Time Evolving Spatio-Temporal Hypergraph for Stock Forecasting

Ramit Sawhney*
ramits@iiitd.ac.in
Georgia Institute of Technology
USA

Shivam Agarwal*
Manipal Institute of Technology
India

Megh Thakkar*
BITS Pilani
India

Vineet Malhotra
IIIT Delhi
India

Sudheer Chava
sudheer.chava@scheller.gatech.edu
Georgia Institute of Technology
USA

ABSTRACT

Selecting the most profitable stocks to trade is a complex financial task as future stock trends are highly unpredictable and non-stationary. Despite advances in stock forecasting, existing methods that leverage the inter-connectivity of stocks do not account for the stochasticity of this underlying network over the trading period among these stocks. To this end, we propose TEDHGN: Time Evolving Dynamic Stock Hypergraph Network for Quantitative day trading. We model the complex relations between stocks through a time-evolving dynamic hypergraph and tailoring a new hypergraph deep neural network that jointly captures the interdependence between stocks and the temporal evolution of their prices and relations with other stocks. Through extensive experiments on real-world markets: NASDAQ, NYSE, and China Exchanges spanning over six years, we show that TEDHGN outperforms state-of-the-art stock forecasting methods by 24% in terms of risk-adjusted returns.

CCS CONCEPTS

• **Computing methodologies → Artificial intelligence**; • **Social and professional topics** → *Economic impact*.

KEYWORDS

stock market, graph neural network, hypergraphs, finance

ACM Reference Format:

Ramit Sawhney, Shivam Agarwal, Megh Thakkar, Vineet Malhotra, and Sudheer Chava. 2022. Time Evolving Spatio-Temporal Hypergraph for Stock Forecasting. In *Proceedings of The 15th ACM International Conference on Web Search and Data Mining (WSDM).* ACM, New York, NY, USA, 8 pages. https://doi.org/XXXXXXX.XXXXXXX

1 INTRODUCTION

Globally, the multi-trillion-dollar stock market observes billions of dollars of traded volume daily, leading to traders and investors adopting diverse investment strategies for profit generation. At the core of such strategies lies the fundamental task of stock price movement prediction. However, stock prices are often stochastic in nature, making it hard to predict their future movements and trends [1], in turn making it difficult to execute profitable trades and develop successful investment strategies. Prices are driven by a multitude of factors, including company performance, historical trends, public sentiment, and more. Consequently, recent efforts revolve around leveraging historic financial factors, investor sentiments across social media [41], financial news [19], and inter-stock relations [23] for stock prediction. Efforts have been made to exploit knowledge graphs for representing stocks as nodes and their relations as edges (pairwise modeling). This representation is followed by employing graph-based deep learning, for instance, using graph neural networks (GNNs) for stock movement prediction [46]. The methods however represent stock interactions is that they utilize an oversimplified model of the stock market with a simple graph consisting of pairwise relations between individual companies when in reality, the system inherently consists of higher-order complex relations among stocks [2]. Sawhney et al. [35] develops over simple graph based methods and uses hypergraph learning to capture these complex relations.

However, a limitation of existing graph-based methods is that they only model static stock relations, for instance, corporate relationships or sector-industry groupings [13, 23]. Often in reality, there exist dynamic correlations among stocks [33] that can be leveraged to enhance graph-based stock prediction. For instance, a one-time event related to a group of companies, which are otherwise not connected through corporate or industry relationships, may lead to a similar surge in their stock prices. In such cases, the static graph connections based on pre-defined static relations would fail to model the impact of the new event across disconnected stocks, hampering stock prediction performance. Owing to the complex interactions between the various elements of a stock market, it is imperative to regard the dynamic correlations between different stocks in a financial market.

Building on the gaps in existing research, we model stocks as a hypergraph, and propose **TEDHGN: T**ime **E**volving **D**ynamic stock **H**yper**G**raph **N**etwork for mid-frequency stock trading, a neural framework that learns to rank stock based on expected profits **(Sec. 3.1)** for maximising daily returns. TEDHGN learns the collective synergy between stock movements through hypergraph learning based on industrial taxonomies, corporate relationships, and dynamic stock correlations in a time-sensitive manner **(Sec.**

*Equal contribution.

WSDM, February 25, 2022, Phoenix, AZ

ACM ISBN 978-1-4503-XXXX-X/18/06…$15.00
https://doi.org/XXXXXXX.XXXXXXX

3.3). Building on the success of graph-based learning, TEDHGN uses dynamic hypergraph convolutions **(Sec.3.4)** to learn the relationships and price trends among related stocks. TEDHGN exploits temporal attention and gated temporal convolution mechanisms to model the temporal evolution of stock price movements **(Sec. 3.2)**. Through extensive experimentation **(Sec. 4)** on real-world data **(Sec. 4.1)** pertaining to NASDAQ, NYSE, Shanghai, Shenzhen, and Hong Kong markets, we show that TEDHGN significantly outperforms state-of-the-art methods **(Sec. 5.1)** for quantitative day trading, with an improvement of 24% in terms of risk-adjusted returns. Lastly, we probe into an ablation analysis **(Sec. 5.2)** and exploratory studies of TEDHGN **(Sec. 5.3, Sec. 5.2)** to contextualize each component's effectiveness for quantitative day trading and pave the future directions towards more dynamic hypergraph problems **(Sec. 6)**. Our contributions can be summarized as:

- We blend temporal attention with gated temporal convolutions and dynamic hypergraph convolutions and propose **TEDHGN**. TEDHGN is capable of efficient modeling of dynamic Spatio-temporal dependencies among the stocks.
- We explore local and global hidden relationships among stocks using k-means clustering to construct dynamic hypergraphs that extend the hyperedge set defined via static stock relations.
- Through experiments on three real-world stock indexes in NYSE, NASDAQ, and China markets, over 2,848 stocks spanning 1,245 trading days, we demonstrate the applicability of **TEDHGN** towards stock prediction and forecasting.

2 RELATED WORK

Traditional Methods: Most existing methods formulate stock prediction as either classification or regression tasks [22], or ranking tasks [35, 37]. Predicting the stock prices or directional trends in stock movements finds numerous practical applications, which include designing investment strategies [9], portfolio management [17], and much more. To predict stock movements, financial models conventionally focused on technical analysis (TA). These TA methods include discrete [10], continuous [3], and neural approaches [32].

Contemporary Methods: Recent works employ graph-based methods to model pairwise relations among stocks using sector-industry metadata and links between managerial and board members of companies [31]. These works employ variants of graph neural networks (GNNs) to model the the price movements of related stocks. In a recent study, [23] proposes an attention-based GNN for stock movement prediction. [13] uses graph convolution along with temporal convolutions in an attempt to demonstrate the benefits of studying temporal price evolution along with inter-stock relations over stock movement prediction. Sawhney et al. [37] and Sawhney et al. [35] use hypergraph learning toto capture the higher-order complex relations among stocks. A limitation of graph-based methods is that they do not dynamically capture relationships among the nodes, and to counter this limitation, various attempts have been made at dynamically learning hidden relations in graphs [15, 34, 44].

Hypergraph Representation and Learning: Hypergraphs have proved to be an efficient approach for a variety of applications [6, 14, 42, 49, 50] and numerous tasks, including visual object recognition [14], emotion recognition [38], classification of gene expression [39], owing to its ability to extract patterns from higher-order relationships. However, a gap in existing hypergraph neural networks is that they are not specifically designed for temporal learning from time-evolving features such as daily stock prices. Limited early research such as Han et al. [16] demonstrated the effectiveness of clustering stocks as hypergraphs based on stock activity data, using association rule clustering. Luo et al. [29] and Yang Shen et al. [43] have previously represented stocks via hypergraphs by forming hyperedges that group stocks having similar price movement trends. Recent studies, such as [48] aim to optimize the hypergraph structure along with the learning task simultaneously. Jiang et al. [20] proposed to utilize kNN to dynamically create hyperedges. However, [20] treats the nodes in a similar fashion at all time steps, inherently disregarding the temporal information among nodes along different time-steps.

3 METHODOLOGY

3.1 Problem Formulation

Following Sawhney et al. [35], we formulate stock trading as a learning to rank task. Given a set $\mathcal{S} = \{s_1, s_2, \ldots, s_N\}$, of N stocks, with a corresponding closing price p_i^t for each stock $s_i \in S$ on trading day t, we define the 1-day return ratio $r_i^t = \frac{p_i^t - p_i^{t-1}}{p_i^{t-1}}$. On any given trading day t, there exists an optimal ranking $Y^t = \{y_1^t > y_2^t \cdots > y_N^t\}$ of the stocks, such that $y_i^t > y_j^t$ for any two stocks $s_i, s_j \in \mathcal{S}$, if their associated return ratios $r_i^t > r_j^t$. Such an ordering of stocks $\mathcal{S}$ represents a ranking list, where stocks achieving higher ranking scores Y are expected to achieve a higher investment revenue (profit) on day t. Formally, given historic stock data from lookback period of length τ (i.e., $[t - \tau, t - 1]$), we target to learn a ranking function that outputs a score $\hat{y}_i^t$ to rank each stock s_i on day t in terms of expected profit.

We present an overview of TEDHGN in Figure 1. We first explain how we extract temporal features from the evolution of historical stock prices (**Sec. 3.2**), construct a time-evolving dynamic hypergraph between stocks (**Sec. 3.3**), perform attentive dynamic hypergraph convolutions over the hypergraphs for node aggregation (**Sec. 3.4**), and then combine the temporal and dynamic hypergraph convolutions to capture the spatiotemporal evolution of stock features (**Sec. 3.5**). We optimize TEDHGN to rank stocks in terms of their expected profitability in **Sec. 3.6**.

3.2 Attentive Gated Temporal Convolution

Research by finance experts shows that historical stock prices can be a strong measure to predict future stock movements [32]. However, the influence of each historical point is not uniform, but is variable. To capture this variation, we follow Sawhney et al. [36] and employ a temporal attention mechanism, which learns to capture the varying influence of different historical stock prices. We first describe a single attentive gated temporal convolution layer here, which is used throughout the TEDHGN framework. The attentive gated temporal convolution consists of an attention mechanism followed by a gated temporal convolution. Given C is the number of features per stock, we feed stock features $\boldsymbol{X}^l \in \mathbb{R}^{N \times C \times \tau}$ corresponding to

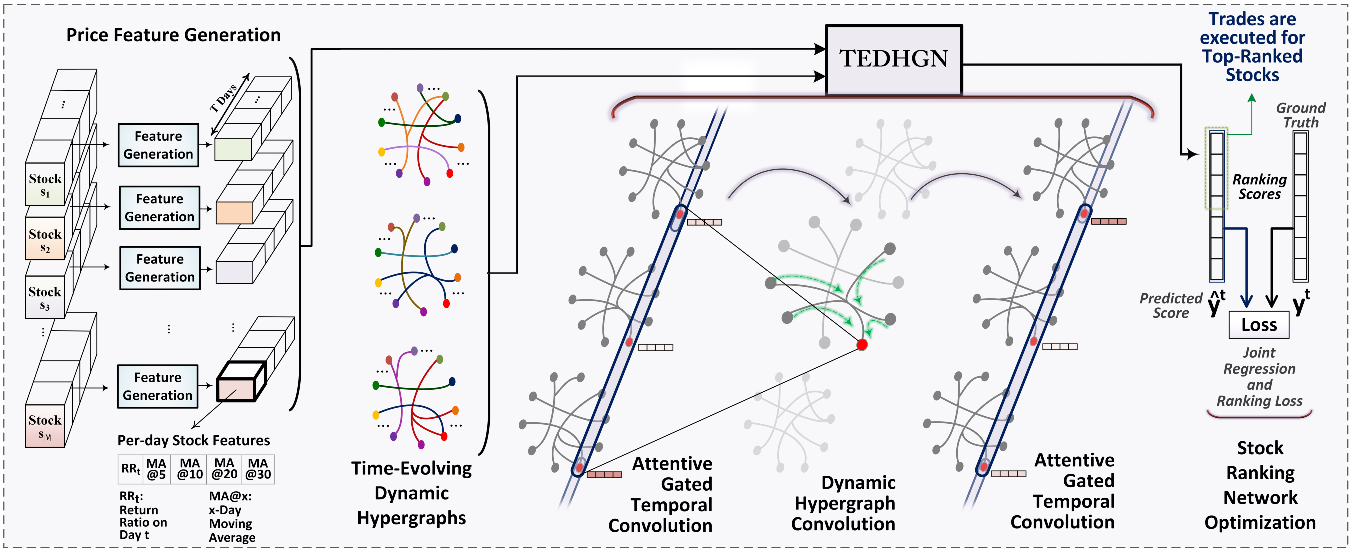

Figure 1: An overview of TEDHGN, feature extraction, time evolving dynamic hypergraph construction, dynamic hypergraph convolutions and optimisation.

all the N different stocks for a lookback period τ to the l^{th} attentive gated temporal convolution layer. This attentive layer is parameterized by a matrix $\boldsymbol{E}$, where the element $E_{i,j}$ learns the extent of the temporal dependency between stock features of days i and j. We perform a softmax operation over $\boldsymbol{E}$ as $\boldsymbol{E'} = \text{softmax}(\boldsymbol{E})$ to normalize the attention coefficients across different days. Formally,

$$\boldsymbol{E} = \boldsymbol{V} \cdot \sigma\left((\boldsymbol{X}^{l^T} \boldsymbol{U}_1) \boldsymbol{U}_2 (\boldsymbol{U}_3 \boldsymbol{X}^l) + \boldsymbol{b} \right)$$
$$E'_{i,j} = \frac{\exp(E_{ij})}{\sum_{j=1}^{\tau} \exp(E_{i,j})} \tag{1}$$

where, $\boldsymbol{V}, \boldsymbol{b} \in \mathbb{R}^{\tau \times \tau}$, $\boldsymbol{U}_1 \in \mathbb{R}^N$, $\boldsymbol{U}_2 \in \mathbb{R}^{C \times N}$, $\boldsymbol{U}_3 \in \mathbb{R}^C$ are learnable parameters and σ is logistic sigmoid activation function. Following Sawhney et al. [36], we apply the temporal attention matrix $\boldsymbol{E'}$ to the input $\boldsymbol{X}^l$ and learn a feature vector $\boldsymbol{X}^l_{\text{attn}}$,

$$\boldsymbol{X}^l_{\text{attn}} = \boldsymbol{X}^l \boldsymbol{E}' = (\boldsymbol{X}^l_1, \boldsymbol{X}^l_2, \cdots, \boldsymbol{X}^l_\tau) \boldsymbol{E}' \tag{2}$$

We now describe the gated temporal convolution operation $*$. The GLU consists of gating mechanisms that highlight important information in the temporal feature vectors [47]. The input to this gated temporal convolution layer is the output of the temporal attention mechanism $\boldsymbol{X}^l_{\text{attn}}$. The trainable convolution kernel $\boldsymbol{\Gamma} \in \mathbb{R}^{K_t \times C \times 2F}$ transforms the input $\boldsymbol{X}^l_{\text{attn}}$ to output $\boldsymbol{X}^{l'}_{\text{attn}} \in \mathbb{R}^{N \times (T-K_t+1) \times 2F}$ where, $2F$ is the number of output features per time-step. The GLU produces the output $\boldsymbol{X}^{l+1} \in \mathbb{R}^{N \times (T-K_t+1) \times F}$ given by,

$$\boldsymbol{X}^{l+1} = \boldsymbol{X}^l_{\text{attn}} * \boldsymbol{\Gamma} = \boldsymbol{A} \cdot \sigma(\boldsymbol{B}) \tag{3}$$

where, $\cdot$ represents point-wise product.

3.3 Dynamic Hypergraph Construction

Financial literature shows that relations between companies evolve over time [18] and are dynamic in nature [4]. We model the dynamic stock interdependence using dynamic hypergraphs, where the nodes represent the stocks, and the hyperedges model the higher-order relations between the stocks.

Hypergraph Initialisation: We construct a hypergraph $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathbf{W})$ where each vertex $v \in \mathcal{V}$ represents a stock $s \in S$, and each hyperedge $e \in \mathcal{E}$ represents a subset of related stocks $\{s_1, s_2, \ldots, s_n\} \in S$. Each hyperedge e is assigned a positive weight $w(e)$ with all weights stored in a diagonal matrix $\mathbf{W} \in \mathbb{R}^{|\mathcal{E}| \times |\mathcal{E}|}$. Following previous works, we let $\mathbf{W} = \mathbf{I}$ indicating equal weights for all hyperedges. We inject domain knowledge by constructing hyperedges between stocks for hypergraph initialization.

Dynamic Hypergraph Update Rule: In order to update the initialised hypergraph $\mathcal{G}$, we combine k-means and k-Nearest Neighbors clustering methods [21]. Given node features $\boldsymbol{X}^l = [x_1, \cdots x_N]$ and adjacent hyperedge set size H we perform k-means (KMeans($\cdot$)) algorithm on the whole feature map $\boldsymbol{X}^l$ of each layer l according to Euclidean distance and obtain clusters C, given by,

$$C = \text{KMeans}(\boldsymbol{X}^l) \tag{4}$$

For each node s, we find the nearest $H-1$ clusters based on the Euclidean distance (topK($\cdot$)) and assign them as adjacent hyperedges $\mathcal{E}_{new} = \{e_1, \cdots, e_{H-1}\}$ of the node s. Formally, for each node s we find the set of adjacent hyperedges $\mathcal{E}_{new}$ as,

$$\mathcal{E}_{new} = \text{topK}(C, H-1, \boldsymbol{X}^l[s]) \tag{5}$$

where, $\boldsymbol{X}^l[s]$ denotes the feature vector of node s.

On the other hand, given hyperedge size k, we compute $k-1$ nearest neighbors (KNN($\cdot$)) of each node s. These neighboring nodes along with the node s forms a new hyperedge e_{new} in the hypergraph $\mathcal{G}$. We obtain the hyperedge e_{new} as,

$$e_{\text{new}} = \text{KNN}(\boldsymbol{X}^l[s], \boldsymbol{X}^l, k) \tag{6}$$

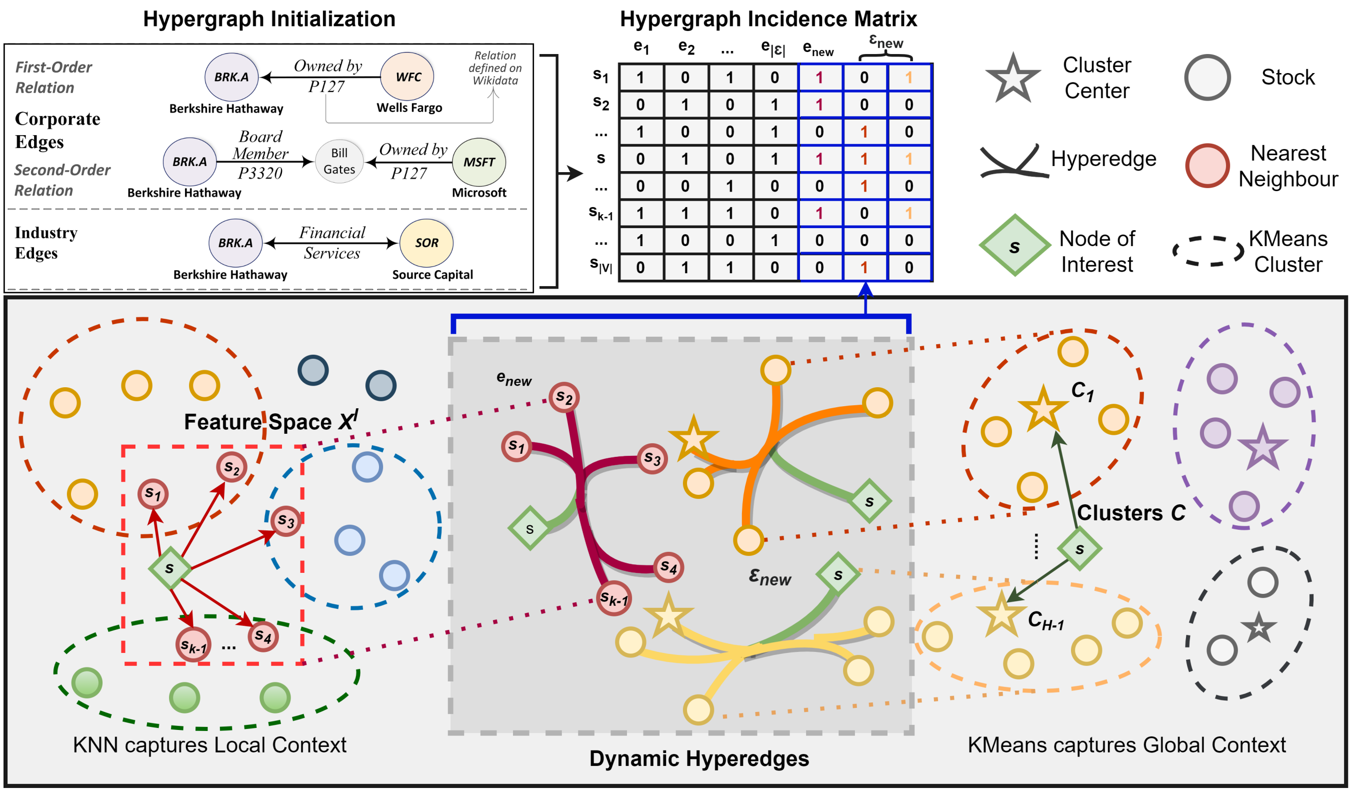

Figure 2: Dynamic hypergraph initialisation and update rule for one time-step. Top: Stock hypergraph initialisation using corporate and industry relations as hyperedges. Bottom: Dynamic hypergraph convolution update rule. The initialised hyergraph is updated on each time step using KNN and KMeans clustering.

3.4 Dynamic Hypergraph Convolution

To learn the price movement correlations between multiple related stocks, we use a dynamic hypergraph convolution [6] on the dynamic stock relation hypergraph $\mathcal{G}$. The dynamic hypergraph convolution comprises two modules stacked one after the other: (1) Node Convolution and (2) Hyperedge Convolution. We detail the node convolution first.

Node Convolution: The node convolution aggregates node features X^l to the hyperedge containing these nodes/stocks. This convolution first applies a transformation matrix W on each node to obtain more expressive node features. We then use a 1-dimension convolution (Conv($\cdot$)) to combine the node features into a hyperedge feature x_e. Formally, given input features X^l we describe the node convolution as:

$$x_e = \text{Conv}(W \cdot \text{MLP}(X^l)) \quad W = \text{MLP}(X^l) \tag{7}$$

where, $\cdot$ represents point-wise product and MLP denotes a fully connected layer.

Hyperedge Convolution: The hyperedge convolution aggregates hyperedge features x_e to centroid node features x_u for each node u. To learn the varying degree of influence each stock relation (Hyperedge) has on each stock; the hyperedge convolution is based on the self-attention mechanism. For each node/stock, the attention mechanism learns weights w corresponding to each incident hyperedge. The output features of the centroid node x_u is computed as a linear combination of hyperedge features x_e and the attention scores w. Formally, we define this attention mechanism as,

$$x_u = \sum_i w^i x_e^i \quad w = \text{softmax}(x_e P + b) \tag{8}$$

where, P, b are the weights and biases of a feed forward neural network and the summation is taken over all adjacent hyperedges on node u.

We now combine the node convolution with the self-attention based hyperedge convolution to devise the dynamic hypergraph convolution DHC($\cdot$). The dynamic hypergraph convolution updates node features $F \in \mathbb{R}^{N\times C}$ to a new output features $F' \in \mathbb{R}^{N\times C'}$ where C and C' are the number of input and output features respectively. Formally, given a hypergraph $\mathcal{G}$ the update rule of the dynamic hypergraph convolution is given as,

$$F' = \text{DHC}(F, \mathcal{G}) \tag{9}$$

We now generalise the above dynamic hypergraph convolution layer to temporal input features $X^l \in \mathbb{R}^{N\times\tau\times C}$ and denote it as DHC$^\tau(\cdot)$. The dynamic hypergraph convolution applies the same dynamic hypergraph convolution DHC($\cdot$) to all time steps in the of the input, and produces a new set of features $X^{l+1} \in \mathbb{R}^{N\times\tau\times C'}$, given by,

$$X^{l+1} = \mathrm{DHC}^{\tau}(X^{l}, \mathcal{G}) \tag{10}$$

3.5 Dynamic Spatio-Temporal Hypergraph Convolution

We combine attentive gated temporal and spatial hypergraph convolutions as a dynamic Spatio-temporal hypergraph convolution block, $\mathrm{DSTC}(\cdot)$ to jointly learn the dynamic correlations between related stocks and the temporal evolution of stock prices. We first describe a single $\mathrm{DSTC}(\cdot)$ block that is stacked to build TEDHGN. We sandwich the dynamic hypergraph convolution between two attentive gated temporal convolutions. The dynamic hypergraph convolution acts as a bridge between the two attentive gated temporal convolutions. This design choice allows the propagation of spatially updated features along the time axis through attentive gated temporal convolutions. Formally, we feed temporal stock features $Q \in \mathbb{R}^{\tau \times N \times C}$ to TEDHGN's first layer X^0. For all N stocks, we feed $C = 5$ temporal features: 1-day return ratio, 5, 10, 20 and 30 day moving averages, which represent the daily, weekly, and monthly trends over a lookback of τ days. We feed these temporal features Q to the attentive gated convolution to produce output X^1 followed by the dynamic hypergraph construction module to construct the hypergraph $\mathcal{G}$. We then apply the dynamic hypergraph convolution on the dynamically constructed hypergraph $\mathcal{G}$. Then, we employ the second attentive gated temporal convolution, which produces the final output X^2.

We summarise these transformations made by TEDHGN to the input features Q to obtain the learned Spatio-temporal stock features X^2 as:

$$\begin{aligned} X^2 &= \mathrm{DSTC}(Q, \Gamma^0, \Gamma^1, E'^0, E'^1, \mathcal{G}) \\ &= \left(\mathrm{DHC}^{\tau}\left((QE'^0) * \Gamma^0, \mathcal{G}\right) E'^1\right) * \Gamma^1 \end{aligned} \tag{11}$$

where, Γ^0, Γ^1 are weights of the first and second gated temporal convolutions, respectively. E'^0, E'^1 represent the attention coefficients of the first and second temporal attention mechanisms, respectively.

3.6 Network Optimization

We apply two dynamic Spatio-temporal hypergraph convolution layers $\mathrm{DSTC}(\cdot)$ with ReLU activation function between the first and the second layer. The outputs of the second layer are fed into an output layer to obtain a ranked list of stocks $\hat{y}^t$. The output layer consists of an attentive gated temporal convolution layer followed by a fully connected layer. The output layer projects the output of the final $\mathrm{DSTC}(\cdot)$ block to a single time-step prediction. Formally, we obtain the predicted ranking list of stocks $\hat{y}^t$ as:

$$\hat{y}^t = \left(W\left(\mathrm{DSTC}\left(\mathrm{DSTC}(Q\Gamma^0, \Gamma^1, E'^0, E'^1, \mathcal{G})\right), \Gamma^2, \Gamma^3, E'^2, E'^3, \mathcal{G}\right) E'^4\right) * \Gamma^4 + b \tag{12}$$

where, W, b are the weights and biases of the fully connected layer and Γ^4, E'^4 are the parameters of the attentive gated temporal convolution in the output layer.

We optimize TEDHGN using a combination of a pointwise regression and pairwise ranking-aware loss to minimize the difference between the predicted and actual return ratios while maintaining the relative order of top-ranked stocks with higher expected return for investment as:

$$L = \|\hat{y}^t - y^t\|^2 + \beta \sum_{i=0}^{N} \sum_{j=0}^{N} \max\left(0, -\left(\hat{y}_i^t - \hat{y}_j^t\right)\left(y_i^t - y_j^t\right)\right) \tag{13}$$

where, $\hat{y}^t$ and y^t are the predicted and actual ranking scores, respectively, and β is a loss weighting parameter.

Table 1: Dataset statistics detailing number of time-stamps of the three markets along with their corresponding initial hypergraph statistics.

	NASDAQ	NYSE	CSE
Train Period (Tr)	01/13-12/15	01/13-12/15	12/14-05/18
Val Period (Va)	01/16-12/16	01/16-12/16	05/18-06/19
Test Period (Te)	01/17-12/17	01/17-12/17	06/19-07/20
No. of Days	1245	1245	1293
# Days Tr:Va:Te	756:252:237	756:252:237	756:252:285
No. of HG Nodes	1026	1737	85
No. of Hyperedges	862	1595	89

4 EXPERIMENTS

4.1 Datasets and Preprocessing

For an extensive evaluation of TEDHGN and other models, we use *three* datasets extracted from real-world stock markets based in *US* and *China* and contain data spanning over *six* years. We summarise the dataset statistics in Table 1 . For a fair comparison, we follow the same data pre-processing as adopted by the works that introduced the datasets.

4.2 Evaluation Metrics

Returns based metrics. To evaluate all models' profitability, we use Sharpe Ratio (SR) and the Cumulative Investment Return Ratio (IRR). We follow the trading strategy in [13] to compute these metrics .

Ranking based metrics. We evaluate the ranking ability of our model using Normalized Discounted Cumulative Gain (NDCG@k). NDCG is one of the most commonly used metrics for assessing ranking quality.

For both returns and ranking metrics, we report results for top 5 stocks.

4.3 Training Setup

We perform all experiments on a Tesla P100 GPU. We use grid search to find optimal hyperparameters to maximize the validation NDCG@5 for all models. We experiment with the lookback window range $\tau \in [2, 20]$, loss weighting factor $\beta \in [1, 10]$, and learning rate $\in [1e-03, 5e-04]$. All learnable weights are initialized with Xavier uniform initialization and the convolution layers have 64 temporal and 16 spatial channels. For the KMeans clustering, we experiment with $H = \{8, 12, 16\}$ clusters. We experiment with $k = \{8, 16, 32\}$ nearest neighbors for the KNN based dynamic hypergraph construction. We train all models for 50 epochs and use Adam [25] optimizer with a weight decay of $5e-4$.

Table 2: Profitability comparison with classification, regression, reinforcement learning and ranking methods (average of 5 independent runs). Purple and Pink represent best and second best results, respectively. $*$ and $\diamond$ indicate the improvement over RSR-I and STHAN-SR, respectively, is statistically significant ($p < 0.01$), under the Wilcoxon's Signed Rank Test.

Formulation	Model	NASDAQ		CSE		NYSE	
		SR@5	IRR@5	SR@5	IRR@5	SR@5	IRR@5
Regression Methods	**SFM** [45]	0.16	0.09	0.21	0.31	0.19	0.11
	LSTM [7]	0.48	0.13	0.17	0.63	0.13	0.09
Classification Methods	**ARIMA** [40]	0.55	0.10	0.37	0.43	0.33	0.10
	HGCluster [30]	0.06	0.10	0.15	0.20	0.10	0.11
	A-LSTM [12]	0.97	0.23	0.83	0.80	0.81	0.14
	GCN [27]	0.75	0.13	0.73	0.75	0.70	0.10
	HATS [24]	0.80	0.15	0.77	0.72	0.73	0.12
Reinforcement Learning Methods	**DQN** [11]	0.93	0.20	0.69	0.71	0.72	0.12
	iRDPG [28]	1.32	0.28	*1.16*	**0.87**	0.85	0.18
Ranking Methods	**LSTM** [7]	0.95	0.22	0.74	0.74	0.79	0.12
	GCN [26]	0.46	0.13	0.73	0.79	0.72	0.16
	RSR-E[13]	1.12	0.26	0.82	0.81	0.88	*0.20*
	RSR-I[13]	1.34	0.39	0.85	*0.86*	0.95	0.21
	STHAN-SR[35]	*1.42**	*0.44**	1.09	*0.86*	*1.12**	*0.33**
	TEDHGN	**1.49**$^{*\diamond}$	**0.51**$^{*\diamond}$	**1.32**$^{*\diamond}$	0.81	**1.53**$^{*\diamond}$	**0.54**$^{*\diamond}$

4.4 Baselines

For an extensive comparison, we choose baselines across four categories based on their training objective, namely regression, classification, reinforcement learning, and ranking.

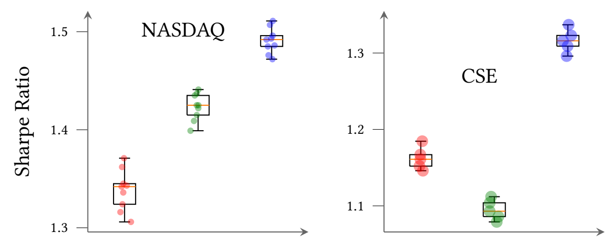

Figure 3: Distribution of Sharpe Ratio of iRDPG (left, red), STHAN-SR (middle, green) and TEDHGN (right, blue) over 5 independent runs.

5 RESULTS AND ANALYSIS

5.1 Performance Comparison with Baselines

We compare TEDHGN with the baselines in terms of returns based metrics in Table 2. Ranking and reinforcement learning methods optimized for higher returns are more profitable than classification and regression methods, which do not necessarily select the most profitable stocks to trade, validating the formulation of stock prediction as a learning to rank problem [35]. Furthermore, methods that encode stock interdependence (RSR-E, RSR-I) outperform price-only methods (LSTM, iRDPG), as they capture the spatial correlations amongst movements of related stocks. Overall, TEDHGN consistently yields significantly ($p < 0.01$) larger risk-adjusted returns than all baselines across all datasets. We attribute this improvement to two aspects: 1) hypergraph learning and 2) Dynamic hypergraph construction. TEDHGN represents higher-order stock relations as a hypergraph instead of constraining them as pairwise edges in ordinary graphs. Furthermore, the hypergraph is updated dynamically using the temporal stock features instead of using static pre-defined hypergraph structure [37]. This enhances the representation encountered by the underlying hypergraph convolution operation, infusing the dynamically evolving relational knowledge at each time step. The performance of TEDHGN puts forward its applicability as an effective stock forecasting model. We now further analyze TEDHGN through an ablation study to probe the contribution of its various components.

5.2 Ablation Study

We now probe TEDHGN's stock ranking ability and profitability benefits from each of its components in Table 3 and with state-of-the-art baselines in Figure 3. First, we observe that formulating stock prediction as a learning to rank problem significantly improves over a classification objective. Next, we note that the temporal attention mechanism significantly ($p < 0.01$) improves gated temporal convolutions, as it can effectively learn crucial long term and short term temporal dependencies. We observe that complementing price only methods with graph based learning leads to significant improvements ($p < 0.01$), reiterating the importance of exploiting spatial correlations amongst movements of related stocks. We observe large gains in using dynamic hypergraphs instead of static domain knowledge-based hypergraphs. This observation suggests that TEDHGN can encode hidden stock correlations that are not based on their industry or corporate relations. Our observation ties up with financial literature [4, 5, 18], which shows that such latent correlations may exist in the stock market for certain time-intervals.

Table 3: Ablation study over TEDHGN and its components (mean of 5 independent runs). $*, \diamond$ indicates the improvement over STHGCN-Ranking and dynamic graph (G) convolution with gated temporal convolution (TConv), respectively is statistically significant ($p < 0.01$) over Wilcoxon's Signed Rank Test. HG and TAttn stand for Hypergraph and Temporal Attention, respectively.

Ablation Study	NASDAQ			CSE			NYSE		
	NDCG	IRR	SR	NDCG	IRR	SR	NDCG	IRR	SR
STHGCN Classification	-	0.27	0.9	-	0.24	0.89	-	0.22	0.81
STHGCN Ranking	0.83	0.31	1.31	0.76	0.29	0.93	0.77	0.37	1.19
TConv+TAttn	0.77	0.3	0.98	0.75	0.16	0.51	0.76	0.28	0.93
HG+TConv+TAttn	0.81	0.38^{*}	1.32^{*}	0.76	0.31^{*}	0.94^{*}	0.8^{*}	0.46^{*}	1.33^{*}
Dynamic G+TConv	0.83	0.34^{*}	1.04	0.85^{*}	0.33^{*}	0.87	0.79^{*}	0.45^{*}	1.39^{*}
Dynamic G+TConv+TAttn	0.81	$0.39^{*\diamond}$	$1.12^{\diamond}$	$0.89^{*\diamond}$	$0.36^{*\diamond}$	$0.90^{\diamond}$	$0.81^{*\diamond}$	$0.46^{*\diamond}$	$1.44^{*\diamond}$
Dynamic HG+TConv	0.83	$0.36^{*\diamond}$	$1.46^{*\diamond}$	$0.90^{*\diamond}$	$0.42^{*\diamond}$	$1.24^{*\diamond}$	$0.83^{*\diamond}$	$0.51^{*\diamond}$	$1.46^{*\diamond}$
TEDHGN	$\mathbf{0.92}^{*\diamond}$	$\mathbf{0.51}^{*\diamond}$	$\mathbf{1.49}^{*\diamond}$	$\mathbf{0.94}^{*\diamond}$	$\mathbf{0.81}^{*\diamond}$	**1.32***	$\mathbf{0.85}^{*\diamond}$	$\mathbf{0.54}^{*\diamond}$	$\mathbf{1.53}^{*\diamond}$

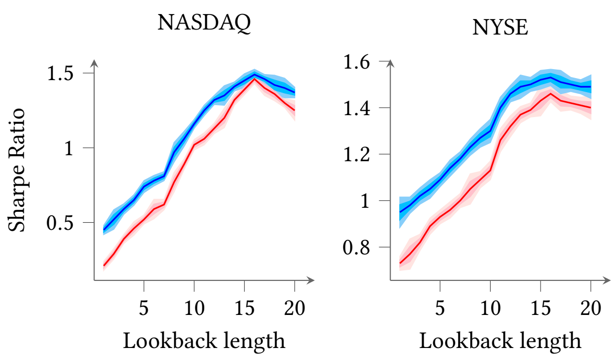

Figure 4: Sensitivity to the length of lookback period size τ. Red denotes Dynamic HG+TConv and blue denotes TEDHGN.

Effectiveness of Hypergraph Learning: To further quantify the improvements from representing stock relation as hypergraphs, we compare it with its ordinary graph-based counterpart in Table 3. Specifically, we decompose the initial hypergraph into an ordinary graph by decomposing each hyperedge of degree n into $\binom{n}{2}$ pairwise edges. Similar to the dynamic hypergraph update rule, the obtained graph is then dynamically updated using k-means and k-NN clustering methods. Specifically, we make pairwise connections between vertices from the nearest $H - 1$ clusters obtained via KMeans clustering. We also construct pairwise connections between $k - 1$ nearest neighbors obtained using KNN clustering. We observe significant improvements ($p < 0.01$) on representing stock relations as hypergraphs instead of ordinary graphs. This improvement empirically validates that hypergraphs effectively capture higher-order relations between stocks, as opposed to simple graphs.

5.3 Impact of Historical Context on Profitability

We probe TEDHGN's performance with different lengths of the lookback period in Figure 4. We note that the temporal attention component in TEDHGN surpasses gated temporal convolution due to its ability to capture more salient features in the lookback period. Next, we observe that using shorter lookback periods leads to poorer performance, potentially because of lesser historical information for a coherent stock ranking. As we increase the size of the lookback period, we note larger periods allow the inclusion of stale market information from older days, which may not contribute to future stock trends [8]. However, we find that TEDHGN using attention-based temporal convolution can selectively filter out crucial information from larger windows to an extent. Overall we observe that TEDHGN is robust over varying window lengths and works best with mid-sized windows.

6 CONCLUSION AND FUTURE WORK

We propose TEDHGN, the first neural architecture for stock ranking that leverages attentive Spatio-temporal hypergraph convolutions to model stock prices' temporal evolution while accounting for dynamic relationships between connected stocks. Our proposed architecture can be generalized for Spatio-temporal feature learning over time-evolving hypergraphs across problems in varying domains such as traffic forecasting and sensor prediction. We empirically demonstrate that TEDHGN outperforms state-of-the-art methods by 24% in terms of risk-adjusted returns across three real world stock markets. In the future, we aim to explore relationships between stocks beyond industrial and corporate relations and account for other data sources and modalities like online financial news, company earning, and social media.

REFERENCES

[1] Klaus Adam, Albert Marcet, and Juan Pablo Nicolini. 2016. Stock Market Volatility and Learning. *The Journal of Finance* 71, 1 (Jan. 2016), 33–82. https://doi.org/10.1111/jofi.12364

[2] Sameer Agarwal, Kristin Branson, and Serge Belongie. 2006. Higher order learning with graphs. In *Proceedings of the 23rd international conference on Machine learning - ICML '06*. ACM Press. https://doi.org/10.1145/1143844.1143847

[3] Leif BG Andersen. 2007. Efficient simulation of the Heston stochastic volatility model. *Available at SSRN 946405* (2007).

[4] Magnus Andersson, Elizaveta Krylova, and Sami Vähämaa. 2008. Why does the correlation between stock and bond returns vary over time? *Applied Financial Economics* 18, 2 (2008), 139–151. https://doi.org/10.1080/09603100601057854 arXiv:https://doi.org/10.1080/09603100601057854

[5] Yuta Arai, Takeo Yoshikawa, and Hiroshi Iyetomi. 2015. Dynamic Stock Correlation Network. *Procedia Computer Science* 60 (2015), 1826–1835. https://doi.org/10.1016/j.procs.2015.08.293

[6] S. Bai, F. Zhang, and P. Torr. 2019. Hypergraph Convolution and Hypergraph Attention. *ArXiv* abs/1901.08150 (2019).

[7] Wei Bao, Jun Yue, and Yulei Rao. 2017. A deep learning framework for financial time series using stacked autoencoders and long-short term memory. *PLOS ONE* 12, 7 (July 2017), e0180944. https://doi.org/10.1371/journal.pone.0180944

[8] Dan Bernhaedt and Jianjun Miao. 2004. Informed Trading When Information Becomes Stale. *The Journal of Finance* 59, 1 (2004), 339–390. http://www.jstor.org/stable/3694898

[9] David C Blitz and Pim Van Vliet. 2007. The volatility effect. *The Journal of Portfolio Management* 34, 1 (2007), 102–113.

[10] Tim Bollerslev. 1986. Generalized autoregressive conditional heteroskedasticity. *Journal of econometrics* 31, 3 (1986), 307–327.

[11] Salvatore Carta, Anselmo Ferreira, Alessandro Sebastian Podda, Diego Reforgiato Recupero, and Antonio Sanna. 2020. Multi-DQN: an Ensemble of Deep Q-Learning Agents for Stock Market Forecasting. *Expert Systems with Applications* (2020), 113820.

[12] Fuli Feng, Huimin Chen, Xiangnan He, Ji Ding, Maosong Sun, and Tat-Seng Chua. 2019. Enhancing stock movement prediction with adversarial training. In *Proceedings of the 28th International Joint Conference on Artificial Intelligence*.

[13] Fuli Feng, Xiangnan He, Xiang Wang, Cheng Luo, Yiqun Liu, and Tat-Seng Chua. 2019. Temporal Relational Ranking for Stock Prediction. *ACM Transactions on Information Systems* 37, 2 (Mar 2019), 1–30.

[14] Yifan Feng, Haoxuan You, Zizhao Zhang, Rongrong Ji, and Yue Gao. 2019. Hypergraph Neural Networks. *Proceedings of the AAAI Conference on Artificial Intelligence* 33 (Jul 2019), 3558–3565. https://doi.org/10.1609/aaai.v33i01.33013558

[15] Xiang Gao, Wei Hu, and Zongming Guo. 2019. Exploring Structure-Adaptive Graph Learning for Robust Semi-Supervised Classification. arXiv:1904.10146 [cs.LG]

[16] E. Han, G. Karypis, V. Kumar, and B. Mobasher. 1997. Clustering Based On Association Rule Hypergraphs. In *DMKD*.

[17] Alexandre Hocquard, Sunny Ng, and Nicolas Papageorgiou. 2013. A constant-volatility framework for managing tail risk. *The Journal of Portfolio Management* (2013).

[18] Changjian Hu, Liqin Xu, Guoyang Shen, and Toshikazu Fukushima. 2009. Temporal Company Relation Mining from the Web. In *Advances in Data and Web Management*. Springer Berlin Heidelberg, 392–403. https://doi.org/10.1007/978-3-642-00672-2_35

[19] Ziniu Hu, Weiqing Liu, Jiang Bian, Xuanzhe Liu, and Tie-Yan Liu. 2018. Listening to chaotic whispers: A deep learning framework for news-oriented stock trend prediction. In *Proceedings of the eleventh ACM international conference on web search and data mining*. 261–269.

[20] Jianwen Jiang, Yuxuan Wei, Yifan Feng, Jingxuan Cao, and Yue Gao. 2019. Dynamic Hypergraph Neural Networks.. In *IJCAI*. 2635–2641.

[21] Jianwen Jiang, Yuxuan Wei, Yifan Feng, Jingxuan Cao, and Yue Gao. 2019. Dynamic Hypergraph Neural Networks. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence*. International Joint Conferences on Artificial Intelligence Organization. https://doi.org/10.24963/ijcai.2019/366

[22] WeiWei Jiang. 2020. Applications of deep learning in stock market prediction: recent progress. *ArXiv* abs/2003.01859 (2020).

[23] Raehyun Kim, Chan Ho So, Minbyul Jeong, Sanghoon Lee, J. Kim, and Jaewoo Kang. 2019. HATS: A Hierarchical Graph Attention Network for Stock Movement Prediction. *ArXiv* abs/1908.07999 (2019).

[24] Raehyun Kim, Chan Ho So, Minbyul Jeong, Sanghoon Lee, Jinkyu Kim, and Jaewoo Kang. 2019. HATS: A Hierarchical Graph Attention Network for Stock Movement Prediction.

[25] Diederik P. Kingma and Jimmy Ba. 2015. Adam: A Method for Stochastic Optimization. *CoRR* abs/1412.6980 (2015).

[26] Thomas Kipf and M. Welling. 2017. Semi-Supervised Classification with Graph Convolutional Networks. *ArXiv* abs/1609.02907 (2017).

[27] Wei Li, Ruihan Bao, Keiko Harimoto, Deli Chen, Jingjing Xu, and Qi Su. 2020. Modeling the Stock Relation with Graph Network for Overnight Stock Movement Prediction. In *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI-20*, Christian Bessiere (Ed.). 4541–4547.

[28] Yang Liu, Qi Liu, Hongke Zhao, Zhen Pan, and Chuanren Liu. 2020. Adaptive Quantitative Trading: An Imitative Deep Reinforcement Learning Approach. *Proceedings of the AAAI Conference on Artificial Intelligence* 34, 02 (April 2020), 2128–2135.

[29] Y. Luo, J. Hu, X. Wei, D. Fang, and H. Shao. 2014. Stock trends prediction based on hypergraph modeling clustering algorithm. In *2014 IEEE International Conference on Progress in Informatics and Computing*. 27–31.

[30] Y. Luo, J. Hu, X. Wei, D. Fang, and H. Shao. 2014. Stock trends prediction based on hypergraph modeling clustering algorithm. In *2014 IEEE International Conference on Progress in Informatics and Computing*.

[31] Daiki Matsunaga, Toyotaro Suzumura, and Toshihiro Takahashi. 2019. Exploring Graph Neural Networks for Stock Market Predictions with Rolling Window Analysis. *ArXiv* abs/1909.10660 (2019).

[32] Thi-Thu Nguyen and Seokhoon Yoon. 2019. A Novel Approach to Short-Term Stock Price Movement Prediction using Transfer Learning. *Applied Sciences* 9, 22 (Nov 2019), 4745. https://doi.org/10.3390/app9224745

[33] Fei Ren and Wei-Xing Zhou. 2014. Dynamic evolution of cross-correlations in the Chinese stock market. *PloS one* 9, 5 (2014), e97711.

[34] Aravind Sankar, Yanhong Wu, Liang Gou, Wei Zhang, and Hao Yang. 2018. Dynamic graph representation learning via self-attention networks. *arXiv preprint arXiv:1812.09430* (2018).

[35] Ramit Sawhney, Shivam Agarwal, Arnav Wadhwa, Tyler Derr, and Rajiv Ratn Shah. 2021. Stock Selection via Spatiotemporal Hypergraph Attention Network: A Learning to Rank Approach.

[36] Ramit Sawhney, Shivam Agarwal, Arnav Wadhwa, and Rajiv Shah. 2021. Exploring the Scale-Free Nature of Stock Markets: Hyperbolic Graph Learning for Algorithmic Trading. In *Proceedings of the Web Conference 2021* (Ljubljana, Slovenia) *(WWW '21)*. Association for Computing Machinery, New York, NY, USA, 11–22. https://doi.org/10.1145/3442381.3450095

[37] Ramit Sawhney, Shivam Agarwal, Arnav Wadhwa, and Rajiv Ratn Shah. 2020. Spatiotemporal hypergraph convolution network for stock forecasting. In *2020 IEEE International Conference on Data Mining (ICDM)*.

[38] J. Shao, J. Zhu, Y. Wei, Y. Feng, and X. Zhao. 2019. Emotion Recognition by Edge-Weighted Hypergraph Neural Network. In *2019 IEEE International Conference on Image Processing (ICIP)*. 2144–2148.

[39] Ze Tian, TaeHyun Hwang, and Rui Kuang. 2009. A hypergraph-based learning algorithm for classifying gene expression and arrayCGH data with prior knowledge. *Bioinformatics* 25, 21 (07 2009), 2831–2838. https://doi.org/10.1093/bioinformatics/btp467 arXiv:https://academic.oup.com/bioinformatics/article-pdf/25/21/2831/16890170/btp467.pdf

[40] Jung-Hua Wang and Jia-Yann Leu. 1996. Stock market trend prediction using ARIMA-based neural networks. In *Proceedings of International Conference on Neural Networks (ICNN'96)*, Vol. 4. IEEE, 2160–2165.

[41] Yumo Xu and Shay B. Cohen. 2018. Stock Movement Prediction from Tweets and Historical Prices. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Association for Computational Linguistics, Melbourne, Australia, 1970–1979.

[42] Naganand Yadati, Madhav Nimishakavi, Prateek Yadav, Vikram Nitin, Anand Louis, and Partha Talukdar. 2019. HyperGCN: A New Method For Training Graph Convolutional Networks on Hypergraphs. In *Advances in Neural Information Processing Systems 32*, H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett (Eds.). Curran Associates, Inc., 1511–1522.

[43] Yang Shen, J. Hu, Yanan Lu, and Xiaofeng Wang. 2012. Stock trends prediction by hypergraph modeling. In *2012 IEEE International Conference on Computer Science and Automation Engineering*. 104–107.

[44] Seongjun Yun, Minbyul Jeong, Raehyun Kim, Jaewoo Kang, and Hyunwoo J Kim. 2019. Graph transformer networks. In *Advances in Neural Information Processing Systems*. 11983–11993.

[45] Liheng Zhang, Charu Aggarwal, and Guo-Jun Qi. 2017. Stock price prediction via discovering multi-frequency trading patterns. In *Proceedings of the 23rd ACM SIGKDD international conference on knowledge discovery and data mining*. 2141–2149.

[46] Muhan Zhang and Yixin Chen. 2018. Link Prediction Based on Graph Neural Networks. In *Advances in Neural Information Processing Systems 31*, S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett (Eds.). Curran Associates, Inc., 5165–5175.

[47] Xin Zhang and J. You. 2020. A Gated Dilated Causal Convolution Based Encoder-Decoder for Network Traffic Forecasting. *IEEE Access* 8 (2020), 6087–6097.

[48] Zizhao Zhang, Haojie Lin, Yue Gao, and KLISS BNRist. 2018. Dynamic Hypergraph Structure Learning.. In *IJCAI*. 3162–3169.

[49] Dengyong Zhou, Jiayuan Huang, and Bernhard Schölkopf. 2007. Learning with hypergraphs: Clustering, classification, and embedding. In *Advances in neural information processing systems*. 1601–1608.

[50] Dengyong Zhou, Jiayuan Huang, and Bernhard Schölkopf. 2007. Learning with Hypergraphs: Clustering, Classification, and Embedding. In *Advances in Neural Information Processing Systems 19*, B. Schölkopf, J. C. Platt, and T. Hoffman (Eds.). MIT Press, 1601–1608.