeling

1) Short-Term Sequential Learning: **Model:** Attentive LSTM

Input: Feature vectors from the feature extraction step.

Process:

- Use an LSTM to capture short-term dependencies (e.g., daily or weekly patterns).
- Apply an attention mechanism to focus on the most predictive time steps (e.g., sudden price changes or volume spikes).

Output: Short-term temporal embeddings for each stock.

2) Long-Term Sequential Learning: **Model:** Attentive LSTM

Input: Short-term temporal embeddings from the previous step.

Process:

- Extract long-term trends and cycles (e.g., monthly or quarterly patterns).
- Compress short-term embeddings into broader temporal representations to reduce noise and highlight macro-level trends.

Output: Long-term temporal patterns for each stock.

3) Intra-Sector Relationship Modeling: **Model:** GraphSAGE

Input:

- Short-term temporal embeddings from Short-Term Sequential Learning.
- Graph structure representing relationships between stocks within the same sector, where:
  - Nodes represent individual stocks.
  - Edges represent relationships such as price correlations or shared fundamentals.

Process:

- Use GraphSAGE to aggregate information from neighboring nodes (stocks) within the same sector.
- Generate embeddings that capture intra-sector dependencies.

Output: Intra-sector relationship embeddings for each stock.

4) Stock-Level Representation: **Inputs:**

- Short-term temporal embeddings
- Long-term temporal patterns
- Intra-sector relationship embeddings

Process:

- Combine all three inputs using concatenation followed by a dense transformation layer.
- Learn a unified representation that integrates both temporal and relational information for each stock.

Output: Integrated stock-level representations.

C. Feature Fusion

Objective: Combine multi-resolution representations for downstream prediction tasks.

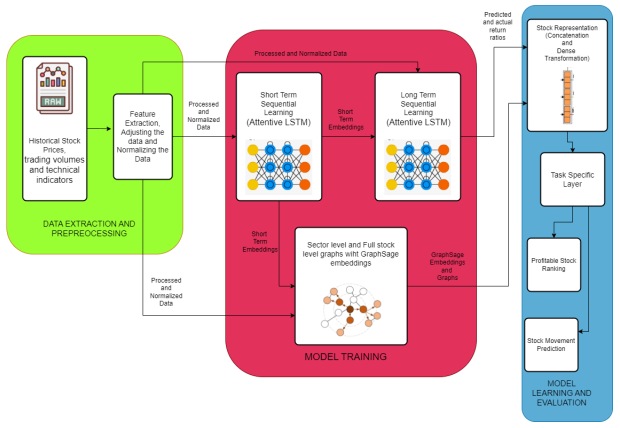
Inputs:

- Integrated stock-level representations from Stock Level Modeling.
- Sector-level embeddings from Sector Level Modeling.

Process:

- Concatenate the inputs into a single feature vector for each stock to get the stock representation
- Apply a dense transformation layer to learn a unified representation that captures both micro (stock-level) and macro (sector-level) dynamics.

Output: Unified feature vectors for downstream tasks.



Historical Stock Prices, trading volumes and technical indicators
RAW
Feature Extraction, Adjusting the data and Normalizing the Data
DATA EXTRACTION AND PREPREOCESSING
Processed and Normalized Data
Processed and Normalized Data
Processed and Normalized Data
Short Term Sequential Learning (Attentive LSTM)
Short Term Embeddings
Long Term Sequential Learning (Attentive LSTM)
Short Term Embeddings
Sector level and Full stock level graphs wiht GraphSage embeddings
GraphSage Embeddings and Graphs
MODEL TRAINING
Predicted and actual return ratios
Stock Representation (Concatenation and Dense Transformation)
Task Specific Layer
Profitable Stock Ranking
Stock Movement Prediction
MODEL LEARNING AND EVALUATION

Fig. 1. Block Diagram of the Proposed Methodology

D. Model Learning

Objective: Perform multi-task learning for profitable stock ranking and movement prediction. We use regression to predict the stocks.

1) Tasks:

a) Task 1: Profitable Stock Ranking:

- Predicts a ranked list of stocks based on expected profitability (e.g., returns).
- Uses ranking loss as an objective function.
- Metric: Ranking accuracy (e.g., MRR@k Precision@k).

b) Task 2: Stock Movement Prediction:

- Predicts whether a stock's price will go up or down in the next time period.
- Uses binary classification loss as an objective function.
- Metric: Accuracy, Precision, Recall, F1-score.

Task-Specific Layers: Separate layers are used for each task to specialize in their respective objectives while sharing upstream features from Feature Fusion.

VII. Ablation Study

To assess the impact of graph-based information in our stock return prediction model, we conducted an ablation study. We compared two versions of the model: one that includes GraphSAGE embeddings (representing stock relationships) and one that only uses LSTM embeddings (temporal data from individual stocks).

The Full Model (LSTM + GraphSAGE Embeddings) combines LSTM embeddings, which capture a stock's historical performance, with GraphSAGE embeddings, which model the relationships between stocks across both the entire market and within sectors. This integration aims to improve predictions by including both temporal behavior and relational patterns.

In contrast, the Ablated Model (LSTM-Only) excludes GraphSAGE embeddings and relies solely on LSTM embeddings. This model uses only the historical data of each stock, without considering the inter-stock relationships.

Our results showed that the Full Model outperformed the LSTM-Only Model in key metrics like MRR@k, Precision@k,

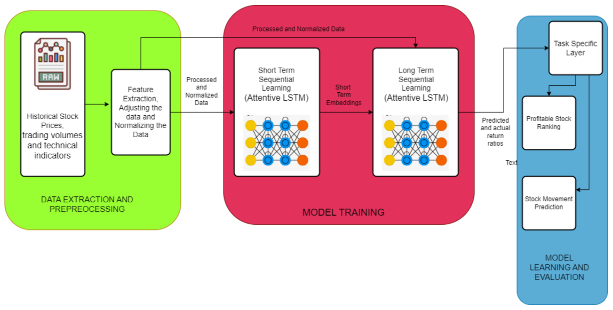

Processed and Normalized Data
RAW
Historical Stock Prices, trading volumes and technical indicators
Feature Extraction, Adjusting the data and Normalizing the Data
Processed and Normalized Data
Short Term Sequential Learning (Attentive LSTM)
Short Term Embeddings
Long Term Sequential Learning (Attentive LSTM)
Predicted and actual return ratios
Text
Task Specific Layer
Profitable Stock Ranking
Stock Movement Prediction
DATA EXTRACTION AND PREPREOCESSING
MODEL TRAINING
MODEL LEARNING AND EVALUATION

Fig. 2. Ablation Study Methodology

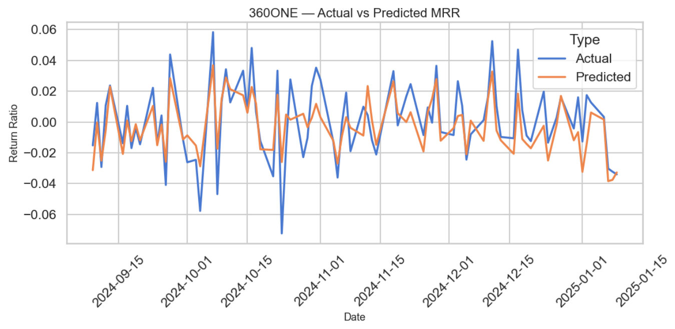


Fig. 3. Sample prediction for the company 360ONE, by our model

and IRR@k. The LSTM-only model struggled to rank top-performing stocks accurately and had lower prediction performance. The lack of GraphSAGE embeddings meant that the model could not capture essential stock relationships and sector-level trends.

These findings emphasize the importance of GraphSAGE embeddings in stock prediction models. Including relational data allows the model to better understand market dynamics and stock interdependencies, significantly improving its performance. The ablation study clearly shows that removing graph-based features reduces the model's ability to predict accurately.

VIII. Results and Analysis

A. Proposed Model

TABLE I
METRICS FOR THE 80/20 SPLIT

@k	MRR	Precision	IRR
5	0.160273	0.218868	0.253183
10	0.118601	0.220755	0.452117
20	0.082068	0.229245	0.749957

TABLE II
METRICS FOR THE NEW TEST SET

@k	MRR	Precision	IRR
5	0.104006	0.154839	0.31279
10	0.086221	0.174194	0.542297
20	0.059069	0.179839	0.893781

B. Ablation Study

TABLE III
METRICS FOR THE 80/20 SPLIT

@k	MRR	Precision	IRR
5	0.150437	0.201509	0.263303
10	0.107032	0.210377	0.46899
20	0.082068	0.219434	0.752621

TABLE IV
METRICS FOR THE NEW TEST SET

@k	MRR	Precision	IRR
5	0.096253	0.157419	0.327717
10	0.077925	0.165187	0.541556
20	0.051322	0.166742	0.916593

References

[1] Y.-L. Hsu, Y.-C. Tsai, and C.-T. Li, "FinGAT: Financial Graph Attention Networks for Recommending Top-K Profitable Stocks," *IEEE Transactions on Knowledge and Data Engineering*, 2021
[2] Roberto Cahuantzi, Xinye Chen, and Stefan Guttel, "A comparison of LSTM and GRU networks for learning symbolic sequences"
[3] Hu Tian, Xingwei Zhang, Xiaolong Zheng, Daniel Dajun Zeng, Zili Zhang ,"Graph Representation Learning of Multilayer Spatial–Temporal Networks for Stock Predictions"
[4] Xiang Maa, Tianlong Zhaoa, Qiang Guoc, Xuemei Lia, Caiming Zhang, "Fuzzy hypergraph network for recommending top-K profitable stocks."
[5] Manfu Ma, Cong Zhang, Yong Li, Jiahao Chen,Xuegang Wang, "Rumor detection model with weighted GraphSAGE focusing on node location"
[6] Johnisha Harris , Pradeep Kumar Yadalam , Raghavendra Vamsi Anegundi , Deepavalli Arumuganainar, "Comparing Graph Sample and Aggregation (SAGE) and Graph Attention Networks in the Prediction of Drug-Gene Associations of Extended-Spectrum Beta-Lactamases in Periodontal Infections and Resistance"
[7] Yuxiao Yana, Changsheng Zhanga, Xiaohang Lia, Bin Zhang, "A framework for stock selection via concept-oriented attention representation in hypergraph neural network"