December 11, 2025

# Visual Graph Learning with GCN for Financial Forecasting: CNN-Based OHLC Image Embeddings and Market Topology

Chun Hung Tsang[*], Kuan-Lin Lai[†]

## Abstract

This study examines whether combining image-based deep learning with financial network structure can improve stock prediction and portfolio performance in the Hong Kong equity market. We approach this problem by integrating price-chart representations with network-based relational learning to capture both individual visual patterns and inter-stock dependencies. Our model first transforms stock price charts into visual features using a Convolutional Neural Network (CNN), then constructs a market-topology representation from pairwise similarities among the visual features with Minimum Spanning Tree (MST) filtering, and finally applies a Graph Convolutional Network (GCN) over a filtered network to learn cross-stock dependencies. Empirical evaluation on long–short portfolios shows that image-driven models alone already outperform the market benchmark, suggesting that technical price structure contains exploitable information even in a stagnant market. The most striking finding is that graph network-based learning enhances performance only when the predictive task is framed as a cross-sectional ranking problem. When absolute return thresholds are used instead, the relational component provides little benefit. These results highlight the importance of aligning supervisory signals with market structure and demonstrate the potential of visual–topological modeling for equity forecasting.

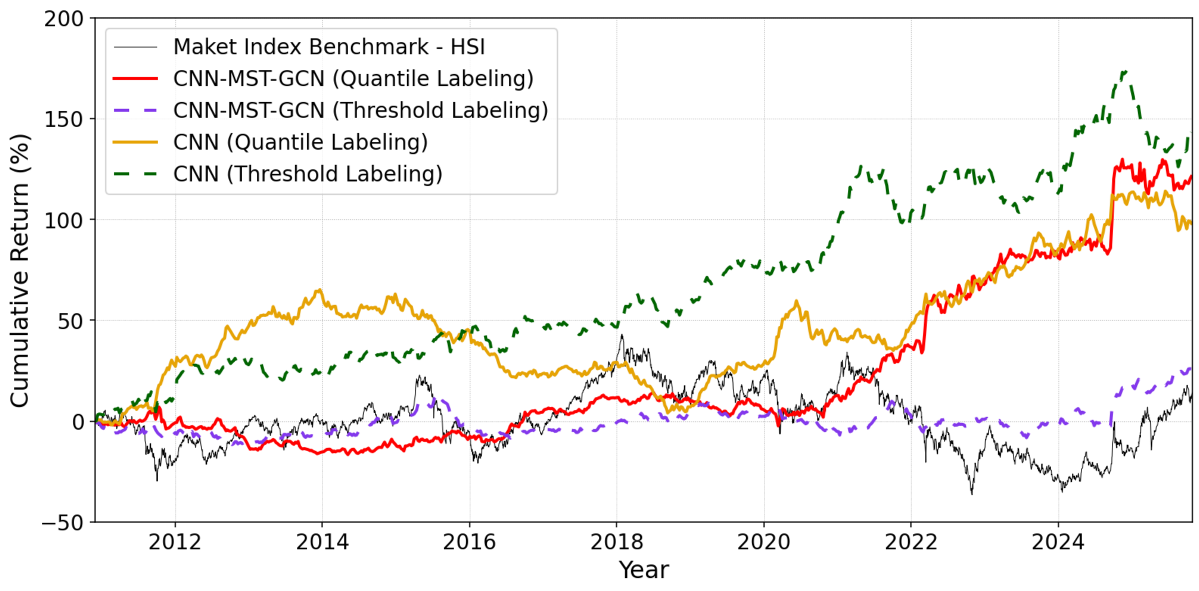

Figure 1. **Out-of-sample cumulative return of CNN–MST–GCN and mere-CNN models under different labeling schemes**. The HSI market benchmark represents the performance of the stock market examined, and the four models only select and invest in the constituent stocks of HSI.

[*] 曾俊雄. Department of Applied Mathematics, National Sun Yat-sen University (genniswork@gmail.com)
[†] 賴冠霖. Department of Applied Mathematics, National Sun Yat-sen University (M132040012@student.nsysu.edu.tw)

## 1 Introduction

Financial forecasting has long relied on two distinct methodological traditions: technical analysis, which extracts predictive patterns from price charts, and portfolio theory, which emphasizes interdependence among assets. While recent advances in deep learning—particularly Convolutional Neural Networks (CNNs)—have significantly improved the extraction of price–volume patterns from candlestick images to extend the spirit of technical analysis (Jiang et al., 2023; Zhu and Zhu, 2024), these image-based methods still evaluate each stock in isolation. This omits the fundamental rules of portfolio theory, which emphasizes risk diversification through the correlation between price movements in the stock market such that risk-adjusted return is maximized (Markowitz, 1952). As a result, traditional CNN-focused trend forecasting models leverage visual signals but overlook the systemic relationships that underlie risk propagation, sectoral clustering, and cross-stock dependencies (Chen et al., 2021).

On the other hand, financial network analysis views markets as graphs, where securities are nodes and correlations represent edges. Empirical studies show that the topology of such networks, especially node centrality, influences optimal portfolio weights (Pozzi et al., 2013; Peralta and Zareei, 2016; Millington and Niranjan, 2021; Olmo, 2021). Yet, these network-based methods rely purely on return correlations and neglect the visual and temporal dynamics embedded in market prices. In addition to building network based on return correlations, various types of financial data are used in building and examining the interdependence among stocks, including exploitation of the similarity among price charts of stocks (Li et al., 2022; Ma and Yuan, 2024; Ma et al., 2024). To further extract the market interdependence other than the topological properties of financial networks, these studies introduce Graph Convolutional Networks (GCNs) to study the implicit non-Euclidean market structures embedded with the graphs built by financial networks. Prior research demonstrates that graph representations derived from correlations, distances, or economic linkages can capture market topology and enhance predictive modeling. However, existing GCN-based studies typically construct graphs from time-series data, correlations, or textual events. Far less attention has been given to capturing visual relationships among price charts, despite compelling evidence that image embeddings encode nonlinear trend information often invisible to numerical predictors.

This research seeks to bridge these two lines of work by developing a visual graph learning framework that integrates CNN-based image representation with graph-based modeling using Minimum Spanning Tree (MST). The objective is to jointly capture individual visual pattern representations within each asset and collective visual co-movements across the market to enhance financial forecasting and portfolio construction. Our CNN-MST-GCN approach is summarized as follows. First, stock price data are converted into candlestick chart images. Second, visual features of stocks are extracted by CNN and then their associations are captured by financial network. We then employ GCN to learn this financial network to predict returns and construct portfolios. Finally, we examine the out-of-sample investment performance of the portfolio.

The novelty of this research lies in:

1. The key paper we follow and modify is Jiang et al. (2023), which build investment portfolios by capturing price–volume patterns from candlestick images using CNN. We modify their CNN architecture by (i) the ResNet architecture on similar task advocated by Zhu and Zhu (2024), (ii) different pooling layer design, and (iii) cutting-edge functions, including AdamW optimizer and GELU activation function.
2. We modify the ResNet-18 architecture by 1×1 bottleneck design.
3. Apart from using return-based threshold for training target adopted in Jiang et al. (2023) with modification to set a new minimum return threshold, we also introduce a quantile-based target to encourage the model to identify stocks that outperform peers within the same period, aligning more directly with the objective of rank-based portfolio selection.
4. Pozzi et al. (2013), Peralta and Zareei (2016), and Olmo (2021) investigate interrelationships among stocks and construct portfolios by building financial networks. Li et al. (2022), Ma and Yuan (2024), and Ma et al. (2024) build financial networks by similarity among price charts of stocks. We modify the element of building financial networks in using image representations extracted from CNN model trained.
5. Li et al. (2022), Ma and Yuan (2024), and Ma et al. (2024) apply dynamic time warping (DTW) and Pearson correlation coefficient to compute the similarity between the raw price chart images. Narayanaswamy and Gowda (2024) demonstrate that cosine similarity effectively captures image information when applied within the VGG19 and Gated Recurrent Unit (GRU) frameworks. We here adopt Frobenius cosine similarity to compute the similarity between the image representations.
6. Millington and Niranjan (2021) demonstrate the advantage of implementation of topological filtration to a financial network. Hence, we apply Minimum Spanning Tree (MST) to financial networks built by image representations for further mitigation of noise and preservation of most meaningful connections.
7. Chen et al. (2021) propose a graph convolutional feature based convolutional neural network (GC–CNN) model to predict future trends of target stock. Li et al. (2022), Ma and Yuan (2024), and Ma et al. (2024) utilize GCNs in extraction of information from financial networks for stock movement prediction. In this research, we extend the CNN model advocated by Jiang et al. (2023) to a CNN-MST-GCN framework to capture price–volume patterns from candlestick images such that the MST financial networks built by image representations from CNN model are trained by GCN model proposed.
8. Apart from learning the interrelationships among stock through financial networks in GCN learning, we also set image representations as the stock-specific features in GCN learning. However, the dimension of stacking image representations is too high when acting as a stock-specific feature. We employ depthwise separable convolution to reduce the dimension to improve the efficiency of GCN learning.
9. We customize the GCN architecture to be more fitting with our OHLC-specific task and network topology by incorporating pre-activation, the graph normalization proposed by Cai et al (2021), and

the residual connections advocated by Zhang et al. (2023).

10. We generalize Markowitz (1952) mean–variance intuition into a deep learning framework by jointly modeling expected returns via CNN embeddings and cross-asset dependencies via GCN propagation.

## 2 Convolutional Neural Network

In this section, we provide detailed explanation of the design of the CNN architecture employed in our research, which serves as the visual feature extractor in our CNN–MST–GCN framework and also functions as a standalone return classifier for performance benchmarking. and outline the training methodology.

### 2.1 Generation of CNN Inputs: OHLC Charts

In this subsection, we explain how stock market data are transformed into image-based inputs for the CNN forecasting model. Yahoo! Finance is popular in visualizing historical price charts for stocks. We use 1-year stock information of HSBC Holdings plc up to October 31, 2025 as an example of the chart as shown in Figure 2. It includes daily open, high, low, and close prices (so-call "OHLC"). It also plots daily trading volume at the bottom and overlays a 20-day moving average line.

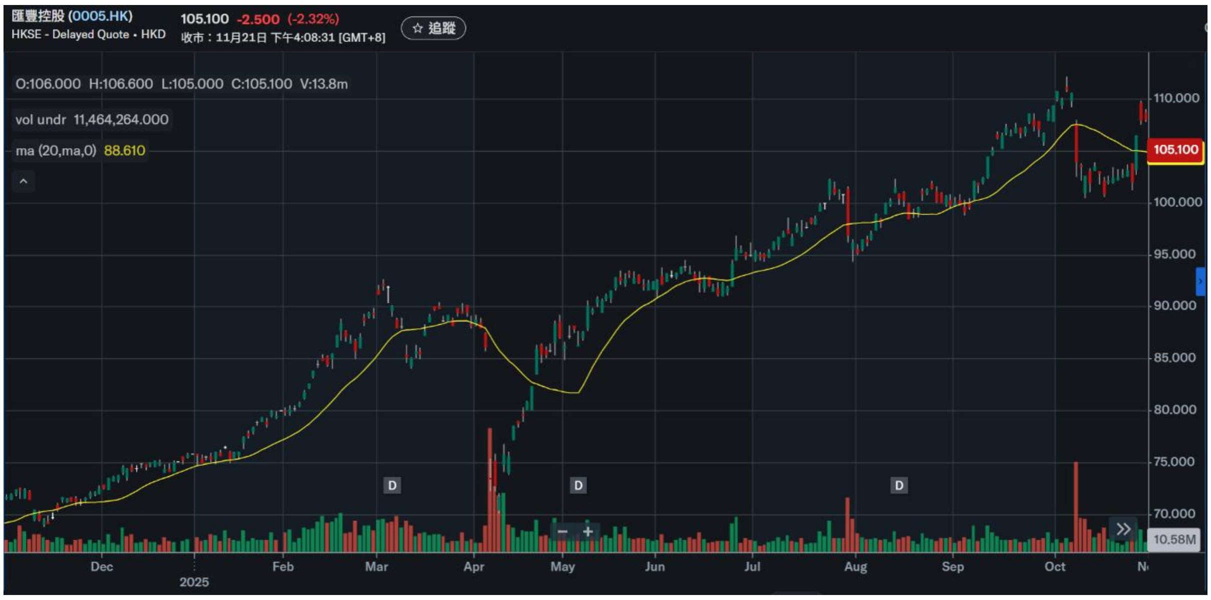

Figure 2. **HSBC OHLC chart from Yahoo! Finance**. This figure displays an OHLC chart for stock of HSBC Holdings plc with 20-day moving average price line and daily volume bars. Data are daily from November 1, 2024, to October 31, 2025. The moving average line is colored yellow. Green elements indicate upward movement—green candlesticks show days when the closing price is higher than the opening price, and green volume bars indicate trading volume on such up days. Red elements indicate downward movement—red candlesticks show days when the closing price is lower than the opening price, and red volume bars indicate trading volume on down days.

Following Jiang et al. (2023), we generate OHLC bar images (colored in white with black background) as our CNN model inputs. First, each price chart visualizes OHLC data using white bars with black background, where the high and low prices correspond to the upper and lower ends of the central vertical bar, while the open and close prices are indicated by short horizontal ticks on the left and right sides, respectively. In this design, a single trading day spans three pixels in width—one for the central bar and one each for the open and close markers. Second, 20-day moving average line is drawn by connecting one-pixel markers for each day. Third, daily trading volume bars position in the lower one-fifth of the image, while the upper four-fifths display the price chart. To ensure comparability, trading volumes are scaled so that the highest volume in each image reaches the upper boundary of the volume section, with all other values adjusted proportionally. Similarly, maximum-minimum scaling is also applied to the upper four-fifths of the image.

Mathematically, each input image is made up by a binary 2-D tensor with dimensions of $(H \times 3\rho)$, where $H$ represents the vertical pixel height and $\rho$ corresponds to the number of trading days in the observation window. The width of $3\rho$ results from assigning three-pixel columns per trading day. Each pixel takes a value of 1 to indicate the presence of chart elements and 0 to denote the background, effectively capturing the structure of the price pattern in a simplified black-and-white format. We illustrate an example of OHLC image constructed as Figure 3.

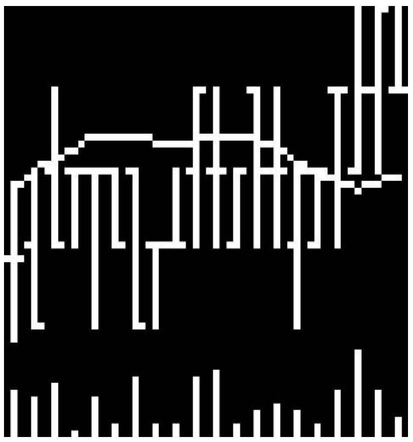

Figure 3. **Generated OHLC image of HSBC from May 16, 2005 to June 12, 2005**. This figure shows an example input image with 20 trading days.

## 2.2 Feature and Target of CNN

Consistent with the configuration in Jiang et al. (2023), we utilize a 20-day OHLC image, corresponding roughly to a one-month trading window. Each image is rendered with a height of 64 pixels and a width of 60 pixels, providing sufficient resolution to capture short-term price dynamics while maintaining computational efficiency; i.e., $H = 64$ and $\rho = 20$. The investment strategy is based on the information extracted from the OHLC image of past 20-day and make an investment with 5-day holding period. In this research, we call the period used to form features as *observation period* and that to form target as *investment period*.

In our empirical analysis, we investigate two different definitions of the training target $y$ to understand how distinct supervisory structures influence the learning performance of the CNN–MST–GCN framework. For the first target, we follow the directional return labeling approach of Jiang et al. (2023), but with a modified classification threshold. Instead of setting positive class when the subsequent 5-day return is strictly positive, we assign positive class if the 5-day return is greater than or equal to 0.3%, and negative class otherwise. This adjustment introduces a minimum return threshold that aligns the labeling rule with our investment objective. Since the goal is to outperform the market, we prefer not to classify a stock as attractive merely because it generates a small positive return, even if the model predicts that outcome with high probability. A stock that increases only marginally is unlikely to beat the market benchmark, and treating such cases as positive labels could distort the learning objective. Therefore, we set the cutoff at 0.3%, which corresponds to the weekly benchmark return of 0.3092%. We will introduce the benchmark adopted in the section of Data Description subsequently. This ensures that a stock is labeled as positive class only when its expected performance meaningfully exceeds the market baseline, making the target more economically relevant and better aligned with the portfolio selection task. Let $r_{i,t}$ be 5-day holding period return for stock $i$ at time $t$. The threshold-based label $y_{i,t}^{(thr)}$ is defined as:

$$y_{i,t}^{(thr)} = \begin{cases} 1, & \text{if } r_{i,t} \geq 0.3\%; \\ 0, & \text{otherwise.} \end{cases} \tag{1}$$

To complement this absolute-return criterion, we also employ a quantile-based target, where positive class if a stock's 5-day return falls within the top 30% cross-sectionally on that investment date, and negative class otherwise. This relative performance labeling encourages the model to identify stocks that outperform peers within the same period, aligning more directly with the objective of rank-based portfolio selection. In particular, such modification aims to find out the most profitable stocks at any moment. Since financial markets would continuously encounter bull and bear scenarios, this target definition avoid frequent assignment of positive class in the bull markets and negative class in the bear markets, so that the CNN model proposed can always identify the most profitable stocks used in stock selection of portfolio construction, instead of learning the same target given different features embedded in the OHLC images of stocks. Together, these two target designs allow us to assess the model's sensitivity to both absolute and relative measures of short-term return predictability. The quantile-based label $y_{i,t}^{(q)}$ is:

$$y_{i,t}^{(q)} = \begin{cases} 1, & \text{if } rank(r_{i,t}) \leq 0.3S; \\ 0, & \text{otherwise,} \end{cases} \tag{2}$$

where $rank(\cdot)$ denotes the rank of stock $i$ in descending order that assigns 1 to the largest 5-day return and $S$ to the smallest return, and $S$ is the number of stocks.

## 2.3 CNN Architecture

In this section, we provide a concise overview of our CNN's architectural design and the procedures used to train it. In Figure 4, the left diagram displays the CNN's architectural design used in Jiang et al.

(2023), and the right diagram presents our setting. As shown in Figure 4, our neural network follows a ResNet-style design proposed in Zhu and Zhu (2024) in OHLC image task and modify the design with combining shallow BasicBlocks with deeper bottleneck residual blocks to balance expressive power and parameter efficiency on OHLC images.

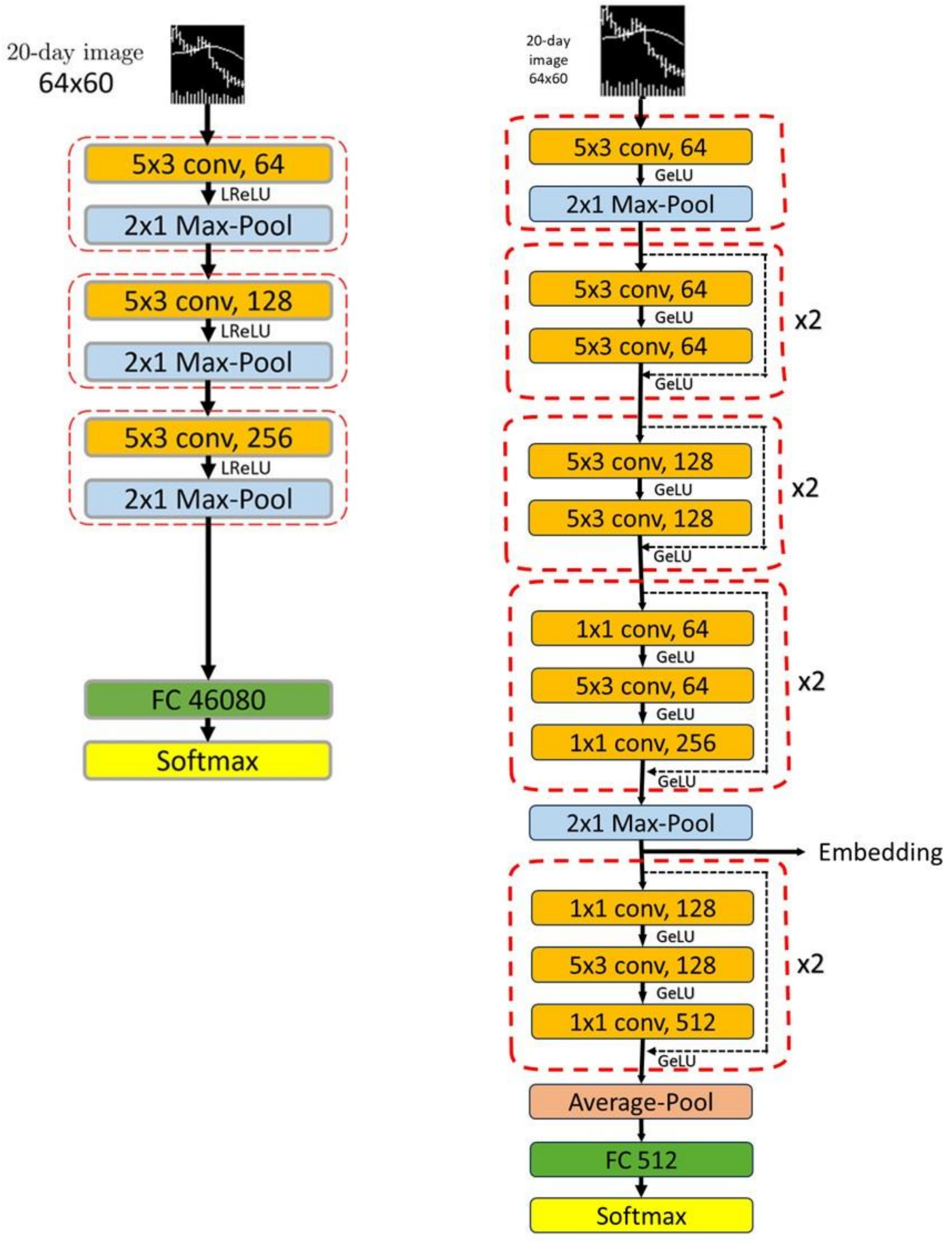

Figure 4. **CNN model architecture**.

As described in Section 2.1, the geometric structure of OHLC images is highly anisotropic: the horizontal axis corresponds to discrete time steps, while the vertical dimension reflects price ranges and volume zones. The CNN is therefore designed so that its convolutional kernels, pooling operators, and residual stages respect this inherently anisotropic spatial structure. The neural network begins with a stem comprising batch normalization and GELU activation, followed by a $5 \times 3$ convolution with 64 channels and padding $(2,1)$. This asymmetric kernel aligns exactly with the daily 3-pixel encoding and thus captures the smallest valid candlestick unit, while its 5-pixel vertical span reflects meaningful variations in price ranges and wick structures that occur across multiple days. A $2 \times 1$ max-pooling layer follows the stem, reducing only the vertical resolution so as not to disturb the strict left-to-right temporal ordering of the 20-day image.

The core modification of our model consists of four residual stages. The first two stages adopt the pre-activation BasicBlock design of ResNet-18 (He et al., 2016b). In these blocks, batch normalization and GELU activation are applied before each $5 \times 3$ convolution, followed by identity shortcuts. Stage 1 maintains the representation width at 64 channels, while Stage 2 expands the width to 128 channels through a projection shortcut in the first block. These stages extract local and intermediate-scale features such as candle-shape geometry, body–wick asymmetries, short-term reversals, and 2–4 day oscillations. Because each block uses pre-activation ordering rather than post-activation, the identity path remains clean, improving optimization stability and preserving gradient flow. Stages 3 and 4 replace BasicBlocks with the bottleneck residual structure originally introduced by He et al. (2016a). Each bottleneck block consists of a $1 \times 1$ convolution for channel compression, a $5 \times 3$ convolution for spatial feature extraction, and a $1 \times 1$ convolution for channel expansion, with batch normalization and GELU activation preceding every convolution. This squeeze–expand pattern significantly reduces the number of parameters and floating-point operations, allowing depth expansion and wide channel growth without compromising computational efficiency. Such efficiency is especially important for financial images, which are smaller, sparser, and noisier than natural images. Stage 3 increases the channel dimension from 128 to 256 and is followed by a second $2 \times 1$ max-pooling layer to reduce vertical resolution to 16 pixels while preserving the full 60-pixel horizontal span. The resulting output tensor has dimension $256 \times 16 \times 60$. Stage 4 further expands the representation to 512 channels and extracts higher-level abstractions that summarize extended multi-day patterns relevant to classification objectives. Table 1 summarizes the key differences between the original model adopted by Jiang et al. (2023) and our model.

A crucial aspect of the architectural design is the production of two distinct outputs serving two distinct roles in the overall framework. The output of Stage 3, prior to any global spatial aggregation, is extracted as the visual embedding for constructing the financial network. This representation preserves its full spatial resolution and contains mid-level abstractions that remain closely tied to the geometric structure of the OHLC images. Extracting the embedding at this depth is supported by insights from computer vision, where intermediate feature maps in residual networks are known to retain structural and textural

| | **Jiang et al. (2023)** | **Our CNN Model** |
|---|---|---|
| **Core Architecture** | Plain CNN | ResNet-like |
| **Module Design** | Single convolution stack | Basic Block + Bottleneck |
| **Batch Normalization** | Post-Activation | Pre-activation |
| **Activation Function** | Leaky ReLU | GELU |
| **Feature Aggregation** | Flatten | Global Average Pooling |
| **Input Processing** | Simple input | Stem Layer |
| **Input Data** | OHLC image (grayscale) | OHLC image (grayscale) |
| **Kernel Size** | 5x3 | 5x3 |
| **Pooling Layer Size** | 2x1 | 2x1 |
| **Dropout** | 50% (Only FC) | 50% (Only FC) |
| **Final Activated** | Softmax | Softmax |

Table 1. **Architectural Comparison Between Jiang et al. (2023) and Our CNN Model**.

information that is essential for computing meaningful similarity measures, whereas deeper layers become increasingly invariant and thus less suitable for characterizing pairwise relationships (Mehrer et al., 2020; Klabunde et al., 2025). The $256 \times 16 \times 60$ tensor produced at Stage 3 therefore strikes an effective balance: it is abstract enough to encode multi-day price–volume dynamics, yet sufficiently spatially detailed for similarity computation to capture geometric correspondences between stocks' chart patterns in subsequent procedure of financial network construction in section of Financial Network. From a financial-chart perspective, this stage preserves visual attributes such as candlestick body–wick morphology, short-term momentum shifts, volatility contractions, and local reversal structures—patterns that would be partially lost if deeper layers or globally pooled features were used. These spatially anchored characteristics are essential for constructing a financial network that reflects economically interpretable co-movements across assets.

In contrast, Stage 4 supports the classification pathway. This stage provides the deepest level of abstraction in the CNN, and after its bottleneck blocks the model applies global average pooling to compress each feature map into a single scalar, producing a compact 512-dimensional representation. Global average pooling (GAP) eliminates the need to flatten the feature map into nearly half a million units, thereby preventing parameter explosion in the fully connected layer and introducing robustness to minor spatial distortions caused by the max–min normalization applied to OHLC charts. A dropout layer with probability 0.5 is then applied, followed by a fully connected layer and a softmax activation that yield the probability of a positive return under the chosen labeling scheme. This branch constitutes the mere-CNN forecast model and provides a benchmark for evaluating the incremental value of incorporating cross-stock dependencies toward investment portfolio via subsequent CNN–MST–GCN framework.

To replace the original Leaky ReLU activation used in Jiang et al. (2023), our architecture adopts the Gaussian Error Linear Unit (GELU), following Hendrycks (2016). The choice of GELU is motivated by its smooth, probabilistic gating mechanism, which preserves fine-grained variations in the input and introduces stochastic regularization through its dependence on the Gaussian cumulative distribution. OHLC images contain subtle variations in candlestick geometry and volume patterns, and these nuances are often attenuated or truncated by piecewise-linear activations such as LReLU. In contrast, GELU produces smoother gradients and avoids hard thresholding, resulting in improved sensitivity to small but economically meaningful chart perturbations. This property is critical for financial image forecasting, where micro-structural differences in candlestick shape or wick length may correspond to different behavioral regimes. Moreover, the use of GELU within a pre-activation residual framework has been observed to stabilize optimization by reducing gradient discontinuities, thereby enhancing convergence in deeper networks.

The model is trained using the AdamW optimizer with a weight decay of $10^{-4}$ (Loshchilov and Hutter, 2017), a cosine-annealing learning-rate schedule (Loshchilov and Hutter, 2016), Xavier initializer, and a batch size of 64 for 30 epochs. The initial learning rate is selected through a grid search over the candidate set $\{10^{-3}, 5 \times 10^{-4}, 10^{-4}, 5 \times 10^{-5}, 10^{-5}\}$ with minimum learning rate $10^{-6}$. Cross-entropy loss is used as the objective function, and the model achieving the lowest validation loss is selected. This training procedure has been found to stabilize optimization in deep residual architectures and prevent overfitting on high-variance financial image data. In addition to the standard softmax cross-entropy formulation, we incorporate label smoothing into the CNN classification objective. This modification is known to improve generalization by preventing the classifier from becoming overly confident in its predictions and by mitigating the effect of noisy or imperfect labels—conditions that are especially relevant in financial forecasting, where short-horizon returns are inherently stochastic and subject to microstructure noise. In this study, we apply label smoothing with smoothing coefficient $\epsilon = 0.1$, so that the positive and negative labels are encoded as $(0.9, 0.1)$ and $(0.1, 0.9)$, respectively, instead of the conventional $(1,0)$ and $(0,1)$. This adjustment stabilizes optimization in deep residual architectures, reduces susceptibility to overfitting, and helps the network learn softer decision boundaries in the presence of noisy price dynamics. The modified mechanisms aforementioned are later also adopted in the GCN architecture to ensure consistency across the full CNN–MST–GCN pipeline. Table 2 contrasts our setting against the one in Jiang et at. (2023).

Although early stopping was introduced in Jiang et al. (2023), we ultimately decided not to adopt it because the training dynamics of CNN models exhibit non-monotonic and noisy validation loss trajectories, especially in the early and middle stages of training. As later elaborated in the training performance presented in Section 6.1.1, the validation loss often fluctuates substantially before stabilizing, and the best-performing epoch for both CNN and GCN typically occurs after several oscillations rather than at the first local minimum. Relying on early stopping would therefore risk halting training prematurely at a suboptimal point, leading to underfitted models and weaker

| | **Jiang et al. (2023)** | **Our CNN Model** |
|---|---|---|
| **Optimizer** | SGD and Adam | AdamW |
| **Weight Decay** | N/A | 1e-4 |
| **Initializer** | Xavier | Xavier |
| **Batch Size** | 128 | 64 |
| **Initial Learn Rate** | 1e-5 | Grid Search |
| **Learning Rate Schedule** | Not Specified | Cosine Annealing |
| **Early Stopping** | Patience = 2 | Cancelled |
| **Num Epochs** | Use Early Stop | 30 |
| **Loss Function** | Cross-Entropy Loss | Cross-Entropy Loss |
| **Softmax Trick** | Not Specified | Label Smoothing ($\varepsilon = 0.1$) |

Table 2. **Hyperparameter and Training Configuration Comparison Between Jiang et al. (2023) and Our CNN Model**.

downstream performance. By training for a fixed number of epochs and selecting the checkpoint with the lowest validation loss ex post, it provides a more reliable and consistent way to identify the most generalizable model given the noisy validation patterns observed in practice.

Overall, the proposed CNN integrates domain-informed convolutional design, asymmetric pooling, pre-activation residual learning, computationally efficient bottleneck blocks, and dual-purpose output representations. These components enable the model to extract expressive mid-level visual features suitable for subsequent similarity-based market-topology construction, while simultaneously providing a strong stand-alone predictive signal for short-horizon equity returns.

## 2.4 Embedding Feature Representations

The embedding feature representation serves as the intermediate output that connects the CNN-based visual analysis with the graph-learning stage. This embedding captures the essential temporal–spatial characteristics of price and volume movements, including trend direction, short-term and longer-term volatilities, and perceptual patterns that appear in the visual chart. These embeddings are obtained from 256 channels of our modified ResNet-18 layer after a 2×1 maximum pooling.

We refer $\boldsymbol{I}_{i,t,c}$ to the feature representation of stock $i$ at time $t$ from output channel $c$ extracted from the CNN embedding. Each $\boldsymbol{I}_{i,t,c} \in \mathbb{R}^{H' \times 3\rho}$ is an image embedding (matrix) with a height of 16 pixels (i.e., $H' = H/4$) and a width of 60 pixels, since an OHLC image passes two 2×1 maximum pooling layers and then is extracted.

Conceptually, these embeddings represent the visual signatures of market behavior. For each stock, the embedding reflects the latent representation of return patterns that the CNN has inferred from its historical OHLC images. Figure 5 shows examples of image embeddings of HSBC Holdings plc and Hang Seng Bank Ltd over the period from October 11, 2005 to November 07, 2005.

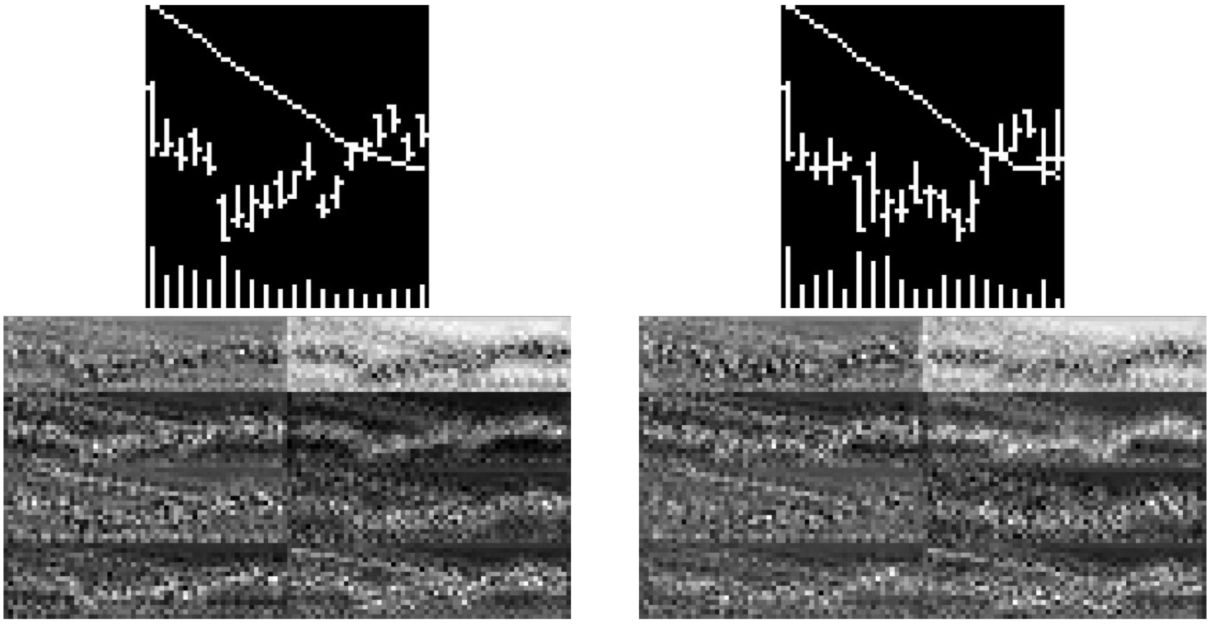

Figure 5. **Raw OHLC charts and CNN image representations for HSBC and Hang Seng Bank**. The left panel shows the raw OHLC price chart of HSBC Holdings plc over 20-trading-day observation period, followed by eight CNN-derived image representations obtained from its feature channels. The right panel presents the raw OHLC chart of Hang Seng Bank Ltd over the same period, along with its eight corresponding channel-aligned image representations. Each row of feature maps is ordered identically across the two stocks, enabling direct visual comparison of how the same CNN channels respond to different price patterns.

## 3 Graph Convolutional Network

The approach suggested by Jiang et al. (2023) and Zhu and Zhu (2024) focuses on exploration of messages embedded in individual OHLC images, but overlooks the fundamental financial insight that asset returns are not independent. Instead, optimal investment depends critically on the correlations among assets when constructing portfolio (Markowitz, 1952). This theoretical foundation motivates the introduction of financial network modeling and GCN to capture the inter-stock dependencies that complement the CNN-based analysis of individual price charts.

In this section, we present a detailed explanation of the GCN architecture design, which builds upon the CNN-derived feature representations and incorporates key financial principles underlying inter-stock relationships.

### 3.1 Portfolio Theory

This subsection introduces the essential financial concepts required to understand our model and interpret its performance. It provides a brief overview of modern portfolio theory (MPT), which views investment as a balance between expected return and risk with consideration of inter-stock relationship. Under this Markowitz's framework, investors aim to construct portfolios that maximize return for a given level of risk, or equivalently, minimize risk for a desired level of return, forming the theoretical basis upon how the proposed model is designed.

Although the investment horizon is set to five trading days, investment decisions can only be made after the completion of the observation period; i.e., once the final closing price has been obtained for constructing the OHLC image. As a result, each position is initiated at the market open on the first trading day and fully liquidated at the market close on the fifth trading day over investment period. The 5-day holding period of a stock $i$ at time $t$ is defined retrospectively as $r_{i,t}$:

$$r_{i,t} = \frac{C_{i,t+4} - O_{i,t}}{O_{i,t}}, \tag{3}$$

where $C_{i,t+4}$ is the close price of stock $i$ at time $t+4$ and $O_{i,t}$ is the open price of stock $i$ at time $t$. Since we apply CNN in return prediction task, it is noteworthy that the expected return $\mathbb{E}(r_{i,t})$, denoted as $\mu_{i,t}$, is based on the OHLC image constructed with the observation period from $t-20$ to $t-1$. For convenience, in this study, we fix time $t$ as the first day of the investment period, with the observation window spanning from $t-20$ to $t-1$; i.e., $t$ ranges from 21 to $T-4$, where $T$ refers to the final trading day in the corresponding dataset.

Suppose the market at time $t$ consists of $S$ stocks. Given a portfolio weight vector $\boldsymbol{w_t} = \left(w_{1,t}, w_{2,t}, \cdots, w_{S,t}\right)^{\mathsf{T}}$ and an expected 5-day holding period return vector $\boldsymbol{\mu_t} = \left(\mu_{1,t}, \mu_{2,t}, \cdots, \mu_{S,t}\right)^{\mathsf{T}}$, the expected portfolio return $\hat{R}_{p,t}$ and risk $\sigma^2_{p,t}$ are expressed as:

$$\hat{R}_{p,t} = \boldsymbol{w_t}^{\mathsf{T}} \boldsymbol{\mu_t}, \quad \sigma^2_{p,t} = \boldsymbol{w_t}^{\mathsf{T}} \boldsymbol{\Sigma}_{\mathbf{t}} \boldsymbol{w_t}, \tag{4}$$

where $\boldsymbol{\Sigma}_{\mathbf{t}}$ denotes the covariance matrix of stock returns at time t, and $\sum_{i=1}^{S} w_{i,t} = 1$. Under this framework, it aims to maximize $\hat{R}_{p,t}$ given $\sigma^2_{p,t}$ (or equivalently, minimize $\sigma^2_{p,t}$ given $\hat{R}_{p,t}$) by finding optimal values of $\boldsymbol{w_t}$ with feature variables $\boldsymbol{\mu_t}$ and $\boldsymbol{\Sigma}_{\mathbf{t}}$. In subsequent sections, our proposed model generalizes this classical mean-variance framework into a graph-learning setting, where CNN embeddings and financial network structures jointly characterize return-risk dynamics. We aim to maximize the total return by calculating the optimal weight $\boldsymbol{w_t^*}$ based on the GCN outputs, which would be explained in the section of Experimental Setup and Evaluation subsequently.

## 3.2 Financial Network

Traditional portfolio optimization methods based on MPT rely heavily on the estimation of covariance matrices $\boldsymbol{\Sigma}_{\mathbf{t}}$ to describe relationships among assets. However, in high-dimensional and dynamically changing markets, these covariance estimates are often unstable and sensitive to sampling noise, leading to unreliable and inefficient portfolio allocations (Jobson and Korkie, 1980; Pástor and Stambaugh, 2000; Kan and Zhou, 2007). To address these limitations while preserving the basic logic of MPT, this research adopts a financial network approach. Peralta and Zareei (2016) and Olmo (2021) show that financial network embodies the core principles of MPT, while leveraging information more efficiently.

This approach leads to a more efficient allocation of capital in portfolio construction.

We denote financial network by $G = \{N, E\}$, which consists of a set of nodes $N$ and a set of edges $E \subseteq N \times N$. The edges represent the linkages between the nodes, and an edge is indicated as $(i, j) \in E$ if there is a link between nodes $i$ and $j$. As a financial network, each stock is represented as a node (i.e., $|N| = S$), and the edges stand for similarity or dependence relationships between stocks. The network structure can be conveniently depicted by a $S \times S$ adjacency matrix $\boldsymbol{A} = [a_{ij}]$, where each element $a_{ij} \neq 0$ indicates the presence of a link between nodes $i$ and $j$; i.e., $a_{ij} \neq 0$ whenever $(i, j) \in E$. In this research, we apply an undirected, weighted network to describe the linkages existed in the financial market, which has following properties: $\boldsymbol{A} = \boldsymbol{A}^\top$ (no causal association between the nodes; i.e., $(i, j) \in E \Leftrightarrow (j, i) \in E$), $a_{ij} \in [-1, 1]$ (weighted), $a_{ii} = 0$ (no self-loop), and $\boldsymbol{A} \succcurlyeq 0$ (positive semidefinite).

In this research, we define the linkage between two stocks as the similarity between their local patterns captured by our CNN model. Narayanaswamy and Gowda (2024) demonstrate that cosine similarity effectively captures image information when applied within the VGG19 and Gated Recurrent Unit (GRU) frameworks. Although Ma and Yuan (2024) and Ma et al. (2024) apply similarity calculation on the time-series information of stock prices directly, we prefer to compute similarities between feature representations instead of OHLC images with several reasons. First, raw image comparisons may capture irrelevant features or noises rather than the price patterns of interest. Embeddings map the images into a "latent space" where distances or similarities are more meaningful. For example, an embedding may reflect "this stock shows breakout behavior" rather than just color or positional similarity. Second, this approach focuses on task-relevant features. Our CNN is trained for predicting the most profitable stocks. Its embeddings therefore emphasize patterns relevant to return dynamics. Similarity on embeddings is hence more aligned with the task accomplished by GCN.

We here define Frobenius cosine similarity between two feature representations of stock $i$ and $j$ from an output channel $c$ at time $t$, which denoted as $\boldsymbol{I_{i,t,c}}$ and $\boldsymbol{I_{j,t,c}}$ respectively, as:

$$\cos(\theta_{(i,j),t,c}) = \frac{\langle \boldsymbol{I_{i,t,c}}, \boldsymbol{I_{j,t,c}} \rangle_F}{\left\|\boldsymbol{I_{i,t,c}}\right\|_F \left\|\boldsymbol{I_{j,t,c}}\right\|_F}, \tag{5}$$

where $\langle \cdot,\cdot \rangle_F$ is Frobenius inner product, $\|\cdot\|_F$ means Frobenius norm, and $\cos(\theta_{(i,j),t,c})$ is the similarity score between $\boldsymbol{I_{i,t,c}}$ and $\boldsymbol{I_{j,t,c}}$.[1] Geometrically, $\theta_{(i,j),t,c}$ can be viewed as an angle between two matrices in a high-dimensional $\mathbb{R}^{H' \times 3\rho}$ space. When $\theta_{(i,j),t,c}$ is closed to 0, the two feature representations are highly aligned. When $\theta_{(i,j),t,c}$ is closed to $\pi/2$, there is no obvious visual relationship between two feature representations. When $\theta_{(i,j),t,c}$ is closed to $\pi$, these two feature representations display opposite patterns. Figure 6 exhibits three specific image embeddings with their

[1] Our equation is mathematically equivalent to the equation (7) derived in Narayanaswamy and Gowda (2024) with setting the weighted coefficients as constant.

similarity scores as examples. To summarize the relationships between two stocks across all of the channels, we define the similarity score between two stocks $i$ and $j$ at time $t$, symbolized as $m_{(i,j),t} \in [-1,1]$, by average out their similarity scores of image embeddings $\cos(\theta_{(i,j),t,c})$ over channels:

$$m_{(i,j),t} = \frac{1}{256}\sum_{c=1}^{256} \cos(\theta_{(i,j),t,c}). \tag{6}$$

With respect to the example shown in Figure 5, when the Frobenius cosine similarity score between the two raw OHLC images is 0.0181 only, the similarity score between the feature embeddings rises up to 0.5919, indicating the capability of Frobenius cosine similarity to suppress noise and to refine the underlying trend-related information against spatial variance.

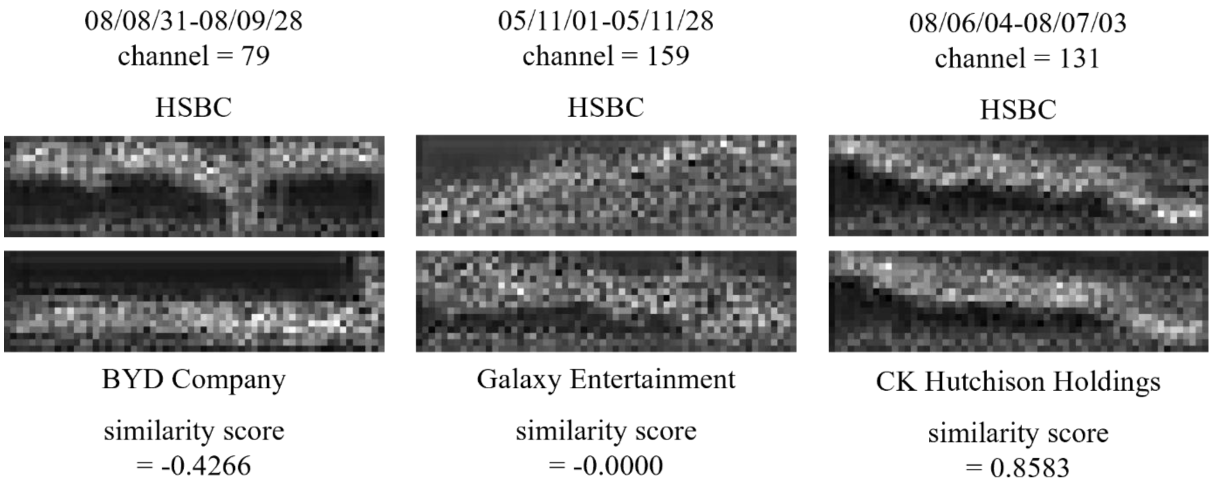

Figure 6. **Examples of CNN image representation pairs with different similarity scores**. The figure shows three pairs of CNN image representations illustrating cases where the cosine similarity between two representations is highly negative (left pair), approximately zero (middle pair), and highly positive (right pair). Each pair visualizes how the learned feature maps can exhibit contrasting, neutral, or strongly aligned patterns across different stocks in the same channel over the same observation window.

In addition to capturing the price patterns across stocks, we also generate two sets of images of the same stock with using time windows that overlap halfway for comparison. As illustrated in Figure 7, the overlapping regions—appearing on the right side of the image in left panel and the left side of the one in right panel—exhibit nearly identical visual structures, indicating similar price and volume dynamics during those periods. Nevertheless, due to the max-min scaling with different price ranges, these patterns appear in different pixel locations in raw OHLC images, resulting in a similarity score with 0.0020. In contrast, the similarity score raises up to 0.3672 for the image representations between these two observation windows for the HSBC stock, indicating the capability of capturing similar price dynamics even when those patterns appear at shifted spatial locations.

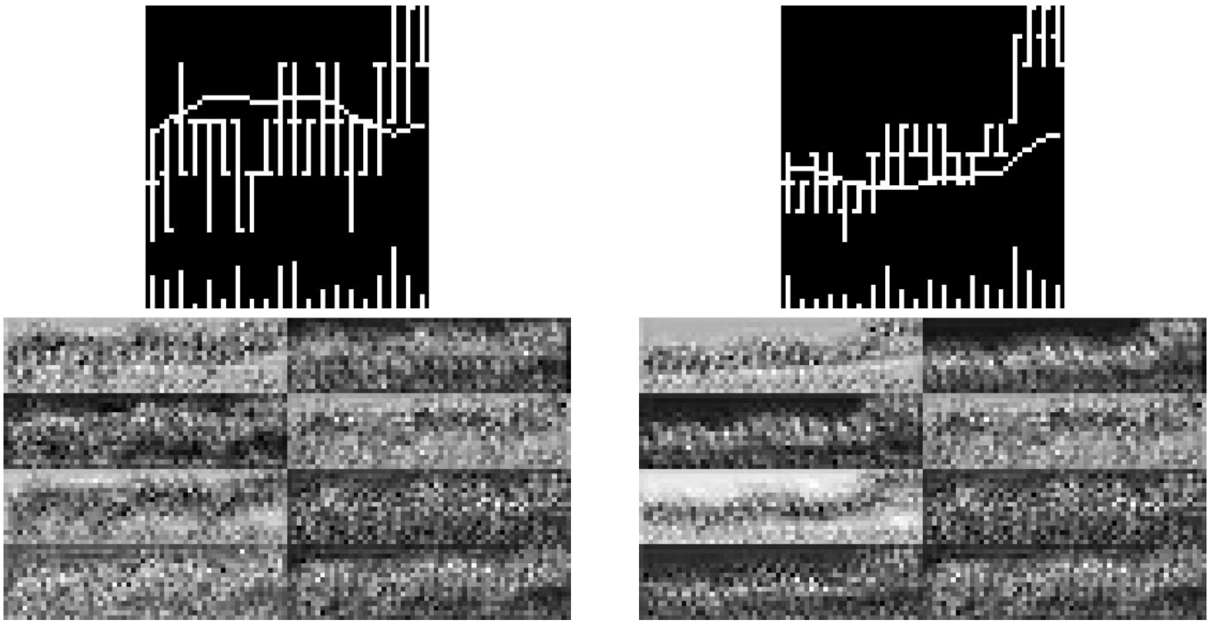
Figure 7. **Comparison of overlapping time-window OHLC images and their CNN image representations**. The left panel displays the OHLC price chart of HSBC Holdings plc over the period May 16, 2005 to June 12, 2005, with eight CNN-derived image representations shown underneath. The right panel presents the OHLC chart for the subsequent window May 30, 2005 to June 26, 2005, along with its corresponding eight image representations. For both panels, the feature maps are arranged in the same channel order, enabling direct comparison of how each CNN channel responds to different but overlapping price patterns across time.

Collecting all of pairwise similarity scores $m_{(i,j),t}$ at time $t$, we have a similarity matrix $\boldsymbol{M}_t = \left[m_{(i,j),t}\right] \in \mathbb{R}^{S\times S}$, which describes the stock relationships in the financial market at time $t$. To satisfy the mathematical properties of the adjacency matrix $\boldsymbol{A}$, we employ the distance measure suggested in Mantegna (1999) and obtain the distance matrix $\boldsymbol{D}_t = \left[d_{(i,j),t}\right] \in \mathbb{R}^{S\times S}$, where $d_{(i,j),t} = \sqrt{2(1 - m_{(i,j),t})}$. In order to further mitigate noise and retain only the most meaningful connections, as emphasized in Millington and Niranjan (2021), we utilize Kruskal's algorithm to implement a topological filtration method called Minimum Spanning Tree (MST). As a result, it deletes weaker linkages in $E$ to reserve only $S-1$ the most important edges in our financial network. Let $\boldsymbol{\Phi_t} = \left[\phi_{(i,j),t}\right] \in \mathbb{R}^{S\times S}$ be the MST matrix reduced from $\boldsymbol{D}_t$ using this topological filtration method. After reserving the important edges, we replace the non-zero values of $\phi_{(i,j),t}$ by $m_{(i,j),t}$ to reflect the true relationships among nodes. Since MST matrix is an upper-triangular matrix, we symmetrize the replaced upper-triangular matrix to form the MST-based adjacency matrix $\boldsymbol{A_t} = \left[a_{(i,j),t}\right] \in \mathbb{R}^{S\times S}$ by adding its transpose, thereby constructing an undirected financial network $G_t$ at time $t$:

$$a_{(i,j),t} = \begin{cases} m_{(i,j),t}, & \text{if } \phi_{(i,j),t} > 0; \\ m_{(j,i),t}, & \text{if } \phi_{(j,i),t} > 0\ ; \\ 0, & \text{otherwise.} \end{cases} \tag{7}$$

Figure 8 illustrates the financial networks built by similarity matrix $\boldsymbol{M}_t$ and MST-based adjacency matrix $\boldsymbol{A}_t$, respectively. Each stock is represented as a node in the financial network, and the edges stand for the correlations between the nodes. Since the financial network topology is fully determined by cross-sectional market information, training and test samples are indexed by time stamp $t$ only.

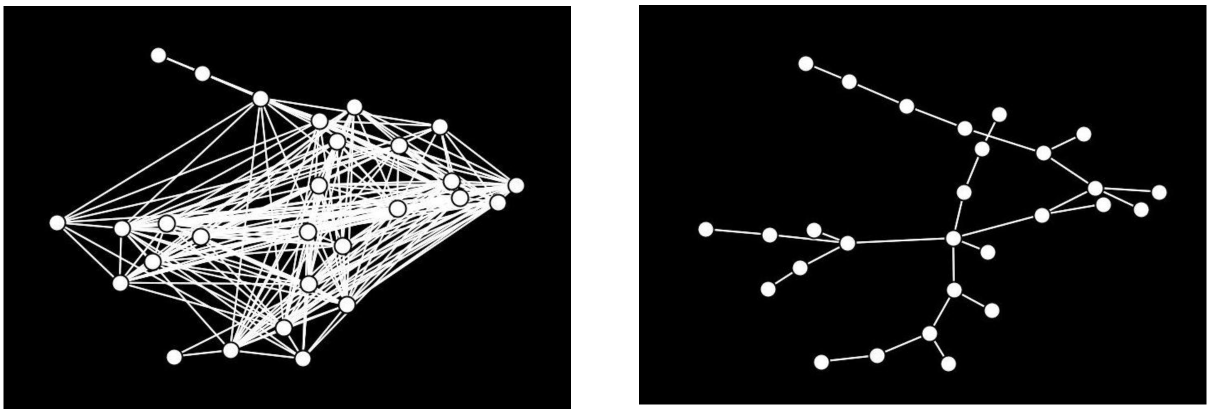

Figure 8. **Financial networks constructed from similarity and MST-based adjacency matrices**. The left panel shows the financial network constructed from the similarity matrix of CNN image representations, resulting in a densely connected graph. The right panel displays the network derived from the MST-based adjacency matrix for the same observation window, which is from October 11, 2005 to November 7, 2005.

MST transformation achieves several advantages in our subsequent GCN task (Liu et al., 2023). First, this filtering process enhances interpretability of market topology shown in the MST financial network. As revealed in left panel of Figure 8, the network derived from the raw similarity matrix exhibits dense and noisy connectivity, where numerous weak or spurious edges obscure the true structural relationships among stocks displayed in the image. In contrast, as displayed in right panel, the MST-based network forms a sparse, tree-like topology (i.e., $\delta$-hyperbolicity) that highlights the dominant linkages within the market. This structure reveals clear hierarchical clusters of stocks, enabling a more interpretable and stable representation of market dependencies presented in the image. Second, the financial network constructed from the MST matrix helps reduce the risk of overfitting in the subsequent GCN learning process. Beyond decreasing the likelihood of learning spurious correlations among stocks, it also lowers model complexity by providing fewer propagation paths due to the reduced number of edges. This constraint limits the model's capacity to memorize noise and acts as a form of structural regularization, analogous to pruning in neural networks.

## 3.3 GCN Architecture

Since our research bridges three domains — mathematics, computer vision, and finance, we explain our work through the insights gained from each of these fields.

- Mathematics: Graph Theory and Geometric Space

As illustrated in Figure 9, although the financial network constructed from the MST differs from a conventional image, it can also be interpreted as a form of non-Euclidean image that conveys structural visual information (Liu et al., 2023). The left panel shows the operation of the convolutional kernel scanning an OHLC image, where pixels are arranged on a regular Euclidean grid and the receptive field moves uniformly across spatially ordered coordinates. In contrast, the right panel displays a financial network image alongside its corresponding MST matrix, where nodes represent stocks and edges indicate relational intensities. Unlike the fixed and evenly spaced pixel neighborhoods in Euclidean space, the MST-based financial network forms an irregular and hierarchical topology in which connections vary dynamically across nodes, reflecting heterogeneous dependencies among assets. This irregular structure cannot be effectively captured by standard CNN, whose convolution relies on uniform local connectivity. Instead, GCN extends convolution to non-Euclidean image data domains by aggregating information from neighboring nodes defined by the (financial) network topology rather than across evenly spaced pixels. Undoubtedly, it adapts with the $\delta$-hyperbolicity exhibited by the MST market topology images.

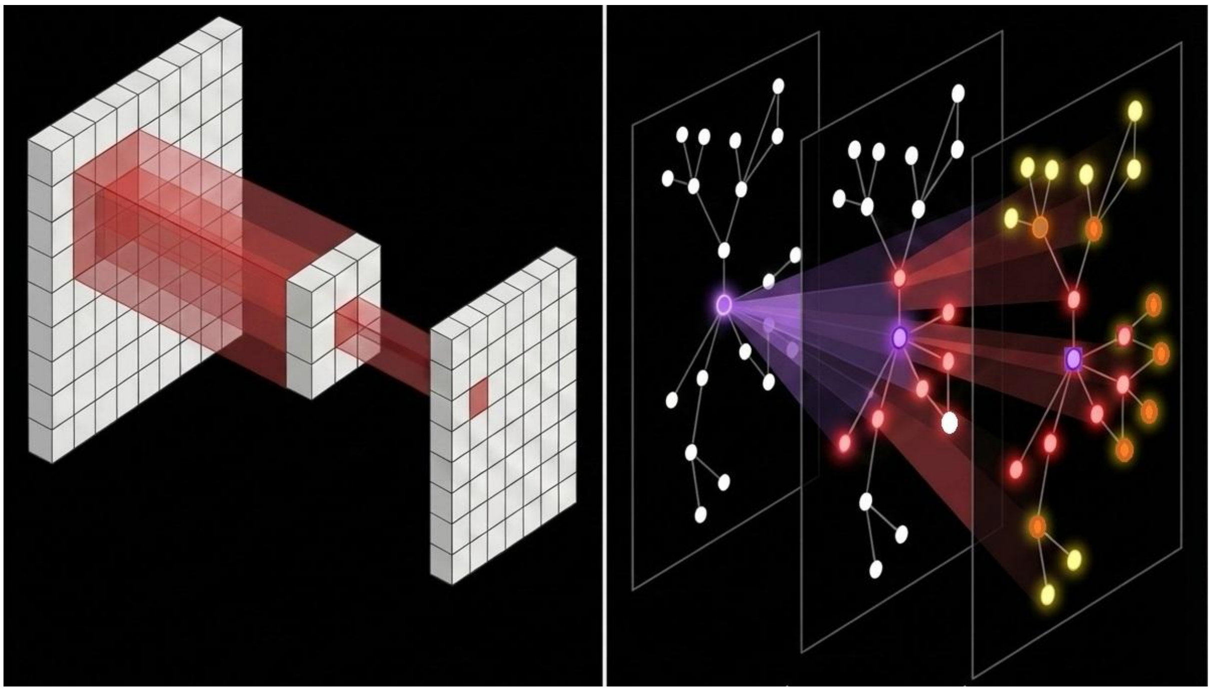

Figure 9. **Convolution in Euclidean vs. Non-Euclidean Geometry: CNN on Image Grids and GCN on Financial Networks**. The left panel illustrates the CNN learning process, where a convolutional kernel scans an image pixel-by-pixel over a regular Euclidean grid. The right panel depicts the GCN learning process on the MST-based financial network. The left layer represents the input graph with initial node features, the middle layer corresponds to the first GCN layer performing one-hop aggregation, and the right layer corresponds to the second GCN layer performing two-hop aggregation. In this visualization, the violet node is the example node being updated, the red nodes are its one-hop neighbors, and the orange nodes are its two-hop neighbors. Because these neighborhoods arise from the irregular connectivity of the graph rather than a fixed spatial grid, the GCN conducts convolution in a non-Euclidean (hyperbolic) geometric space, enabling the extraction of relational patterns that standard Euclidean convolution cannot capture.

- Computer Vision: Learning Market Topology through Graph Representation

From a computer vision perspective, the GCN can be regarded as performing visual learning on a graph-structured image of the market. Figure 10 illustrates this interpretation, showing a simplified financial network where each node is associated with its own stack of image embeddings—learned visual features from the CNN representing a stock's price behavior; i.e., collective visual co-movements across the stock market. The edges describe how these nodes are spatially arranged within the network, defining topological rather than grid-based neighborhoods (Cao et al., 2022; Peng et al., 2024). When the GCN processes this input, it perceives the financial network as a visual object whose "pixels" are stocks and whose spatial layout is determined by economic relationships. Through its topology-aware convolution, the GCN captures structural textures and relational patterns that emerge across the market graph, providing a computer vision perspective on how interconnected assets evolve collectively.

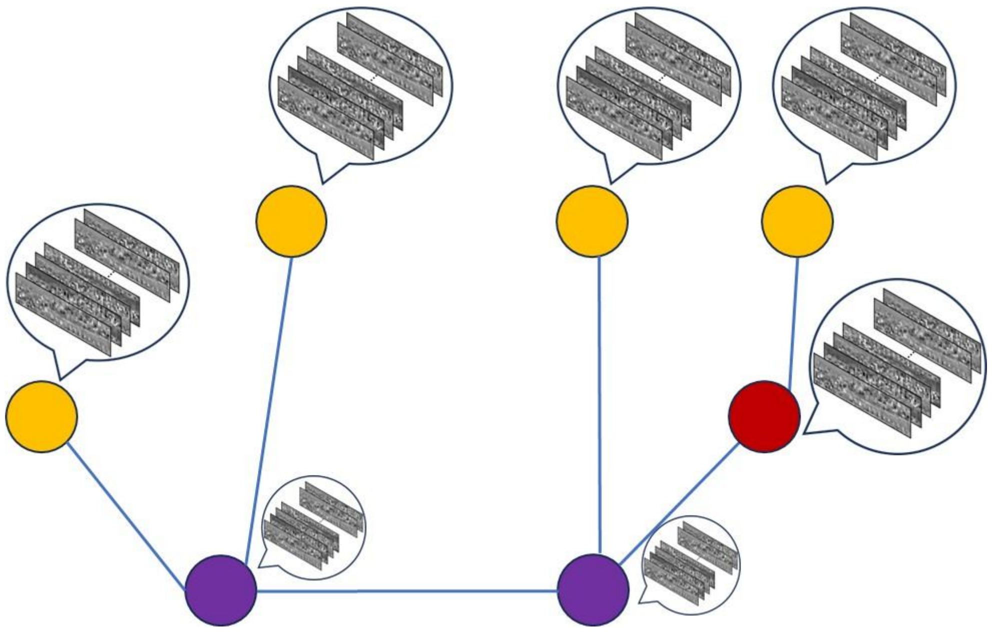

Figure 10. **Visualization of a graph-structured "image" viewed by GCN Using CNN-derived node features**. The figure illustrates how GCN processes the financial network as a graph-structured image. Each node contains information based on a stack of 256-dimensional image embeddings learned from the CNN, and the GCN updates every node's representation by aggregating feature information from its topological neighbors. The node colors reflect their centrality within the network, indicating their relative importance or connectivity. This visualization highlights that the GCN focuses on the relational neighborhood structure—rather than geometric proximity—when interpreting the market graph.

- Finance: Generalization of Mean-Variance Principle

Conceptually, this fusion of CNN-derived node features and MST-based network structure generalizes the classical MPT mean-variance principle within a graph-learning context. The node embeddings encapsulate each stock's expected return characteristics, while the network edges encode covariance-like dependencies that shape collective market dynamics; i.e., $\boldsymbol{\mu}_t$ and $\boldsymbol{\Sigma}_t$ in equation (4) respectively. Our GCN model thus integrates local information with global relational cues, performing a data-driven analogue of the return-risk trade-off that underlies modern portfolio theory. In doing so, it extends traditional financial reasoning into a visual deep-learning framework—one that learns to "see" the balance between individual stock potential and systemic interaction through the geometry of the financial network itself.

In following subsections, we provide a concise overview of our GCN's architectural design and the procedures used to train it. In Figure 11, the left diagram presents our CNN setting, and the right diagram illustrates our GCN setting.

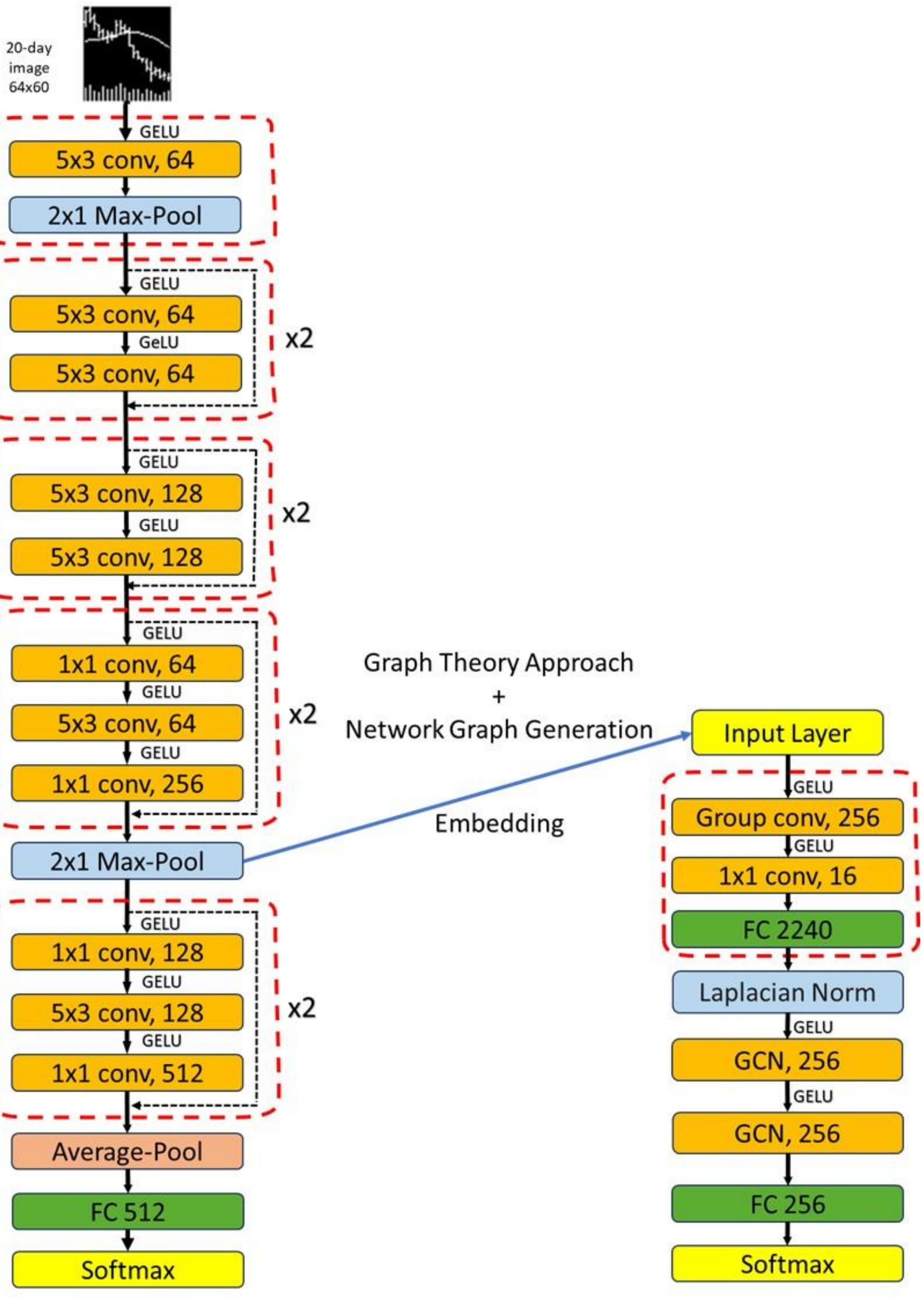

Figure 11. **GCN model architecture**.

### 3.3.1 Dimension Reduction

The raw 256×16×60 feature map for each node is a high-dimensional 3D tensor, which presents two major problems: a GCN expects a 1D feature vector per node, not a 3D map, and this tensor's massive size (245,760 elements) would lead to extreme computational cost and optimization challenges. A naïve flattening would create a sparse, oversized vector and completely destroy the valuable 2D spatial and temporal structure captured by our CNN. We need to distill this 3D visual information into a compact 1D "visual abstract", so that GCN then "sees" the global picture by taking all these individual "visual abstracts" (one for each node) and learning how they are related across the market's non-Euclidean topology.

Therefore, we introduce a learnable dimensionality reduction block to distill this visual information. This block consists of a depthwise separable convolution followed by a fully connected layer. The convolution's 4×3 kernel and 2×3 stride are deliberately chosen to align with the 3-pixel-per-day visual grammar of the OHLC images, thus preserving meaningful local patterns during downsampling. It reduces the size of image embeddings from 256×16×60 to 256×7×20.[2] The pointwise step then acts as an efficient bottleneck, compressing the 256 channels to 16. A final fully connected layer with 2,240 input channels projects the output of the depthwise separable convolution into a dense, 256-dimensional vector.[3] This architecture successfully transforms 256 high-dimensional 3D visual feature map into a compact 1D node embedding with 256 dimensions, providing the GCN with a computationally efficient and semantically rich input. Training configurations follow the setting presented in Tables 1 and 2.

### 3.3.2 Normalization and Convolution

In previous steps, we collect the 1D node embedding for each node and form a node feature matrix $\boldsymbol{X} \in \mathbb{R}^{S\times 256}$, where the $i$th row is the $i$th stock's 1D node embedding. The node feature matrix $\boldsymbol{X}$ and the MST-based adjacency matrix $\boldsymbol{A}$ are the inputs of the first GCN layer. Since the training target is the same for both of CNN and GCN in our research, the node feature matrix built from the image representations extracted based on CNN model trained to fitting the same target. Therefore, $\boldsymbol{X}$ is intrinsically aligned with the mission of GCN and improves the training efficiency. Following the concept of Berg et al. (2017), we update the node features with the information from the neighbors of the nodes by computing:

$$\boldsymbol{Z} = \sigma(\boldsymbol{A}\boldsymbol{X}\boldsymbol{\Omega} + \boldsymbol{b}), \tag{8}$$

where $\boldsymbol{\Omega} \in \mathbb{R}^{256\times 256}$ is the learnable weight matrix for the GCN layer, $\boldsymbol{b} \in \mathbb{R}^{256}$ is the bias vector, $\sigma(\cdot)$ is GELU activation function, and $\boldsymbol{Z} \in \mathbb{R}^{S\times 256}$ is the output of the first GCN layer.[4] Nevertheless, equation (8) encounters two critical problems. First, recalling the mathematical properties of MST-based adjacency matrix $\boldsymbol{A}$ aforementioned, its nodes lose the connection with self-features to prevent

[2] Height= $\frac{16-4}{2} + 1 = 7$; width= $\frac{60-3}{3} + 1 = 20$. As shown by the equation, 4×3 kernel makes sure that all rows and columns of an image embedding are fully scanned by the convolution kernel.
[3] 16×7×20=2,240.
[4] Note that here is broadcasted addition for bias.

self-loop. However, GCN requires each node to gather information from its neighbors and itself. Second, as shown in Figure 8, the central nodes have the greatest number of neighbors, but the peripheral nodes have only one neighbor. For example, a node with 10 neighbors would have a feature vector 10 times larger in magnitude than a node with 1 neighbor. During backpropagation, the gradient for a node requires summing the gradients from all of its neighbors that sends information to the node. In other words, the central nodes face the issue of gradient explosion, while the peripheral nodes encounter the issue of gradient vanishing.

To solve the issues, we modify the MST-based adjacency matrix $\boldsymbol{A}$ by following two steps before implementing GCN learning process in accordance with the procedures presented in Mehrkanoon (2021) and Jiang et al. (2024). First, we add self-connection by summing identity matrix as:

$$\widetilde{\boldsymbol{A}} = \boldsymbol{A} + \lambda \boldsymbol{I_S}, \tag{9}$$

where $\lambda$ is a learnable parameter that represents the relative importance of self-features to its neighbors. Second, we stabilize GCN propagation by converting the self-looped adjacency matrix into a degree-adjusted operator through Laplacian normalization that prevents activations from exploding, suppress the dominance of high-degree nodes, and ensure that information is exchanged symmetrically across edges. We symmetrically normalize $\widetilde{\boldsymbol{A}}$ using the inverse square root of degree matrix to act as the weights in computing the weighted average information presented in $\widetilde{\boldsymbol{A}}$, where the degree matrix $\widetilde{\boldsymbol{\Delta}} \in \mathbb{R}^{S \times S}$ denotes the number of neighbors connected to the nodes in the self-looped network (so $\lambda$ is also counted):

$$\widehat{\boldsymbol{A}} = \widetilde{\boldsymbol{\Delta}}^{-1/2} \widetilde{\boldsymbol{A}} \widetilde{\boldsymbol{\Delta}}^{-1/2}, \tag{10}$$

where the first inverse square root matrix $\widetilde{\boldsymbol{\Delta}}^{-1/2}$ scales the row of $\widetilde{\boldsymbol{A}}$ and the second one scales the column of $\widetilde{\boldsymbol{A}}$. It effectively transforms the aggregation from a node-degree–scaled sum into a balanced average, thereby equalizing the magnitude of updates across nodes. Spectrally, this normalization guarantees that all eigenvalues of the propagation operator fall within a bounded range, enabling stable multi-layer message passing without amplification or vanishing of signals.[5]

Regarding our special data structures of (normalized) adjacency matrix $\widehat{\boldsymbol{A}}$ and node feature matrix $\boldsymbol{X}$, the relationships between the samples (node features) of $\boldsymbol{X}$ are described by $\widehat{\boldsymbol{A}}$ instead of training samples are i.i.d. and drawn from the same distribution. These graphs $(\boldsymbol{X}, \widehat{\boldsymbol{A}})$ differ substantially across trading days due to shifts in market regime, volatility, and topology. Therefore, the node-feature

[5] Regarding the graph theory, when the eigenvalue centrality of a node is greater than 1, its signal grows exponentially and thus is subject to gradient explosion during backpropagation. On the contrary, when the eigenvalue centrality of a node is less than 1, its signal shrinks exponentially and thus is subject to gradient vanishing. Although we cannot perfectly set the eigenvalue centralities as 1 for all nodes, a symmetric normalized Laplacian matrix has a wonderful mathematical property where its eigenvalues are bounded in the range [-1, 1] and usually close to 1. As a result, our financial network is no longer subject to the issue of gradient explosion, and its signal magnitude is preserved since the maximum eigenvalue are preserved near 1 (rather than being crushed to 0), especially for the peripheral nodes.
In addition to GCN backpropagation, Peralta and Zareei (2016) also discuss how the maximum eigenvalue plays a critical role in determination of optimal weights in a well-diversified portfolio. Without a doubt, the topological operation here affects the final optimal weights determined in our generalized mean-variance approach.

distributions across graphs are highly heterogeneous. As a result, batch normalization produces unstable or misleading normalization statistics because it mixes information across graphs using batch-level statistics, which is especially problematic when each graph represents a specific day's market state with its own structural and statistical identity. We apply graph normalization (GraphNorm) advocated by Cai et al. (2021), which normalizes within each graph independently rather than across the batch. First, this isolates the noise of one trading day from another, ensuring that the normalization is tailored to the specific market conditions of that day. Second, since MST graphs are extremely irregular (one high-degree hub with many low-degree leaves), node-feature magnitudes vary widely across nodes. GraphNorm re-centers and re-scales features per graph in a learnable partial magnitude such that respects this irregular topology, preventing hub nodes from dominating propagation and preventing leaf nodes from collapsing into near-zero feature space. These properties collectively make GraphNorm more robust, more stable, and more aligned with the dynamics and geometry of our MST-based financial networks. Third, Cai et al. (2021) demonstrate that the shift operation in GraphNorm serves as a preconditioner for $\widehat{\boldsymbol{A}}$. This preconditioning makes the singular value distribution of $\widehat{\boldsymbol{A}}$ smoother. Consequently, the resulted smoother optimization landscape allows GCN to converge significantly faster than it would with BatchNorm.

Suppose $\boldsymbol{\Psi} = \widehat{\boldsymbol{A}}\boldsymbol{X} = [\psi_{ij}] \in \mathbb{R}^{S\times 256}$, and GraphNorm is applied to $\boldsymbol{\Psi}$; i.e., the operations are performed for each feature dimension $j$ (column) across all $S$ stock nodes (rows) in a single market network:

$$\bar{\bar{\psi}}_{ij} = \gamma_j \cdot \frac{\psi_{ij} - \alpha_j \upsilon_j}{\sigma_j} + \beta_j, \tag{11}$$

where $\gamma_j$, $\alpha_j$, and $\beta_j$ are learnable parameters, feature mean $\upsilon_j = (1/S)\sum_{i=1}^{S} \psi_{ij}$ is subtracted, standard deviation of the shifted value $\sigma_j = \sqrt{(1/S)\sum_{i=1}^{S}\left(\psi_{ij} - \alpha_j \upsilon_j\right)^2 + \epsilon}$ divides, and $\mathrm{GraphNorm}(\widehat{\boldsymbol{A}}\boldsymbol{X}) = [\bar{\bar{\psi}}_{ij}]$.[6] The core of GraphNorm does not only normalize within a single financial market network sample, but also apply a learnable mean retention coefficient $\alpha_j \in [0, 1]$ to scale the feature mean for each feature $j$. It introduces two advantages. First, equation (7) removes noise but retains hierarchical clusters, and hence equation (10) stabilizes the process of backpropagation during learning. Such hierarchical clusters reflect the topological information of the financial market, and a standard mean subtraction might remove the "intensity" of these clusters.[7] Second, when we put our eyes back to the natures of OHLC charts and thus their embedding representations, these images encode absolute signals such as strength of a breakout or the magnitude of trend and relative signals like the trend strength compared to other stocks that day. Subtracting the full mean, just as done in BatchNorm,

[6] $\epsilon$ is a small constant for numerical stability.

[7] For example, leaf nodes tend to have less aggregated information. Subtracting the full mean $\upsilon_j$ disproportionately affects them because their embeddings are often lower magnitude. Nonetheless, a leaf stock with weak CNN signals should not be heavily "pushed" toward the mean. A tunable $\alpha_j$ prevents over-correction and allows the model to learn the correct scaling for nodes with different structural roles, so that the peripheral stocks are not artificially inflated or suppressed.

forces all features to be treated as relative only, eliminating absolute-level predictive information. The learnable coefficient $\alpha_j$ allows the model to choose how much absolute information to keep within each feature column.

Since our graph image is structured from MST, the number of GCN layers directly determines how far information can propagate across the network as each layer aggregates information from one hop of neighbors. To estimate how many hops are needed for meaningful information flow, we rely on the Average Shortest Path Length (ASPL) of the MST. ASPL measures the typical number of steps required to move between two randomly chosen nodes in the network, providing a robust indicator of the graph's effective connectivity:

$$ASPL = \frac{1}{S(S-1)} \sum_{i \neq j} dist(i,j), \tag{12}$$

where $dist(i,j)$ is the distance of the unique path connecting node $i$ and node $j$ in the (MST) financial network $G$. In the context of financial networks, ASPL reflects the average structural proximity between stocks. By designing the GCN depth with consideration of ASPL, we ensure that information is propagated across the market network at a scale that matches the underlying topology: sufficiently deep to capture the essential cross-stock dependencies, but not so deep that it over-smooths node features (Coşkun et al., 2025).[8] In our research, the unweighted ASPL for the training set is 5.24, indicating that information typically requires more than five node-to-node hops to propagate across the entire financial network.[9] We deliberately restrict the GCN depth to two layers to avoid over-smoothing and to process meaningful stock dependencies. In addition to avoidance of the issue of over-smoothing, this choice aligns with well-documented equity market structure. Sectoral, supply-chain, and co-movement effects are strongest within one to two hops, while dependencies beyond three hops tend to be weak or dominated by noise. As a result, 2-layer GCN is the most effective setting: it is sufficiently expressive to learn meaningful cross-stock interactions, yet shallow enough to avoid over-smoothing and to respect the localized nature of financial transmission channels within the MST topology.

In order to further prevent the issue of over-smoothing, we also introduce the residual connection in the GCN layers by following Kawamoto et al. (2018) and Chen et al. (2025b). Since our MST is sparse and highly uneven—containing many leaf nodes and a few central hubs, standard GCN propagation can easily distort or wash out the original CNN-derived node features, especially after multiple message-passing steps. Zhang et al. (2023) argues that residual connections allow each node's initial embedding to flow directly into deeper layers. It prevents over-smoothing and preserves stock-specific

[8] It is subject to a phenomenon where repeated Laplacian aggregation drives node representations to become nearly identical, erasing the discriminative features needed for prediction.

[9] The weighted ASPL computed by equation (12) is 5.49 for our training set, indicating that the most of the constituent stocks are weakly (but positively) associated as long as its value is greater than that of unweighted ASPL. Nonetheless, ASPL is used for deciding the number of GCN layers; i.e., how many "hops" should be consider given the average neighborhood of the networks. Consequently, we here consider unweighted ASPL instead of weighted one.

information extracted from the OHLC image features. This is essential in our task because the CNN embeddings capture fine-grained visual patterns that should not be overwritten by neighborhood aggregation. Residual links also improve gradient flow, making training more stable under day-to-day changes in graph structure and node-feature distributions. Moreover, they allow the GCN to learn refinements to the CNN embeddings—adding relational corrections from the MST—rather than forcing the model to relearn or distort the core predictive features. As a result, residual connections produce more expressive, more stable, and more financially meaningful node representations for return prediction. Since residual connections are introduced, we fix the output dimensions of all of the GCN layers the same as the input dimension; i.e., 256.

Consequently, we modify the equation (8) into equation (13) by incorporating the normalized adjacency matrix in GraphNorm and implementing residual connections and pre-activation:

$$\boldsymbol{Z}^{(1)} = \sigma\left[\text{GraphNorm}\left(\widehat{\boldsymbol{A}}\boldsymbol{X}\right)\right]\boldsymbol{\Omega}^{(1)} + \boldsymbol{b}^{(1)} + \boldsymbol{X}, \tag{13}$$

where the superscript indicates the layer number of GCN.

Now, we apply GCN learning with equations (13) and (14) with the same setting as CNN presented in Section 2.3, including GELU activation and the training configurations presented in Table 2, except using batch size equal to 16, since a single network data consists of $S$ stocks' information. Regarding the output of the first GCN layer with 256 output channels, each row of $\boldsymbol{Z}^{(1)}$ means the updated feature vector of a single node (stock) that contains a learnable, normalized-and-averaged "message" from its neighbors, combined with its own original features from $\boldsymbol{X}$. Technically, each node's features are now "aware" of its 1-hop neighbors. Next, we stack this updated information with the second GCN layer to make every node aware of their 2-hop neighbors, and we have:

$$\boldsymbol{Z}^{(2)} = \sigma\left[\text{GraphNorm}\left(\widehat{\boldsymbol{A}}\boldsymbol{Z}^{(1)}\right)\right]\boldsymbol{\Omega}^{(2)} + \boldsymbol{b}^{(2)} + \boldsymbol{Z}^{(1)}, \tag{14}$$

where $\boldsymbol{Z}^{(2)} \in \mathbb{R}^{S\times 256}$ is the output of the second GCN layer with 256 output channels, $\boldsymbol{\Omega}^{(2)} \in \mathbb{R}^{256\times 256}$ is the learnable weight matrix for the second GCN layer, and $\boldsymbol{b}^{(2)} \in \mathbb{R}^{256}$ is the learnable bias vector for the second GCN layer.[10] Note that both of the threshold-based target and quantile-based target, defined in Section 2.2, are also used in GCN learning, so we would have two GCN models finally.

### 3.3.3 Full Connected Layer and Softmax Output

To convert these learned node embeddings into actionable predictions, we apply a final fully connected layer, which maps the GCN output of each node into a vector of two logits (representing negative and positive classes). Importantly, this fully connected layer is shared across all nodes and operates independently on each row of the GCN output matrix. In other words, although the model processes the entire graph simultaneously, it produces class-specific scores for each stock, reflecting a node-specific

[10] Recent literatures mainly discuss two types of GCNs, spectral-based and spatial-based. Here we implement former setting. For more elaboration of the spectral-based manner applied, see Chen et al. (2020) and Zhang et al. (2023).

prediction. The output logits from the fully connected layer are subsequently passed through a softmax activation function, yielding a probability matrix $\hat{\mathbf{Y}} \in \mathbb{R}^{S\times 2}$.

At each trading window, we pass $\boldsymbol{A_t}$ and $\boldsymbol{X_t}$ that generated from the image embeddings over the observation window from $t-20$ to $t-1$ into the GCN model trained and reap the 2-hop node feature matrix $\boldsymbol{Z_t^{(2)}}$. We get the probabilities of next 5-day return exceeding 0.3% for threshold-based target (or, being top 30% among peers for quantile-based target) for all stocks through the final layer of GCN:

$$\boldsymbol{L_t} = \boldsymbol{Z_t^{(2)}}\boldsymbol{\Omega^{FC}} + \boldsymbol{b^{FC}}, \tag{15}$$

where $\boldsymbol{\Omega^{FC}} \in \mathbb{R}^{256\times 2}$ is the weight matrix for this classification layer, $\boldsymbol{b^{FC}} \in \mathbb{R}^{1\times 2}$ is the bias vector, and $\boldsymbol{L_t} = \left[l_{ij,t}\right] \in \mathbb{R}^{S\times 2}$ is the resulting logit matrix at time $t$, where $l_{(i,0),t}$ is the logit score for negative class for stock $i$ (i.e., $r_{i,t} < 0.3\%$ for threshold-based label and $rank\left(r_{i,t}\right) > 0.3S$ for quantile-based label) and $l_{(i,1),t}$ is the logit score for positive class for stock $i$ at time $t$. The softmax function converts these logit scores into probabilities. For a specific stock $i$ and class $j \in \{0,1\}$ at time $t$, the predicted probability is:

$$\hat{y}_{ij,t} = \frac{e^{l_{ij,t}}}{\sum_{k=0}^{1} e^{l_{ik,t}}}. \tag{16}$$

The probability matrix $\hat{\mathbf{Y}}_{\mathbf{t}} = [\hat{y}_{ij,t}]$ collects the predictions, where the first column is the probabilities of the stocks will get a negative result ($r_{i,t} < 0.3\%$ for threshold-based label and $rank\left(r_{i,t}\right) > 0.3S$ for quantile-based label) and the second column stands for positive result at time $t$. For simplicity, we only consider the probabilities for positive class (i.e., the second column of $\hat{\mathbf{Y}}_{\mathbf{t}}$) and symbolize the estimated probability $\hat{y}_{i,t}$ for stock $i$ using the OHLC image during the observation window spanning from $t-20$ to $t-1$ as:

$$\hat{y}_{i,t} = \begin{cases} \hat{y}_{i,1,t}^{(thr)}, & \text{for model built with threshold} - \text{based label;} \\ \hat{y}_{i,1,t}^{(q)}, & \text{for model built with quantile} - \text{based label,} \end{cases} \tag{17}$$

## 4 Data Description

The data used in this report and in the associated presentation is retrieved from Yahoo! Finance. Its link is <https://hk.finance.yahoo.com/>. The underlying stocks used in the model construction are the constituent stocks of Hang Seng Index (HSI), the primary equity market index in Hong Kong. See the list of stocks in Wikipedia with the link <https://en.wikipedia.org/wiki/Hang_Seng_Index>. It keeps track of the companies selected based on market capitalization, liquidity, and representation of the major sectors of the Hong Kong Stock Exchange.[11] Yahoo! Finance provides the daily data of the stock listed.

[11] This research focuses exclusively on the constituent stocks of a market index to ensure data consistency, representativeness, and reliability in both modeling and evaluation. Constituent stocks of a major index—such as those in HSI—are typically large-cap, highly liquid companies with continuous trading histories and well-maintained data records. This minimizes data gaps, price anomalies, and illiquidity effects that could bias the model's learning process.

We would like to adopt a data period from November 1, 2002 to October 31, 2025. It involves 5,654 trading days and 28 stocks; i.e., $S = 28$.[12]

## 4.1 Training and Validation Scheme

Following Jiang et al. (2023), we also adopt first 8-year trading data used in training and validation, while keeping remaining years as the test period. To be specific, both of our CNN and GCN models are trained and validated using data from November 1, 2002 to October 31, 2010, and are tested throughout the period from November 1, 2010 to October 31, 2025. Figure 12 illustrates the time series of the HSI over the entire investigation period. For the first 8-year data, we randomly split the samples such that 70% is used for model estimation and the remaining 30% is reserved for validation. As a result, we have 38,864 OHLC training samples and 16,660 OHLC validation samples.[13]

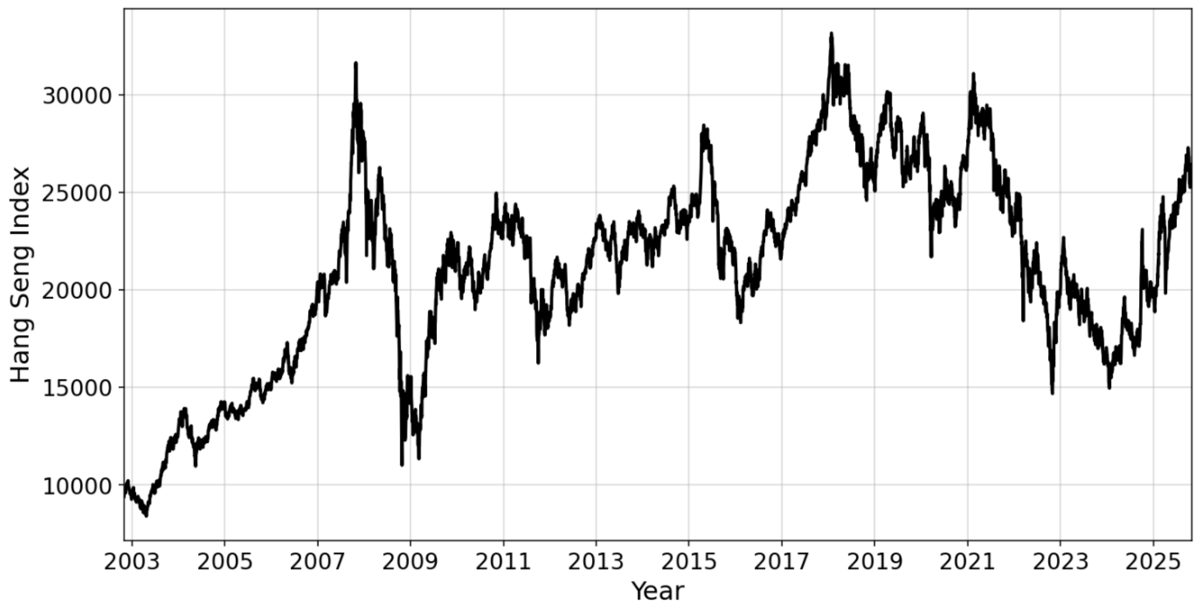

Figure 12. **Hang Seng Index Over the Full Data Period**.

During the 2002-2010 training window, the Hong Kong equity market experienced distinct phases of growth and decline, thereby exposing the model to a wide variety of market behaviors. During this window the HSI experienced significant fluctuations: after a recovery beginning in 2002, the market entered a strong bullish phase between 2003–2007, where the HSI rose by over 30 % per year, followed by the sharp 2008 global financial crisis which caused a decline of nearly 50 % in Hong Kong equities, and then a rebound phase in 2009–2010. This scheme improves model's ability to generalize when applied to "future" test period. In addition, stocks typically exhibit seasonal effects—recurring patterns

[12] In raw dataset, HSI comprises 82 constituent stocks, but not all of them were continuously traded throughout the full training window beginning on November 1, 2002. To ensure a balanced panel, we exclude securities with incomplete historical records or missing trading periods. After applying this filtering criterion, 28 stocks remained in the final research dataset.

[13] Training samples: 1,388 trading days × 28 stocks = 38,864 OHLC images; validation samples: 595 trading days × 28 stocks = 16,660 OHLC images.

tied to calendar months, quarter-ends, or corporate earnings seasons (Aggarwal and Rivoli, 1989). Using multiple complete yearly cycles helps neutralize seasonal biases in the training data. Randomly selecting training and validation samples within this eight-year window further helps balance the number of positive and negative labels in the classification task of threshold-based labeling, preventing over-representation of one label due to specific seasonal or cyclical market phases. In short, we adopt a shorter test period than the one in Jiang et al. (2023) when we implement the same 8-year setting in training and validation. This trade-off is deliberate: retaining a large, rich training set supports stronger learning of patterns and model robustness.

We use the average weekly return of the HSI as the labeling threshold to ensure that a stock is marked as a positive sample only when its realized next 5-day return exceeds the market's typical performance. Since the HSI reflects the overall Hong Kong market trend, its average weekly return during the training window (about 0.3092%) provides a practical and economically meaningful baseline for identifying stocks that truly outperform the market rather than those with negligible positive gains.

### 4.2 Test Scheme

Although there are 3,671 trading days available in test set, our investment period is set as 5 days and would not alter position during a single investment period. To fully utilize the full set of trading data, we divide the test set samples with 5 different starting dates and compute their cumulative investment returns correspondingly. Consequently, each test set has 20,552 OHLC images.[14]

## 5 Experimental Setup and Evaluation

We construct investment portfolios based on the predicted probabilities of gaining high holding period return generated by our trained CNN-MST-GCN framework. On each trading date within the test period, all available stocks are first ranked according to their predicted probabilities. A long–short quintile portfolio is then formed. In financial terms, a long position refers to purchasing a stock with the expectation that its price will rise, thereby generating profit when it appreciates. Conversely, a short position involves borrowing a stock and selling it immediately, with the expectation of repurchasing it later at a lower price to profit from its decline. Combining both sides yields a long–short portfolio, which simultaneously benefits from correctly predicting both rising and falling stocks. Specifically, the initial cash inflow from short positions funds the purchase of the stocks in long positions. Since the amount invested in long positions equals the amount borrowed for short positions, the portfolio requires no initial net capital outlay and is therefore referred to zero-cost. In other words, in contrast to the setting of $\sum_{i=1}^{S} w_{i,t} = 1$ introduced in equation (4), here we have a weight limit that $\sum_{i=1}^{S} w_{i,t} = 0$. In Jiang et al. (2023), a long–short decile portfolio is constructed by purchasing the top decile of stocks with the greatest predicted probabilities and short selling the bottom decile of stocks with the lowest predicted

[14] Each test set has samples: $\lfloor 3{,}671/5 \rfloor =$ 734 trading days $\times$ 28 stocks = 20,552 OHLC images. To make test set performance to be compared fairly, we drop the stock data of final trading day to make all portfolios made in test sets to be invested in equal length of time.

probabilities. However, due to the limited number of stocks available in our research, we adopt long–short quintile portfolio construction instead. Specifically, we buy 6 stocks with the greatest predicted probabilities and short sell 6 stocks with the lowest predicted probabilities at the start of the investment period.[15]

In this research, we no longer consider optimizing the portfolio weight $\boldsymbol{w_t}$ by Markowitz approach introduced in Section 3.1. The evidence shown in DeMiguel et al. (2009) suggests that the straightforward naïve–weight portfolio setting is not defeated by more complex stock allocation strategies. Therefore, we adopt an equally weighted strategy that evenly spend or gain initial capitals to all stocks in both long and short positions. The optimal weight vector $\boldsymbol{w_t^*} \in \mathbb{R}^S$ at time $t$ is defined as:

$$w_{i,t}^* = \begin{cases} \frac{1}{6}, & \text{if } rank(\hat{y}_{i,t}) \leq 6; \\ -\frac{1}{6}, & \text{if } rank(\hat{y}_{i,t}) \geq 23; \\ 0, & \text{otherwise}, \end{cases} \tag{18}$$

where $w_{i,t}^*$ is the $i$th stock's (optimized) weight at time $t$. Once the long–short quintile portfolio constructed at the market open on the first trading day, it is held until the market close on the fifth trading day and then we close all positions and rebalance the portfolio weights by next model prediction; i.e., $\boldsymbol{w_t^*} = \boldsymbol{0}$ at time $t+5$. The portfolio return is realized at the time of rebalance. In other words, the realized stock return $r_{i,t}$ defined in equation (3) replaces the expected term $\mu_{i,t}$ in equation (4) with substituting the optimal weight vector $\boldsymbol{w_t^*}$:

$$R_{p,t} = \boldsymbol{w^*}_{\boldsymbol{t}}^{\top} \boldsymbol{r_t}, \quad \sigma_{p,t}^2 = \boldsymbol{w^*}_{\boldsymbol{t}}^{\top} \boldsymbol{\Sigma}_{\mathrm{t}} \boldsymbol{w_t^*}, \tag{19}$$

where the realized return vector $\boldsymbol{r_t} = (r_{1,t}, r_{2,t}, \cdots, r_{S,t})^{\top}$ and $R_{p,t}$ stands for realized portfolio return at time $t$. Repeating this process across the test period produces a consecutive sequence of realized portfolio returns; i.e., $\{R_{p,t} \mid t = 5(\tau - 1) + k, \tau \in \{1, 2, \cdots, 734\}\}$, where $\tau$ denotes the $\tau$th trading window, and $k \in \{1, 2, \cdots, 5\}$ represents the $k$th test set.

Portfolio performance is evaluated through several standard financial metrics that collectively capture profitability, risk, and efficiency. The first metric is cumulative return over the whole test period that reflects the total appreciation of portfolio, which could vary considerably depending on the specific investment window examined; i.e., the choice of $k$. The cumulative return of the $k$th test set is computed as:

$$Q_k = \prod_{\tau=1}^{734} (1 + R_{p,\tau,k}) - 1. \tag{20}$$

The Sharpe ratio—defined as the average return divided by its standard deviation—serves as the key measure of risk-adjusted performance: a higher ratio indicates better returns relative to the level of risk taken (Sharpe, 1966). Our second metric is annualized realized Sharpe ratio, which is computed as:

[15] $\lceil 28 \times 20\% \rceil = 6$.

$$Sharpe_k = \frac{\frac{1}{734}\sum_{\tau=1}^{734}\left(R_{p,\tau,k} - r_f\right)}{\sqrt{\frac{1}{733}\sum_{\tau=1}^{734}\left(R_{p,\tau,k} - \frac{1}{734}\sum_{u=1}^{734} R_{p,u,k}\right)^2}} \times \sqrt{52} \tag{21}$$

where $r_f$ stands for weekly risk-free rate. Since it is a common practice for short-term long-short strategy evaluation that sets the risk-free rate as 0, we here discriminately set $r_f = 0\%$ for our model and $r_f = 0.02\%$ for benchmark.[16] The third metric is maximum drawdown that records the largest observed percentage decline from a previous peak in cumulative portfolio value, providing insight into downside risk and portfolio stability. Let $V_t$ denote the cumulative portfolio value at time $t$. Define the running maximum up to time $t$ as:

$$RM_t = \max_{1 \le u \le t} V_u. \tag{22}$$

The drawdown at time $t$ is then:

$$DD_t = \frac{RM_t - V_t}{RM_t}. \tag{23}$$

The maximum drawdown (MDD) over the evaluation horizon is:

$$MDD = \max_{1 \le t \le 734} DD_t. \tag{24}$$

Since HSI tracks the overall performance of major listed companies and thus represents a passive "buy-and-hold" investment in the market, we use it as benchmark of the investment performance. By comparing our model-driven long–short portfolio against the HSI's returns, we can determine whether the proposed CNN–MST–GCN framework offers predictive and economic value beyond standard market performance. Together, these evaluation steps connect model prediction accuracy with tangible investment outcomes, allowing an objective assessment of the system's practical relevance.

## 6 Results

### 6.1 Performance of Model Training

This section presents the training behavior of both the CNN and GCN components and summarizes how well each model learns the underlying patterns in the data and how effectively the training objectives are achieved.

#### 6.1.1 CNN

We evaluate the training behavior of the proposed CNN model under both the quantile-based and threshold-based labeling schemes. Although a grid search was planned for selecting the initial learning rate, the CNN exhibits relatively slow convergence and the available computation time is limited. As a result, we adopt $10^{-4}$ as the initial learning rate in the final training procedure and report the corresponding training performance here. With similar reasons, although the training configuration was

[16] It is noteworthy that the systematic risk is hedged in long-short portfolio but still exists in a market equity portfolio presented by HSI benchmark. As a result, it introduces different risk-free reference rate. It is a common practice to take 1% as the annual risk-free rate for convenience and to consider 52 weeks per year; i.e., 1% / 52 = 0.02%.

originally set to 30 epochs, the final number of epochs used differs across the two labeling schemes because of practical computational constraints. It was not feasible to run the full 30-epoch schedule for every experiment. During training, we observed that the validation loss reached a clear local minimum well before the 30th epoch—around 15 epochs for the quantile-based target and around 20 epochs for the threshold-based target. Once this local minimum was identified and the model parameters were saved, we proceeded to the next task rather than rerunning the full 30-epoch configuration. Although a longer training horizon might further reduce the risk of premature termination, the observed early convergence allowed us to obtain a stable model while managing the overall time cost of the research project. Figures 13–16 present the evolution of the training and validation losses and accuracies over the training epochs, together with the corresponding ROC curves computed on the validation set. Across both labeling schemes, the model exhibits stable convergence in the training loss and accuracy, and reaches its best validation performance at an intermediate epoch, consistent with the use of cosine learning-rate annealing and label smoothing.

As shown in Figure 13, under the quantile-based labeling scheme, the CNN achieves its best performance at epoch 15. At this point, the training loss and accuracy reach 0.4132 and 0.9083, respectively, while the validation loss and accuracy attain 0.5929 and 0.7138. The training loss decreases monotonically across epochs due to the stabilizing effect of AdamW optimization, pre-activation residual blocks, and the GELU nonlinearity. The validation loss displays noticeable volatility during the first ten epochs, reflecting the inherent difficulty of the quantile-based labeling rule, which induces a dynamic and cross-sectional target that depends on the relative ranking of stock returns each day. Beginning around epoch 10, the validation loss and accuracy stabilize, and by epoch 15 the model attains its optimal generalization performance. This behavior is consistent with the use of label smoothing, which softens hard labels and prevents overconfidence in early stages of training, thereby allowing the model to settle into a smoother convergence trajectory.

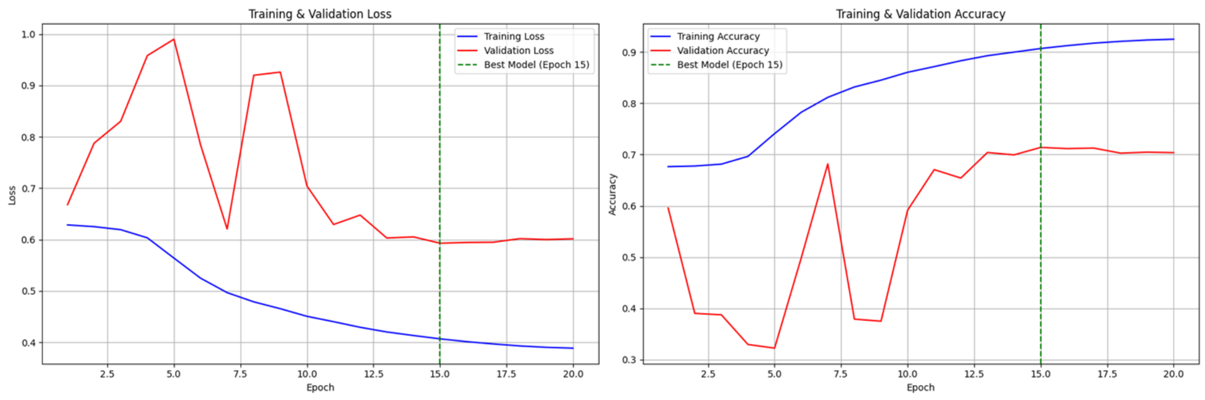

Figure 13. **Training and validation performance of the CNN using the quantile-based target**. The left panel shows the evolution of training and validation loss across epochs, while the right panel displays the corresponding accuracies. The vertical dashed line marks the epoch at which the best validation performance is achieved and the model parameters are saved.

The quantile-based model's ROC curve in Figure 14 yields an area under the curve (AUC) of 0.676, indicating predictive skill above random guessing and demonstrating that the model differentiates reasonably well between future top–30% performers and the remainder of the cross-section. The confusion matrix displayed in Table 3 shows that the classifier achieves a strong true-negative rate and a moderate true-positive rate, consistent with the imbalanced and dynamically defined positive class.

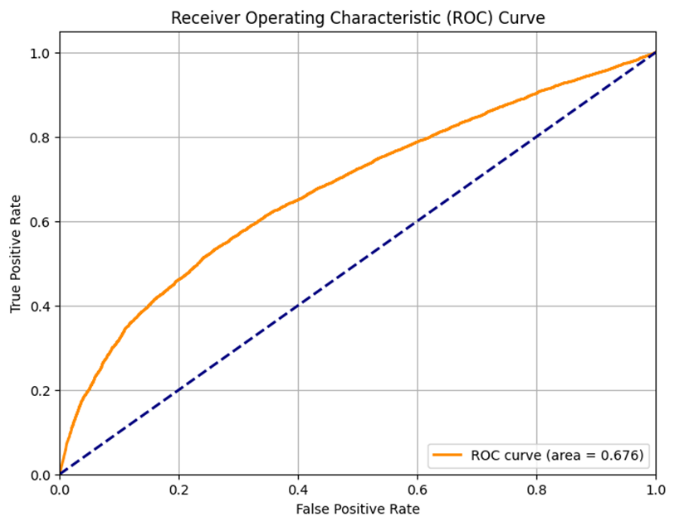

Figure 14. **ROC curve of the CNN model trained with the quantile-based target**.

| | | | |
|---|---|---|---|
| Actual Label | Top 30% Quantile | 1680 | 3675 |
| | Non-Top 30% Quantile | 1093 | 10212 |
| | | Top 30% Quantile | Non-Top 30% Quantile |
| | | Predicted Label | |

Table 3. **Confusion table of the CNN model trained with the quantile-based target**.

As displayed in Figure 15, under the threshold-based labeling scheme, the CNN reaches its best validation performance at epoch 20. At this epoch, the training loss and accuracy are 0.3687 and 0.9544, respectively, while the validation loss and accuracy are 0.6352 and 0.6673. The validation loss exhibits higher volatility than in the quantile-based case, reflecting the fact that threshold labels depend on absolute performance levels and are more sensitive to prevailing market conditions. Despite this noise, both the validation loss and accuracy display a gradual downward and upward trend, respectively, over the first 20 epochs, after which no substantial improvement occurs. As in the quantile-based case, the use of label smoothing and cosine annealing appears to contribute to the stabilization of the training dynamics, particularly in later epochs.

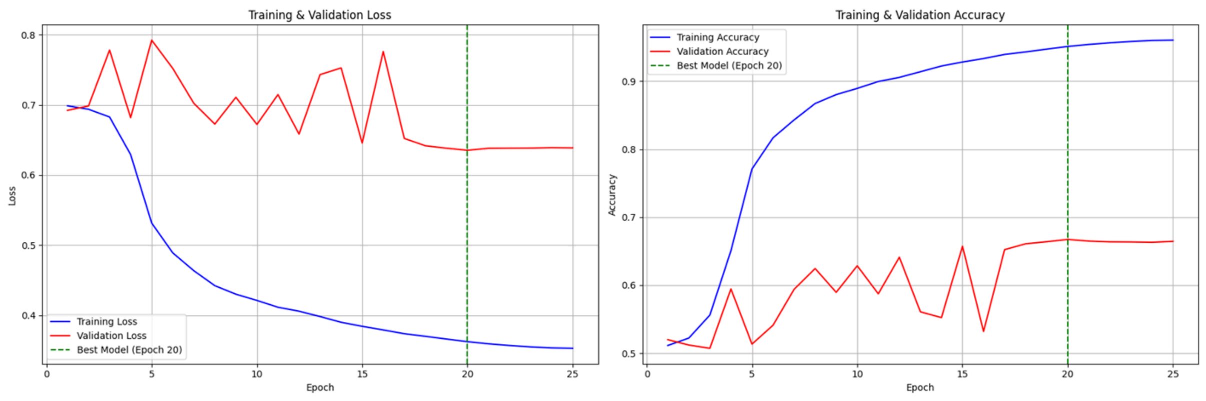

Figure 15. **Training and validation performance of the CNN using the threshold-based target**. The left panel shows the evolution of training and validation loss across epochs, while the right panel displays the corresponding accuracies. The vertical dashed line marks the epoch at which the best validation performance is achieved and the model parameters are saved.

The threshold-based model's ROC curve in Figure 16 yields an AUC of 0.720, demonstrating stronger discriminative capacity than the quantile-based model—an expected outcome given that the fixed return threshold produces a more separable binary target. The corresponding confusion matrix exhibited in Table 4 indicates balanced misclassification behavior, with the classifier displaying both a high true-negative rate and a substantial ability to identify positive-return episodes, despite the inherent noise and regime-dependence of short-horizon return outcomes.

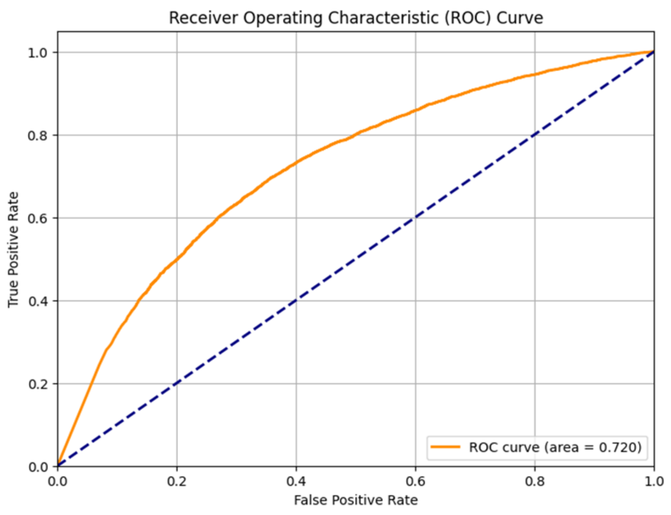

Figure 16. **ROC curve of the CNN model trained with the threshold-based target**.

As revealed in Table 5, compared with the quantile-based case, the threshold-based classifier achieves substantially higher recall and F1-score. It reflects that the absolute performance threshold produces a more separable target. The model therefore identifies positive-return episodes more effectively under this labeling rule, consistent with the improved ROC AUC of 0.720 relative to the quantile-based AUC of 0.676.

| | | | |
|---|---|---|---|
| Actual Label | Return $\geq$ 0.3% | 5499 | 2711 |
| | Return < 0.3% | 2832 | 5618 |
| | | Return $\geq$ 0.3% | Return < 0.3% |
| | | Predicted Label | |

Table 4. **Confusion table of the CNN model trained with the threshold-based target**.

| | **Threshold-based Labeling** | **Quantile-based Labeling** |
|---|---|---|
| **Training Loss** | 0.369 | 0.413 |
| **Validation Loss** | 0.635 | 0.593 |
| **Training Accuracy** | 0.954 | 0.908 |
| **Validation Accuracy** | 0.667 | 0.714 |
| **Precision** | 0.660 | 0.612 |
| **Recall** | 0.666 | 0.310 |
| **F1-Score** | 0.663 | 0.417 |
| **AUC** | 0.720 | 0.676 |

Table 5. **Performance summary of CNN models trained with quantile-based and threshold-based labels**. It reports the performance of the CNN model trained under the two labeling schemes. The precision, recall, F1-score, and AUC are calculated on the validation set, as these metrics assess the model's ability to generalize beyond the training data.

Taken together, the training results confirm that the proposed CNN model is able to learn meaningful visual representations from OHLC images under both labeling frameworks. The differences in training dynamics across the two schemes reflect the structural contrast between absolute and relative performance targets. Importantly, the convergence behavior aligns with the architectural and optimization choices adopted in this study—particularly the use of pre-activation residual blocks, GELU activations, AdamW with cosine learning-rate annealing, and label smoothing—all of which contribute to stable optimization and mitigate overfitting in this high-variance financial-image setting.

#### 6.1.2 GCN

We evaluate the training behavior of the proposed GCN model under both the quantile-based and threshold-based labeling schemes. Since each input image sample corresponds to a full financial network of 28 stocks on a single trading day, we set the batch size to 16, where each batch contains 16 distinct daily market graphs. Since we run 2-layer GCN only, we apply grid search on initial learning rate over five candidates $\{10^{-4}, 7.5 \times 10^{-5}, 5 \times 10^{-5}, 2.5 \times 10^{-5}, 10^{-5}\}$. Under the quantile-based labeling scheme, the optimal learning rate is $5 \times 10^{-5}$, whereas the optimal learning rate is $10^{-4}$ for the threshold-based labeling scheme.

Originally, we planned to train the GCN for 30 epochs with a cosine annealing schedule. However, cosine annealing led to premature flattening of validation loss and insufficient optimization. We therefore adopt a stepwise learning rate schedule with a patience of four epochs and a multiplicative decay factor of 0.8, which provides more gradual decay and improves stability under dynamic MST-based graph inputs. Given the faster convergence of a 2-layer GCN together with the slower learning-rate decay, we extend the training horizon to 100 epochs to allow the schedule to operate effectively.

As exhibited in Figure 17, under the threshold-based labeling scheme, the GCN reaches its best validation performance at epoch 5. At this epoch, the training loss and accuracy are 0.693 and 0.520, respectively, while the validation loss and accuracy are 0.699 and 0.494. The sharp early reduction in validation loss corresponds to fast assimilation of the relatively clean threshold structure, while the slow post-epoch-5 degradation reflects mild overfitting typical for short-horizon return classification. The shallow slope of the validation loss increase indicates that the model maintains stable generalization despite fitting training graphs more closely over time. On the other hand, validation accuracy remains tightly clustered near 0.49–0.50 throughout training, with only short-lived fluctuations during the earliest epochs. This stability is expected given that the threshold task is balanced and that random guessing produces approximately 50% accuracy. Because the model assigns continuous probabilities rather than strict binary predictions, accuracy alone does not reveal the quality of its ranking or class separation. The flat validation accuracy curve confirms that binary accuracy is not a sufficiently sensitive evaluation metric for this task.

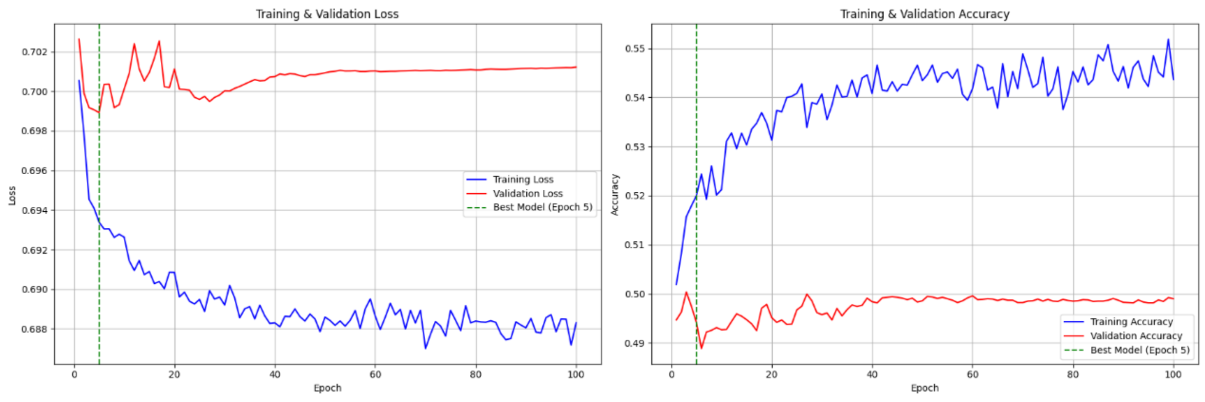

Figure 17. **Training and validation performance of the GCN using the threshold-based target**. The left panel shows the evolution of training and validation loss across epochs, while the right panel displays the corresponding accuracies. The vertical dashed line marks the epoch at which the best validation performance is achieved and the model parameters are saved.

Figure 18 reveals that the ROC curve lies almost exactly on the random 45° diagonal, yielding an AUC of 0.494. This indicates that the model's predicted probabilities provide virtually no rank-order separation between above-threshold and below-threshold returns. The near-random ROC performance likely results from compressed probability outputs induced by GraphNorm and MST sparsity, as well as the inherent difficulty of predicting whether a stock's 5-day return surpasses a fixed absolute threshold in a noisy market environment. Although the model can form a balanced classification boundary, its scores lack the variance required for effective discrimination across the full probability spectrum.

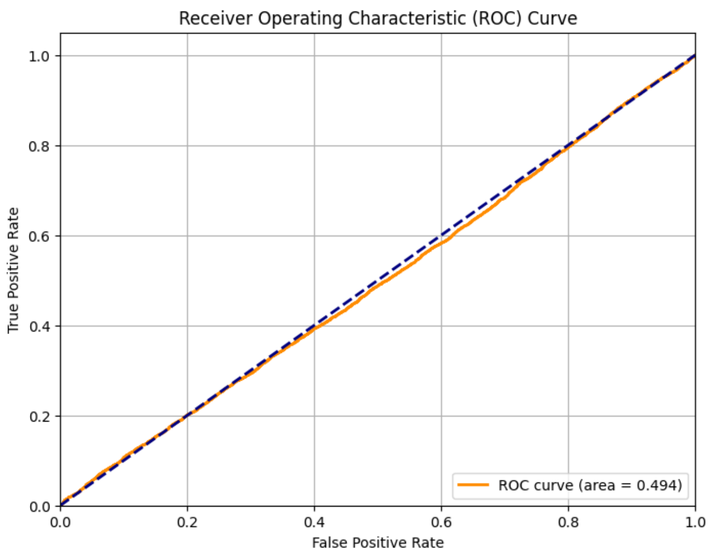

Figure 18. **ROC curve of the GCN model trained with the threshold-based target**.

The confusion matrix shown in Table 6 reflects a balanced classification profile: true positives (3,968) and true negatives (4,265) appear in similar magnitudes, as do false positives (4,185) and false negatives (4,242). This symmetry confirms that the GCN does not exhibit class bias and is capable of identifying above-threshold events at nearly the same rate as below-threshold events. While the ROC curve highlights minimal ranking ability, the confusion matrix shows that the model maintains a coherent and stable decision boundary under the default prediction threshold.

| | | | |
|---|---|---|---|
| Actual Label | Return ≥ 0.3% | 3968 | 4242 |
| | Return < 0.3% | 4185 | 4265 |
| | | Return ≥ 0.3% | Return < 0.3% |
| | | Predicted Label | |

Table 6. **Confusion table of the GCN model trained with the threshold-based target**.

As illustrated in Figure 19, under the quantile-based labeling scheme, the GCN reaches its best validation performance at epoch 46. At this epoch, the training loss and accuracy are 0.700 and 0.461, respectively, while the validation loss and accuracy are 0.699 and 0.494. This scheme produces more complex loss behavior due to the cross-sectional nature of the target. Validation loss initially increases slightly to around 0.717 before declining to approximately 0.709 later in training. This early increase reflects the sensitivity of quantile labels to daily return distributions and MST topology changes. The relatively small train–validation loss gap throughout training suggests minimal overfitting despite the extended 100-epoch horizon. Validation accuracy increases sharply during the initial epochs—from approximately 0.52 to 0.63 within the first 10 epochs—and then oscillates around the 0.60 level. The larger validation accuracy relative to training accuracy indicates that the GCN generalizes relational information better across unseen graph samples than it fits the idiosyncrasies of the training set. The oscillatory plateau reflects the volatile nature of quantile labels, which respond strongly to day-to-day changes in the relative ordering of returns.

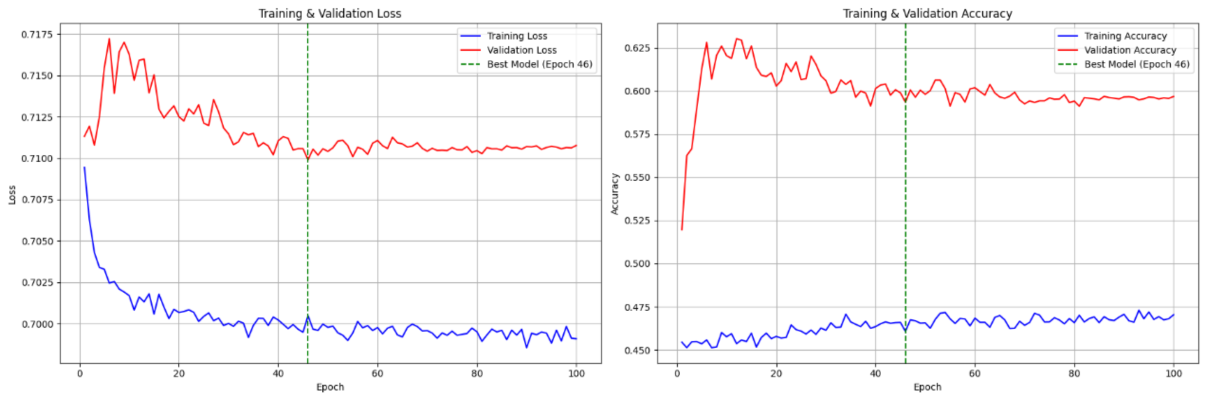

Figure 19. **Training and validation performance of the GCN using the quantile-based target**. The left panel shows the evolution of training and validation loss across epochs, while the right panel displays the corresponding accuracies. The vertical dashed line marks the epoch at which the best validation performance is achieved and the model parameters are saved.

Figure 20 demonstrates that the quantile-based GCN achieves a ROC AUC of 0.518, slightly above random but distinctly higher than the threshold-based counterpart. The ROC curve's shallow curvature indicates limited discriminative ability, yet the AUC above 0.50 confirms the presence of weak but meaningful ranking information. This modest performance is consistent with the objective: quantile labeling rewards correct relative ordering rather than absolute prediction, and short-horizon stock rankings are known to be noisy and weakly predictable. The slightly better AUC relative to the threshold scheme suggests that the GCN captures small but useful differences in the CNN embeddings when interpreted through the graph structure.

The quantile confusion matrix in Table 7 displays a pronounced class imbalance effect. True negatives (8,342) significantly exceed true positives (1,545), and false negatives (3,810) also outnumber false positives (2,963). This asymmetry is expected because the positive class (top 30%) is considerably

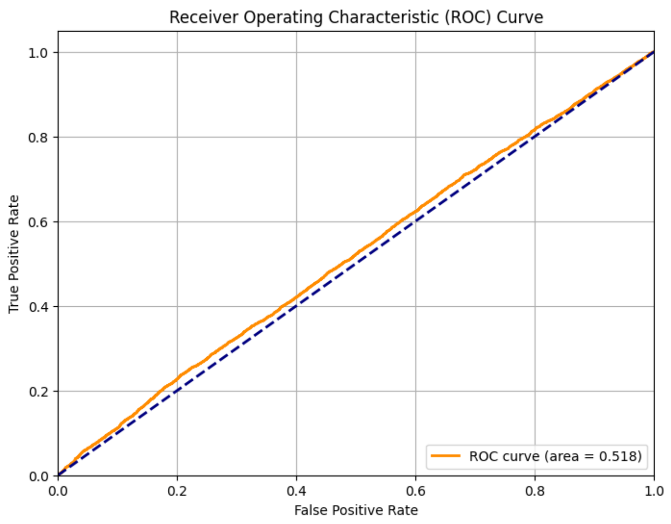

Figure 20. **ROC curve of the GCN model trained with the quantile-based target**.

smaller by construction. The model tends to make conservative predictions, favoring the majority class and thus achieving high true-negative rates but substantially lower recall for the top-quantile category. This behavior is common in cross-sectional ranking tasks, where the positive class is scarce and noisy.

| | | | |
|---|---|---|---|
| Actual Label | Top 30% Quantile | 1545 | 3810 |
| | Non-Top 30% Quantile | 2963 | 8342 |
| | | Top 30% Quantile | Non-Top 30% Quantile |
| | | Predicted Label | |

Table 7. **Confusion table of the GCN model trained with the quantile-based target**.

Table 8 highlights the complementary strengths of the two labeling schemes. The threshold-based GCN achieves substantially higher precision, recall, and F1 score (0.487, 0.483, 0.485), reflecting its ability to learn a clearer binary decision structure. In contrast, the quantile-based GCN produces higher validation accuracy (0.593) and a better AUC (0.518), indicating stronger relative ranking capability despite poorer class-balanced metrics. Training and validation losses follow this pattern: the threshold model attains lower losses due to cleaner labels, while the quantile model exhibits more volatile validation dynamics. Overall, the results confirm that each labeling scheme emphasizes a different predictive dimension—classification coherence for threshold labels and weak ranking discrimination for quantile labels—both of which contribute to a more complete understanding of cross-sectional return predictability when combined with CNN-extracted features.

| | **Threshold-based Labeling** | **Quantile-based Labeling** |
|---|---|---|
| **Training Loss** | 0.693 | 0.700 |
| **Validation Loss** | 0.699 | 0.710 |
| **Training Accuracy** | 0.520 | 0.461 |
| **Validation Accuracy** | 0.494 | 0.593 |
| **Precision** | 0.487 | 0.343 |
| **Recall** | 0.483 | 0.289 |
| **F1-Score** | 0.485 | 0.313 |
| **AUC** | 0.494 | 0.518 |

Table 8. **Performance summary of GCN models trained with quantile-based and threshold-based labels**. It reports the performance of the GCN model trained under the two labeling schemes. The precision, recall, F1-score, and AUC are calculated on the validation set, as these metrics assess the model's ability to generalize beyond the training data.

### 6.2 Investment Performance

This section evaluates the out-of-sample investment performance of the proposed CNN-MST-GCN-based long-short strategy. To ensure the robustness of the results and mitigate potential timing biases related to the 5-day holding period, we examine five distinct test sets, each with a staggered starting trading date. We compare the performance of these individual sets against the market benchmark, and we also present the investment performance of the literature referred.

We first introduce the investment performance of our mere CNN models. Figure 21 reveals the cumulative returns of the long–short strategy over the out-of-sample testing window using the threshold-based labeling scheme. Across all five test sets (shown as light blue lines), the strategy consistently delivers positive cumulative returns, exhibiting strong resilience through multiple market cycles. Although the average performance over the five test sets (black line) generally outperforms the HSI benchmark (red line), a noticeable exception occurs during the 2018–2019 period, where the average strategy briefly falls below the benchmark before recovering. This temporary underperformance indicates that the threshold-based classifier—relying on an absolute return cutoff—can lose discriminative power during market environments where absolute returns are compressed or when directional signals weaken across the cross-section. Nevertheless, the strategy resumes a clear upward trajectory thereafter, demonstrating its ability to identify winning and losing stocks sufficiently well to produce long-term excess returns. Specifically, HSI benchmark displays the investment performance of the weighted average of all of its constituent stocks, and our strategy does successfully identify both outperforming and underperforming stocks.

In contrast, the quantile-based labeling scheme generates a markedly more stable and persistent

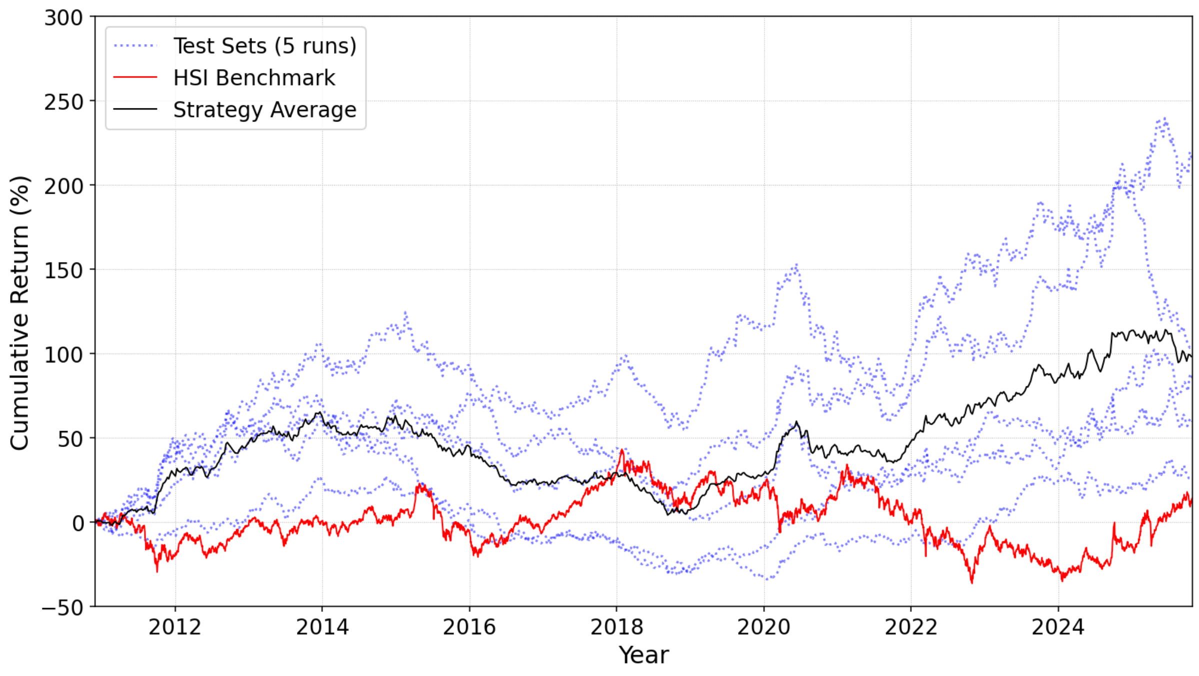

Figure 21. **Investment performance based on mere-CNN model with threshold-based labeling scheme**.

performance profile. As presented in Figure 22, the cumulative return curve of average performance of this GCN model maintains a continuous divergence above the HSI benchmark throughout the entire test period, including the 2018–2019 regime where the threshold-based model temporarily falters. In addition, this strategy remains resilient during major market drawdowns, including the periods of 2015–2016 and 2019–2020, when the benchmark experiences extended weakness. This robustness stems from the relative nature of quantile labels: by ranking stocks cross-sectionally, the model continues to extract meaningful predictive structure even when overall market movements are subdued or broadly negative. As a result, both the variance across the five test sets and the volatility of the average return curve are noticeably reduced compared with the threshold-based case. The quantile-based strategy ultimately achieves a higher cumulative return and smoother compounding, reflecting stronger cross-sectional discriminative power.

We next evaluate the investment performance of the proposed CNN–MST–GCN framework, which integrates visual feature extraction, market-topology construction, and graph-based relational learning. As with the mere-CNN model, long–short portfolios are formed every five trading days and tested across five staggered out-of-sample windows. However, the inclusion of the MST-derived financial network and the GCN component fundamentally changes how predictive information is learned and aggregated across stocks. The resulting performance differs markedly depending on whether threshold-based or quantile-based labels are used to train the models.

Under the threshold-based labeling scheme, the CNN–MST–GCN model exhibits limited effectiveness.

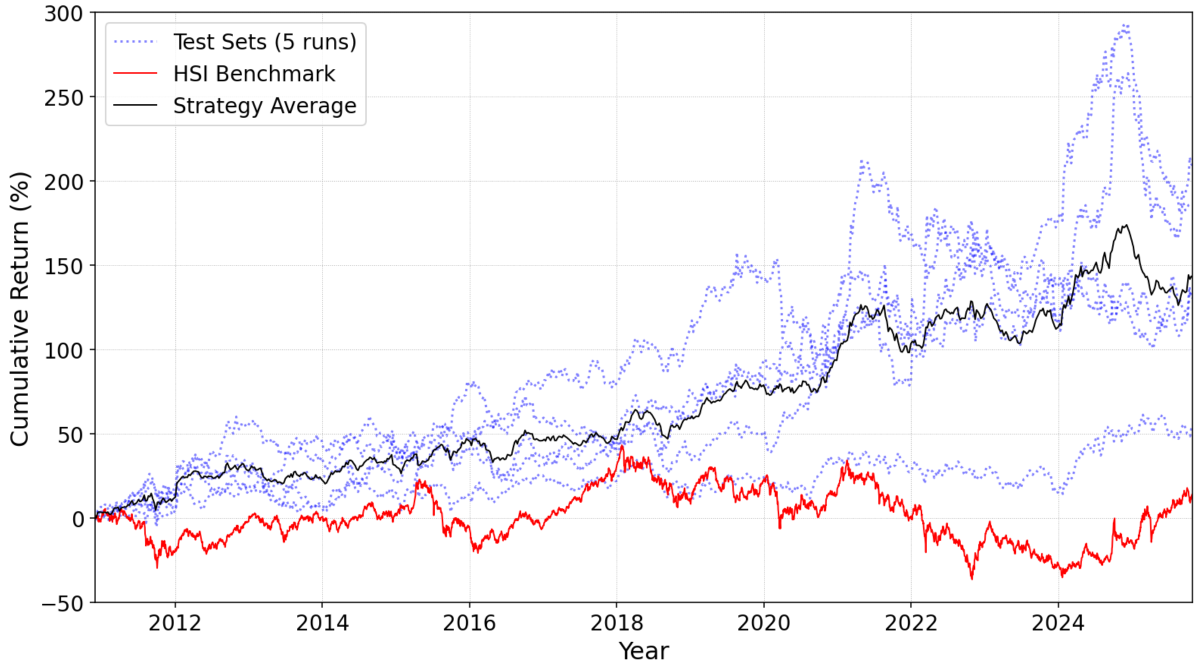

Figure 22. **Investment performance based on mere-CNN model with quantile-based labeling scheme**.

As shown in Figure 23, the average cumulative return remains near zero for most of the sample period and only begins to rise modestly after 2023. Throughout much of the evaluation horizon, the strategy average even lags behind the HSI benchmark, particularly during 2017–2019 and 2021. Individual test sets vary widely, with some producing moderate gains while others remain negative, revealing substantial sensitivity to test-window alignment. This unstable behavior indicates that the threshold-based supervisory signal does not align well with graph-based learning: when labels are defined by an absolute return cutoff, cross-sectional structure becomes noisy, leading the MST–GCN pipeline to propagate weak or inconsistent signals across the network. Relative to the mere-CNN threshold model, which produced strong and persistent outperformance, incorporating MST and GCN under threshold-based labels actually degrades performance, suggesting poor compatibility between absolute-return labeling and relational learning.

In contrast, the quantile-based labeling scheme yields a dramatically different outcome. As shown in Figure 24, the CNN–MST–GCN model achieves strong and persistent cumulative returns, with the strategy average maintaining a clear divergence above the HSI benchmark throughout the entire test period. The average return curve rises steadily—particularly after 2021—and concludes with cumulative gains exceeding 120%, substantially outperforming both the threshold-based GCN model and the quantile-trained mere-CNN model. Although individual test runs exhibit wide dispersion, several achieve exceptionally high returns, exceeding 500%, demonstrating the framework's ability to exploit favorable relational structures when they arise. Importantly, unlike the threshold-based GCN, the quantile-based version remains robust during challenging market conditions such as the 2015–2016

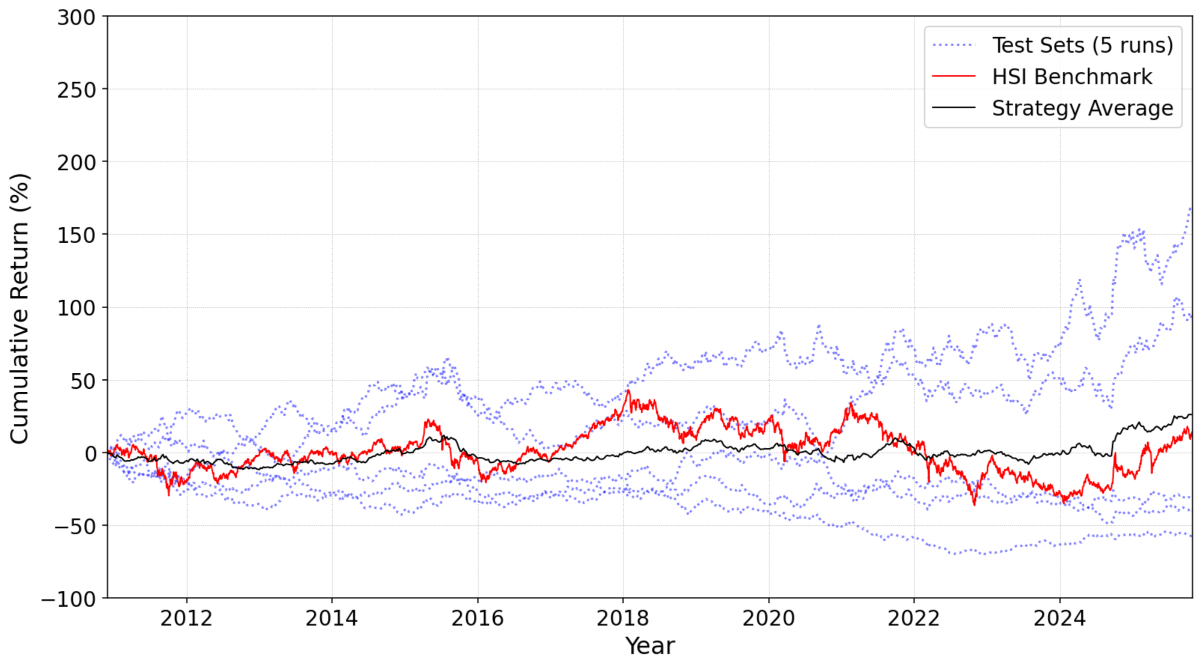

Figure 23. **Investment performance based on CNN-MST-GCN model with threshold-based labeling scheme**.

correction, the 2018–2019 slowdown, and the 2021–2022 volatility. This stability reflects that quantile labels encode cross-sectional ranking, a signal naturally compatible with MST-based topology and GCN aggregation. By learning how stocks relate to each other within the market network, the model enhances the discriminative power of CNN-derived embeddings and amplifies profitable long–short opportunities.

Table 9 presents investment performance with evaluation metrics among our models, benchmark, and the literature referred in this research. For the CNN–MST–GCN model under the threshold-based labeling scheme, the results confirm the weaknesses already visible in the cumulative return plot. Although individual test sets show large variation—ranging from a 164.8% gain to a –57.7% loss—the strategy average yields only a modest cumulative return of 25.9%, with an annualized realized Sharpe ratio of 0.06.[17] The extremely wide dispersion across test sets indicates that the relational learning mechanism struggles to extract stable cross-stock patterns under absolute-return thresholds. Furthermore, the maximum drawdowns exceed –40% in most cases, reaching as low as –70.1%, which highlights the fragility of performance when GCN propagation is guided by noisy or inconsistent labels.

[17] We report all the evaluation metrics for taking average of the values of the metrics of the five test sets for strategy average instead of applying the metrics on the average series, since the latter one would smooth out the individual volatility on its results. In other words, taking average of evaluation reflects the mean performance of investment strategy under implementation of the model, whereas evaluation of taking average represents the investment performance of taking strategy diversification under implementation of the model. In our research, we focus on examination of the investment value with computer vision and market topology insights and compare with the investment performance reported in literature referred. Thus, it shows the potential in increasing risk-adjusted returns by incorporating strategy diversification under our CNN-MST-GCN framework.

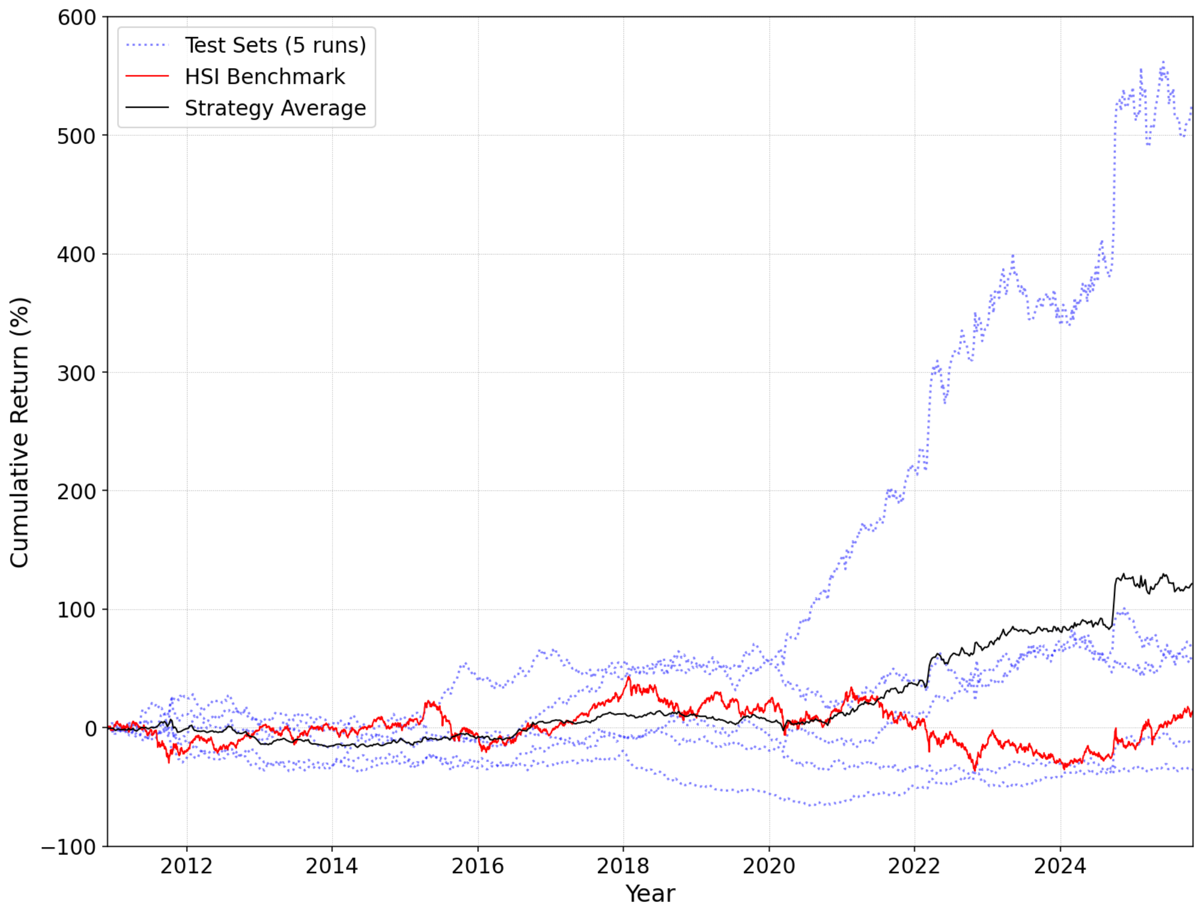

Figure 24. **Investment performance based on CNN-MST-GCN model with quantile-based labeling scheme**.

These results align closely with the flat and often underperforming cumulative return curve observed for this configuration.

In contrast, the CNN–MST–GCN model trained with quantile-based labels delivers markedly stronger and more consistent outcomes. As seen in both the cumulative return plot and the metrics table, several test sets show exceptionally high cumulative returns—including one exceeding 526%—while others still produce substantial gains. The strategy average achieves a cumulative return of 121.4% with an annualized Sharpe ratio of 0.28, indicating a more favorable risk–return profile. Notably, the maximum drawdowns are significantly smaller than those of the threshold-based GCN model, with many test sets remaining above –40%. These quantitative results reinforce the earlier observation that quantile labeling provides supervisory signals aligned with cross-sectional structure, allowing the MST–GCN component to propagate meaningful relational information and amplify profitable long–short opportunities across the market network.

The performance of the mere-CNN model serves as an important benchmark for evaluating the added value of relational learning. Under the threshold-based labeling scheme, the mere-CNN model achieves

| Model Type | Labeling Scheme | Test Set | Cumulative Return | Annualized Return | Annualized Realized Sharpe Ratio | Maximum Drawdown |
|---|---|---|---|---|---|---|
| CNN-MST-GCN | Threshold-based | 1 | -31.3% | -2.5% | -0.10 | -41.1% |
| | | 2 | 164.8% | 6.7% | 0.52 | -42.0% |
| | | 3 | 94.4% | 4.5% | 0.39 | -33.4% |
| | | 4 | -57.7% | -5.6% | -0.34 | -70.1% |
| | | 5 | -40.5% | 3.4% | -0.17 | -49.4% |
| | | Strategy Average | 25.9% (98.3%) | 1.5% | 0.06 (0.37) | -47.2% (14.0%) |
| | Quantile-based | 1 | 526.5% | 13.0% | 0.87 | -37.0% |
| | | 2 | 57.2% | 3.1% | 0.28 | -25.8% |
| | | 3 | -9.9% | -0.7% | 0.03 | -39.1% |
| | | 4 | -35.3% | -2.9% | -0.13 | -67.4% |
| | | 5 | 68.6% | 3.5% | 0.32 | -32.7% |
| | | Strategy Average | 121.4% (230.7%) | 5.4% | 0.28 (0.38) | -40.4% (15.9%) |
| Mere-CNN | Threshold-based | 1 | 187.0% | 7.3% | 0.56 | -30.9% |
| | | 2 | 54.3% | 2.9% | 0.29 | -24.4% |
| | | 3 | 135.6% | 5.9% | 0.49 | -23.1% |
| | | 4 | 130.6% | 5.7% | 0.46 | -33.6% |
| | | 5 | 209.2% | 7.8% | 0.62 | -28.6% |
| | | Strategy Average | 143.3% (60.0%) | 6.1% | 0.48 (0.13) | -28.1% (4.4%) |
| | Quantile-based | 1 | 103.9% | 4.9% | 0.43 | -32.9% |
| | | 2 | 26.6% | 1.6% | 0.19 | -58.8% |
| | | 3 | 59.6% | 3.2% | 0.31 | -45.2% |
| | | 4 | 85.3% | 4.2% | 0.36 | -43.2% |
| | | 5 | 215.4% | 8.0% | 0.65 | -30.8% |
| | | Strategy Average | 98.2% (71.7%) | 4.7% | 0.39 (0.17) | -42.2% (11.2%) |
| HSI | - | - | 13.7% | 0.9% | 0.10 | -55.7% |
| Jiang et al. (2023) | Threshold-based | - | - | 53.7% | 7.15 | - |
| Zhu and Zhu (2024) | Threshold-based | - | 581% | 13% | 4.47 | - |

Table 9. **Portfolio Performance**. For the values presented in Strategy Average, it takes the average value across the five test sets in the same model. The parenthesis indicates the standard deviation of the results of the five test sets in the same model.

strong results, with all test sets yielding positive cumulative returns and the strategy average reaching 143.3%, supported by a Sharpe ratio of 0.48. The maximum drawdowns remain moderate compared with the CNN–MST–GCN threshold model, reflecting the more stable and consistent upward trajectory observed in the cumulative return curve. This contrast is especially notable: while the mere-CNN threshold model demonstrates strong directional predictive power, introducing MST and GCN under the same labeling scheme diminishes performance rather than improving it. This outcome confirms that threshold-based labels lack the relational structure required for effective graph learning.

For the mere-CNN model with quantile-based labeling, the performance remains robust, with a strategy-average cumulative return of 98.2% and a Sharpe ratio of 0.39. Drawdowns are contained, and the cumulative return curve shows a stable upward progression. However, when comparing the mere-CNN and CNN–MST–GCN models under quantile labeling, it becomes clear that the graph-enhanced version offers meaningful improvements in both total return and risk-adjusted performance. The quantile-trained CNN–MST–GCN achieves higher cumulative returns and similar or better Sharpe ratios, while demonstrating greater resilience during adverse market regimes. This confirms that relational information—when paired with cross-sectionally meaningful labels—provides incremental predictive value beyond what can be extracted from price images alone.

Finally, the HSI benchmark sits far below all model configurations in terms of cumulative return (13.7%) and Sharpe ratio (0.10), with a maximum drawdown exceeding –55%. This highlights the advantage of market-neutral long–short strategies relative to simple index exposure and provides a useful baseline against which to measure the effectiveness of deep learning–based forecasting approaches.

In short, the portfolio metrics and cumulative return curves consistently reveal that quantile-based labeling is essential for unlocking the benefits of graph learning, whereas threshold-based labeling favors purely image-driven CNN models. The CNN–MST–GCN framework demonstrates its strongest performance when the supervision signal aligns with the relational structure of the MST network, thereby validating the central motivation of this research: combining visual pattern extraction with network-based relational learning can significantly enhance investment performance when the targets correctly encode cross-sectional market structure.

### 6.2.1 Comparison With Prior Literature

To contextualize the performance of our CNN–MST–GCN framework, we compare our investment outcomes with the results reported in the literature we referred to, Jiang et al. (2023) and Zhu and Zhu (2024), whose annualized returns are included in Table 9. Both studies document substantially higher investment performance than ours; however, these differences arise primarily from the market environments, cross-sectional data availability, and structural constraints of the respective studies rather than from methodological shortcomings of our framework.

A fundamental distinction lies in the size and composition of the investment universe. Our empirical analysis is restricted to the constituent stocks of HSI. After a filtering for full-period data availability, only 28 stocks remain in the asset pool. This narrow universe severely limits the degree of cross-sectional return dispersion, constrains long–short separation, and restricts diversification. As a result, even a model with strong predictive insights is inherently limited by the small number of securities available for portfolio construction. In contrast, Jiang et al. (2023) train their CNN-based trader on every listed firm across the NYSE, AMEX, and NASDAQ—typically over 3,000 stocks. Similarly, Zhu and Zhu (2024) analyze the Chinese A-share market, which contains more than 1,800 stocks with high turnover and strong behavioral momentum. These vastly larger datasets offer richer structural patterns and allow long–short strategies to capitalize on substantial cross-sectional variation—an advantage unavailable in our Hong Kong–focused setting.

Market conditions further amplify these differences. The Hong Kong market experienced prolonged weakness throughout our test window, with the HSI generating an annualized return of just 0.9% and a maximum drawdown exceeding –55%. Such conditions suppress trend persistence and reduce opportunities for systematic strategies to accumulate directional profits. By contrast, the U.S. market during Jiang et al. (2023)'s study period exhibited strong structural growth and deep liquidity, supporting their reported annualized return of 53.7%. Likewise, Zhu and Zhu (2024)'s dataset spans periods of pronounced bull markets in the A-share ecosystem, enabling time-series momentum and chart-based signals to perform exceptionally well, with reported cumulative returns exceeding 581% and an annualized return of 13%. These outcomes reflect not only methodological strength but also the inherent return-generating capacity of the respective equity markets.

When these structural factors are considered collectively—market weakness, limited universe size, and reduced cross-sectional dispersion—the performance gap between our strategies and those reported in the literature is fully explained. The lower returns in our study do not imply that the CNN–MST–GCN framework is ineffective. In fact, when compared against the HSI benchmark, our models still produce substantial excess returns and improved risk-adjusted performance. This demonstrates that the proposed framework extracts meaningful predictive information even in a market environment that is far less favorable than those examined in prior work.

## 7 Discussion

In this section, we discuss the practical and methodological limitations that constrained the predictive and economic performance of the proposed models. Building on these observations, we outline several directions for future work aimed at enhancing data representation, graph construction, neural network architectures, and portfolio construction. Finally, we summarize the main insights drawn from our analysis and experiments.

### 7.1 Analysis of Unfavorable Results and Limitations

Although the proposed CNN–MST–GCN framework demonstrates meaningful predictive and economic value, several limitations arise from the characteristics of the dataset, the modeling choices, and the scope of the experimental design. These limitations are important for interpreting the results, especially the root causes of our unfavorable results demonstrated, and understanding how they may generalize to other settings.

A major constraint of this study comes from the extremely small asset universe considered. Our dataset includes only 28 constituent stocks of the Hang Seng Index that have complete historical records throughout the sample period. This corresponds to training and evaluating models on a dataset with very limited cross-sectional diversity. A small number of stocks reduces the amount of variation available for the model to learn meaningful discriminative features. Furthermore, because these stocks are all large-cap, mature firms, their price movements tend to be slower, less volatile, and more highly correlated, offering weaker visual patterns compared with smaller or more speculative equities (Banz, 1981; Fama and French, 1992). This reduces the expressive advantage of CNN-based price-image modeling. In addition, requiring full-period availability introduces survivorship bias: stocks that delisted or joined the index later are excluded, potentially biasing the sample toward firms with stronger long-term stability (Malkiel, 1995; Shumway, 1997).

From a modeling standpoint, the use of a minimum spanning tree (MST) to represent the financial network introduces a substantial simplification of the underlying inter-stock relationships. The MST reduces a 28×28 similarity matrix—which contains 756 possible undirected connections—to only 27 edges. This means that over 96% of potential relationships are discarded. While MST filtering is helpful for removing noise and simplifying the graph for GCN training, it also eliminates many secondary but potentially informative relationships between stocks (Tewarie et al. 2015). Although this reduces overfitting risk, it may also remove structural information that could help the model generalize better. Additionally, the performance of the proposed models exhibits clear sensitivity to the choice of labeling scheme. Threshold-based labels encode absolute price movements and provide weaker cross-sectional structure, whereas quantile-based labels encode relative rankings and align more naturally with graph propagation. The reliance on labeling structure highlights how sensitive deep learning systems can be to supervisory signal design.

Several limitations also stem from the experimental and evaluation setup. The backtesting results do not incorporate transaction costs, such as bid–ask spreads, execution slippage, or short-selling restrictions. These costs can materially affect real-world trading performance and may reduce the profitability of high-turnover strategies. The portfolio construction method relies on equal-weight allocation, meaning each selected stock contributes the same position size regardless of volatility or uncertainty. More advanced allocation schemes—such as volatility scaling, risk budgeting, or optimization-based weighting—may better leverage predictive signals (Markowitz, 1952; Chow and Kritzman, 2001; Dalio 2005; Almahdi and Yang, 2017). Model selection, including hyperparameter tuning and architecture exploration, was also limited by computational constraints, preventing exhaustive search over deeper CNN/GCN architectures, alternative graph constructions, or larger embedding dimensions.

Finally, the generalization of our findings is limited in several ways. The framework is trained and tested exclusively on the Hong Kong equity market, which differs significantly from U.S. and Chinese mainland markets in terms of liquidity, volatility, and investor behavior. As a result, performance observed in this environment may not extend to markets with different structural characteristics. The study also relies solely on daily OHLC images, excluding other potentially informative modalities such as intraday data, fundamental indicators, macroeconomic signals, or textual sentiment. Likewise, only one type of price-image encoding (candlestick-style charts with trading volume and 20-day moving average line) is used. Alternative or augmented visual representations—such as volume profile overlays, technical indicators (e.g., moving average line with other windows, relative strength Index (RSI), moving average convergence divergence (MACD), etc.), or multi-channel chart embeddings—may yield richer signals and improve robustness. Without such robustness checks, the results should be interpreted with an understanding that the current model reflects only one design choice among many possible alternatives.

## 7.2 Future Works

To address these unfavorable outcomes in model training and investment performance, as well as the weaknesses and constraints identified in the preceding sections, the following subsections outline a set of potential directions for improvement. These future works highlight possible solutions and extensions that may enhance effectiveness, robustness, and generalizability of the CNN–MST–GCN framework.

### 7.2.1 OHLC Image Generation

The predictive capacity of both CNN and GCN components in this study is fundamentally constrained by the small and homogeneous universe of 28 HSI constituent stocks, which limits cross-sectional variation and weakens both image-based signals and network-based relational structure. Future work should therefore broaden the data domain by expanding the stock universe to include a larger share of the Hong Kong market. A richer cross-section would increase dispersion in short-horizon returns, enhance the learning of relational patterns in the MST, and provide a more challenging but informative environment for graph learning. Furthermore, the OHLC image representations can be substantially enriched. The current single-channel candlestick design encodes only geometric price–volume patterns, yet the empirical results—particularly the weak discriminative strength of GCN under threshold labels—suggest that additional visual signals may be needed to strengthen the latent relationships that the graph model propagates. Extending images to include multi-channel technical indicators, alternative OHLC encodings, heatmap-style volume–price interactions, or dynamic observation windows could supply more informative features. For example, Jiang et al. (2023) also set the observation windows and investment windows as 5-day, 20-day, and 60-day, whereas Zhu and Zhu (2024) even consider 1-

day setting in the investment windows. Incorporating self-supervised or contrastive learning techniques to pretrain CNN embeddings may further enhance their robustness, enabling the GCN to operate on more expressive and economically meaningful visual abstractions.

### 7.2.2 Graph Construction

The MST graph suffers from severe information loss, preserving only 4% of potential pairwise relations. The investment results show that this filtration interacts very differently with the two labeling schemes—amplifying performance under quantile labels but degrading it under threshold labels—indicating that the current topology may not capture the full spectrum of market interactions required for more challenging supervisory signals. Future research should explore denser or more flexible graph constructions, such as k-nearest-neighbor graph, planar maximally filtered graph, or sector-aware hybrid networks that incorporate both learned similarity and economic structure (Tumminello et al., 2005; Chen et al., 2020; Shi et al., 2024). Dynamic graph construction, where edge weights evolve over time, such as adaptive MST or dynamic financial networks, may better capture market regime shifts that affect the propagation of relational information (Bardoscia et al., 2021; Wang and Li, 2025). An additional direction is to replace the adjacency-matrix formulation with a graph Laplacian–based construction, enabling the model to encode not only direct connections but also the global smoothness structure of the network (Križmančić and Bogdan, 2024). Laplacian representations may provide richer spectral information for GCN layers and mitigate the oversimplification introduced by the MST.

### 7.2.3 Advancing Graph Learning Models

The empirical findings indicate that traditional spectral GCNs extract useful relational information only when labels encode cross-sectional ranking (quantile-based labels). Under threshold-based labels, GCN learning collapses toward near-random ranking, suggesting that the architecture is insufficiently expressive for absolute-return prediction. Although GCN adapts the hyperbolicity exhibited in our MST network inputs, its processing is still subject to Euclidean domain. Future development should therefore explore more advanced graph neural architectures—Graph Attention Networks, hyperbolic GCNs, or diffusion-based GNNs (Wu et al., 2020; Li et al., 2023; Khemani et al., 2024). These models may be able to capture finer-grained or nonlinear dependencies omitted by standard GCN layers. Because market structure evolves over time, incorporating temporal graph learning (e.g., TGAT or TGN models) may improve the stability of predictions by modeling how cross-stock relations change from window to window (Chen et al., 2025a).

### 7.2.4 Portfolio Construction

The investment performance results reveal that equal-weight long–short portfolios, while simple, may not fully exploit the predictive information contained in the CNN–MST–GCN framework. Future work should evaluate more sophisticated portfolio construction schemes, including volatility-scaled weights, risk budgeting, or optimization-based allocations that translate probabilistic predictions into position sizes aligned with expected risk–return trade-offs (Markowitz, 1952; Chow and Kritzman, 2001; Dalio

2005; Almahdi and Yang, 2017). Incorporating realistic trading frictions—transaction costs, bid–ask spreads, short-selling constraints—would enable more accurate assessment of economic value and prevent overestimation of strategy performance. Alternative long–short selection rules, such as decile-based ranking, dynamic position sizing based on prediction confidence, or holding-period adjustments based on volatility regimes, may yield further stability (Jiang et al., 2023; Zhu and Zhu, 2024). These extensions would bridge the gap between predictive modeling and implementable trading strategies.

### 7.2.5 Enhancing Labeling, Prediction Targets, and Robustness Analysis

The stark contrast between threshold-based and quantile-based performance highlights the central role of label design in deep-learning-based financial forecasting. Future work should consider moving beyond binary classification altogether. Regression-based targets for expected returns or volatility-adjusted returns could provide smoother supervisory gradients and reduce label noise (Schuhmacher and Eling, 2012; Steininger et al., 2021). Multi-class or ordinal ranking targets may better capture subtler distinctions among stocks and improve compatibility with graph-based propagation (Cao et al., 2020; Kamal and Farooq, 2024). Reinforcement learning frameworks may offer an alternative paradigm by directly optimizing reward-driven portfolio outcomes rather than predicting discrete labels (Iranfar et al., 2021). Robustness analysis—across similarity metrics, embedding layers, random seeds, normalization schemes, and other hyperparameters—is also critical, particularly given the sensitivity observed in GCN training behavior. Methods for uncertainty quantification or probabilistic calibration could further improve the reliability of predictions, especially in the presence of volatile or regime-dependent market structure (Horvath et al., 2021; Baschetti et al., 2024).

## 7.3 Concluding Remarks

This project develops a unified CNN–MST–GCN framework that integrates visual price-pattern extraction with relational learning across stocks, offering a novel approach to short-horizon equity prediction in a challenging market environment. Motivated by the limitations of mere-CNN models, which focus exclusively on individual price dynamics, the proposed methodology incorporates market-topology information through similarity-based financial networks and graph neural networks. What is more, it also generalizes the MPT principle such that the computer vision does not only “see” the individual stock behavior to earn maximized returns, but also put the eyes on the co-movements among the stocks to gain optimized risk-adjusted returns. The resulting system represents an interdisciplinary synthesis of computer vision, graph learning, and financial modeling.

Comprehensive experiments on HSI constituent stocks reveal several notable findings. First, the CNN alone exhibits strong predictive power, consistently outperforming the HSI benchmark and demonstrating that image-based representations contain meaningful signals even in a weak and stagnant market. Second, the effectiveness of the CNN–MST–GCN framework is highly dependent on the design of the supervisory target. Under threshold-based labeling, graph propagation fails to enhance predictive accuracy and results in underwhelming investment performance. In contrast, under quantile-based labeling—which better reflects cross-sectional structure—the MST–GCN component significantly improves both predictive and economic outcomes, producing higher cumulative returns with less downside risks than the CNN baseline. These results highlight the importance of aligning target design with the relational nature of graph learning.

At the same time, the study identifies key limitations that constrain overall performance, including the small and homogeneous stock universe, the sparsity imposed by MST topology, and the sensitivity of graph learning to labeling choices. These constraints point toward several promising directions for future improvement, such as expanding the dataset, enhancing image representations, exploring richer and more flexible graph constructions, and designing more expressive learning targets. Together, these insights emphasize that while the proposed framework demonstrates strong potential, its full capability will emerge only when the design of the neural network is more closely aligned with the mathematical properties of financial graphs and the financial theories that govern how price information should be processed and propagated across stocks.

Overall, this project provides evidence that combining computer vision techniques with graph neural networks by the bridge of graph theory offers a viable and effective direction for financial prediction. The findings show that deep learning models can extract both visual and relational signals that translate into meaningful investment performance, even under unfavorable market environment. By laying the groundwork for future enhancements, this study contributes an important step toward more sophisticated graph–vision hybrid models for quantitative finance.

**Acknowledgements**

We would like to sincerely thank Yu-Chien Tseng[◊] for providing valuable insights into the model design and for sharing parts of the CNN implementation. His ideas and code contributions were instrumental in the development of this work.

[◊] 曾宇謙. Department of Applied Mathematics, National Sun Yat-sen University (M122040014@student.nsysu.edu.tw)

## References

Aggarwal, R. and Rivoli, P. (1989) Seasonal and Day-of-the-Week Effects in Four Emerging Stock Markets. *Financial Review*, 24(4), 541-550.

Almahdi, S. and Yang, S.Y. (2017) An Adaptive Portfolio Trading System: A Risk-Return Portfolio Optimization using Recurrent Reinforcement Learning with Expected Maximum Drawdown. *Expert Systems with Applications*, 87, 267-279.

Banz, R. W. (1981). The Relationship between Return and Market Value of Common Stocks. *Journal of Financial Economics*, 9(1), 3-18.

Bardoscia, M., Barucca, P., Battiston, S., Caccioli, F., Cimini, G., Garlaschelli, D., Saracco, F., Squartini, T., and Caldarelli, G. (2021). The Physics of Financial Networks. *Nature Reviews Physics*, 3(7), 490-507.

Baschetti, F., Bormetti, G., & Rossi, P. (2024). Deep Calibration with Random Grids. *Quantitative Finance*, 24(9), 1263-1285.

Berg, R.V.D., Kipf, T.N., and Welling, M. (2017) Graph Convolutional Matrix Completion. *arXiv preprint arXiv:1706.02263.*

Cai, T., Luo, S., Xu, K., He, D., Liu, T. Y., and Wang, L. (2021) GraphNorm: A Principled Approach to Accelerating Graph Neural Network Training. In *Proceedings of the 38th International Conference on Machine Learning*, PMLR 139, 1204-1215.

Cao, W., Mirjalili, V., and Raschka, S. (2020) Rank Consistent Ordinal Regression for Neural Networks with Application to Age Estimation. *Pattern Recognition Letters*, *140*, 325-331.

Cao, P., Zhu, Z., Wang, Z., Zhu, Y., and Niu, Q. (2022) Applications of Graph Convolutional Networks in Computer Vision. *Neural Computing and Applications*, 34(16), 13387-13405.

Chen, M., Wei, Z., Huang, Z., Ding, B., and Li, Y. (2020) Simple and Deep Graph Convolutional Networks. In *International Conference on Machine Learning*, PMLR 119, 1725-1735.

Chen, W., Jiang, M., Zhang, W. G., and Chen, Z. (2021) A Novel Graph Convolutional Feature Based Convolutional Neural Network for Stock Trend Prediction. *Information Sciences*, 556, 67-94.

Chen, X., Hao, A., and Li, Y. (2020) The Impact of Financial Contagion on Real Economy-An Empirical

Research based on Combination of Complex Network Technology and Spatial Econometrics Model. *PloS ONE*, 15(3), e0229913.

Chen, D., Zheng, S., Xu, M., Zhu, Z., and Zhao, Y. (2025a) SiGNN: A Spike-induced Graph Neural Network for Dynamic Graph Representation Learning. *Pattern Recognition*, 158, 111026.

Chen, Z., Lin, Z., Chen, S., Polyanskiy, Y., and Rigollet, P. (2025b) Residual Connections Provably Mitigate Oversmoothing in Graph Neural Networks. *arXiv preprint arXiv:2501.00762*.

Chow, G. and Kritzman, M. (2001) Risk Budgets. *Journal of Portfolio Management*, 27(2), 56-60.

Coşkun, K., Kavisanczki, I., Mirzaei, A., Siegl, T., Hiller, B. C., Lüdtke, S., and Becker, M. (2025) Informed, but Not Always Improved: Challenging the Benefit of Background Knowledge in GNNs. *arXiv preprint arXiv:2505.11023*.

Dalio, R. (2005). *Engineering Targeted Returns and Risks*. Westport, CT: Bridgewater Associates.

DeMiguel, V., Garlappi, L., and Uppal, R. (2009) Optimal Versus Naive Diversification: How Inefficient is the 1/N Portfolio Strategy? T*he Review of Financial Studies*, 22(5), 1915-1953.

Fama, E.F. and French, K.R. (1992) The Cross-Section of Expected Stock Returns. *the Journal of Finance*, 47(2), 427-465.

He, K., Zhang, X., Ren, S., and Sun, J. (2016a). Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 770-778.

He, K., Zhang, X., Ren, S., and Sun, J. (2016b) Identity Mappings in Deep Residual Networks. In *European Conference on Computer Vision*, 630-645. Cham: Springer International Publishing.

Hendrycks, D. (2016) Gaussian Error Linear Units (Gelus). *arXiv preprint arXiv:1606.08415.*

Horvath, B., Muguruza, A., and Tomas, M. (2021) Deep Learning Volatility: A Deep Neural Network Perspective on Pricing and Calibration in (Rough) Volatility Models. *Quantitative Finance*, 21(1), 11-27.

Iranfar, A., Zapater, M., and Atienza, D. (2021) Multi-Agent Reinforcement Learning for Hyperparameter Optimization of Convolutional Neural Networks. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 41(4), 1034-1047.

Jiang, J., Kelly, B., and Xiu, D. (2023) (Re-)Imag(in)ing Price Trends. *The Journal of Finance*, 78(6), 3193–3249.

Jiang, M., Liu, G., Su, Y., and Wu, X. (2024) Self-Attention Empowered Graph Convolutional Network for Structure Learning and Node Embedding. *Pattern Recognition*, 153, 110537.

Jobson, J.D. and Korkie, R. (1980) Estimation for Markowitz Efficient Portfolios. *Journal of the American Statistical Association*, 75(371), 544–554.

Kan, R. and Zhou, G. (2007) Optimal Portfolio Choice with Parameter Uncertainty. *Journal of Financial and Quantitative Analysis*, 42(3), 621–656.

Kamal, K. and Farooq, B. (2024) Ordinal-ResLogit: Interpretable Deep Residual Neural Networks for Ordered Choices. *Journal of Choice Modelling*, 50, 100454.

Kawamoto, T., Tsubaki, M., and Obuchi, T. (2018) Mean-Field Theory of Graph Neural Networks in Graph Partitioning. *Advances in Neural Information Processing Systems*, *31*.

Khemani, B., Patil, S., Kotecha, K., and Tanwar, S. (2024) A Review of Graph Neural Networks: Concepts, Architectures, Techniques, Challenges, Datasets, Applications, and Future Directions. *Journal of Big Data*, 11(1), 18.

Klabunde, M., Schumacher, T., Strohmaier, M., and Lemmerich, F. (2025) Similarity of Neural Network Models: A Survey of Functional and Representational Measures. *ACM Computing Surveys*, 57(9), 1-52.

Križmančić, M. and Bogdan, S. (2024) Adaptive Connectivity Control in Networked Multi-Agent Systems: A Distributed Approach. *PLoS ONE*, 19(12), e0314642.

Liu, Y., Lang, B., and Quan, F. (2023) MST-HGCN: A Minimum Spanning Tree Hyperbolic Graph Convolutional Network. *Applied Intelligence*, 53(11), 14515-14526.

Loshchilov, I. and Hutter, F. (2016) Sgdr: Stochastic Gradient Descent with Warm Restarts. *arXiv preprint arXiv:1608.03983*.

Loshchilov, I. and Hutter, F. (2017) Decoupled Weight Decay Regularization. *arXiv preprint arXiv:1711.05101*.

Ma, D. and Yuan, D. (2024) Enhanced Stock Price Forecasting Through a Regularized Ensemble

Framework with Graph Convolutional Networks. *Expert Systems with Applications*, 250, 123948.

Ma, D., Yuan, D., Huang, M., and Dong, L. (2024) VGC-GAN: A Multi-Graph Convolution Adversarial Network for Stock Price Prediction. *Expert Systems with Applications*, 236, 121204.

Malkiel, B.G. (1995) Returns from Investing in Equity Mutual Funds 1971 to 1991. *The Journal of Finance*, 50(2), 549-572.

Mantegna, R.N. (1999) Hierarchical Structure in Financial Markets. *The European Physical Journal B-Condensed Matter and Complex Systems*, 11(1), 193-197.

Markowitz, H.M. (1952) Portfolio Selection. *The Journal of Finance,* 7(1), 77–91.

Mehrer, J., Spoerer, C. J., Kriegeskorte, N., and Kietzmann, T. C. (2020) Individual Differences among Deep Neural Network Models. *Nature Communications*, 11(1), 5725.

Mehrkanoon, S. (2021) Deep Graph Convolutional Networks for Wind Speed Prediction. *arXiv preprint arXiv:2101.10041.*

Millington, T. and Niranjan, M. (2021) Construction of Minimum Spanning Trees from Financial Returns using Rank Correlation. *Physica A: Statistical Mechanics and its Applications,* 566, 125605.

Narayanaswamy, R.A. and Gowda, V.K. (2024) Integrated VGG19 and Gated Recurrent Unit with Cosine Similarity Metric for Content based Image Retrieval. In *2024 First International Conference on Software, Systems and Information Technology (SSITCON)*, Tumkur, India, pp. 1-5. IEEE.

Olmo, J. (2021) Optimal Portfolio Allocation and Asset Centrality Revisited. *Quantitative Finance*, 21(9), 1475-1490.

Pástor, Ľ. and Stambaugh, R.F. (2000) Comparing Asset Pricing Models: An Investment Perspective. *Journal of Financial Economics*, 56(3), 335–381.

Peng, Y., Guo, Y., Hao, R., and Xu, C. (2024) Network Traffic Prediction with Attention-Based Spatial–Temporal Graph Network. *Computer Networks*, 243, 110296.

Peralta, G. and Zareei, A. (2016) A Network Approach to Portfolio Selection. *Journal of Empirical Finance*, 38(A), 157-180.

Pozzi, F., Di Matteo, T., and Aste, T. (2013) Spread of Risk across Financial Markets: Better to Invest in the Peripheries. *Scientific Report*, 3(1665), 1-7.

Schuhmacher, F. and Eling, M. (2012) A Decision-Theoretic Foundation for Reward-to-Risk Performance Measures. *Journal of Banking & Finance*, 36(7), 2077-2082.

Sharpe, W.F. (1966) Mutual Fund Performance. *Journal of Business*, 39(1), 119–138.

Shi, Y., Qu, Y., Chen, Z., Mi, Y., and Wang, Y. (2024) Improved Credit Risk Prediction based on an Integrated Graph Representation Learning Approach with Graph Transformation. *European Journal of Operational Research*, 315(2), 786-801.

Shumway, T. (1997) The Delisting Bias in CRSP Data. *The Journal of Finance*, 52(1), 327-340.

Steininger, M., Kobs, K., Davidson, P., Krause, A., and Hotho, A. (2021) Density-Based Weighting for Imbalanced Regression. *Machine Learning*, 110(8), 2187-2211.

Tewarie, P., van Dellen, E., Hillebrand, A., and Stam, C. J. (2015) The Minimum Spanning Tree: An Unbiased Method for Brain Network Analysis. *Neuroimage*, 104, 177-188.

Tumminello, M., Aste, T., Di Matteo, T., and Mantegna, R. N. (2005) A Tool for Filtering Information in Complex Systems. *Proceedings of the National Academy of Sciences*, 102(30), 10421-10426.

Wang, R. and Li, J. (2025) Fast Sparse Representative Tree Splitting via Local Density for Large-Scale Clustering. *Scientific Reports*, 15(1), 29398.

Wu, Z., Pan, S., Chen, F., Long, G., Zhang, C., and Yu, P. S. (2020) A Comprehensive Survey on Graph Neural Networks. *IEEE Transactions on Neural Networks and Learning Systems*, 32(1), 4-24.

Zhang, L., Yan, X., He, J., Li, R., and Chu, W. (2023) DRGCN: Dynamic Evolving Initial Residual for Deep Graph Convolutional Networks. In *Proceedings of the AAAI Conference on Artificial Intelligence* (Vol. 37, No. 9, pp. 11254-11261).

Zhu, Z., and Zhu, K. (2024) Enhancement of Price Trend Trading Strategies via Image-Induced Importance Weights. *arXiv preprint arXiv:2408.08483*.