FactorGCL: A Hypergraph-Based Factor Model with Temporal Residual Contrastive Learning for Stock Returns Prediction

Yitong Duan, Weiran Wang, Jian Li
Tsinghua University, Beijing, China
{dyt19, wang-wr21}@mails.tsinghua.edu.cn, lijian83@mail.tsinghua.edu.cn

Abstract

As a fundamental method in economics and finance, the factor model has been extensively utilized in quantitative investment. In recent years, there has been a paradigm shift from traditional linear models with expert-designed factors to more flexible nonlinear machine learning-based models with data-driven factors, aiming to enhance the effectiveness of these factor models. However, due to the low signal-to-noise ratio in market data, mining effective factors in data-driven models remains challenging. In this work, we propose a hypergraph-based factor model with temporal residual contrastive learning (FactorGCL) that employs a hypergraph structure to better capture high-order nonlinear relationships among stock returns and factors. To mine hidden factors that supplement human-designed prior factors for predicting stock returns, we design a cascading residual hypergraph architecture, in which the hidden factors are extracted from the residual information after removing the influence of prior factors. Additionally, we propose a temporal residual contrastive learning method to guide the extraction of effective and comprehensive hidden factors by contrasting stock-specific residual information over different time periods. Our extensive experiments on real stock market data demonstrate that FactorGCL not only outperforms existing state-of-the-art methods but also mines effective hidden factors for predicting stock returns.

Introduction

In the domain of stock investment, the factor model has long been a cornerstone for explaining and predicting stock returns, which employs specific variables, known as factors, to elucidate fluctuations in stock prices (Daniel, Hirshleifer, and Sun 2020; Fama and French 2021). This method, prevalent in both academia and industry, has demonstrated a strong ability to explain and predict stock returns. Consequently, establishing an effective factor model is of paramount importance in stock investment.

The factor model explains stock returns by utilizing various factors, including fundamental, technical, and macroeconomic indicators. Specifically, in a factor model, stocks are described by factors and their corresponding factor exposures, which represent the impact of factors on stocks. In traditional factor models, these factors are designed based on expert practical experience. For instance, the well-known

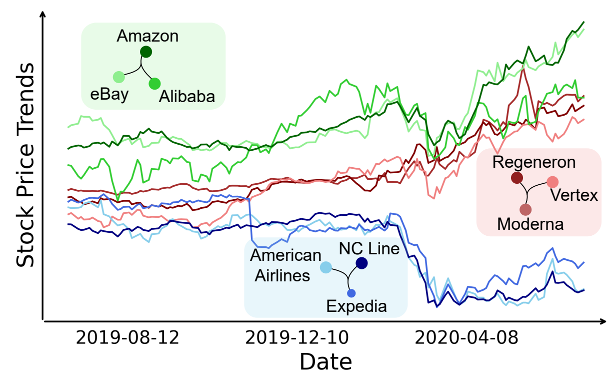

Figure 1: Stock price trends vary across different sectors and industries. During the COVID-19 pandemic, the stock price trends of the electronic consumer and medical industries exhibited high correlations, which are insufficiently explained using human-designed industry factors.

Fama-French model (Eugene and French 1992) employs three manually designed factors: *market*, *size*, and *value*. However, these human-designed factors, while effective, are limited in number and may not sufficiently explain stock returns. For example, as illustrated in Figure 1, stock price trends across different industries exhibited high correlations that could not be adequately explained by industry-specific factors based on prior human experience. Moreover, most existing factor models are linear, explaining stock returns through a linear combination of factors weighted by factor exposures. However, recent studies (Levin 1995; Almeida and Freire 2023; Bansal and Yaron 2004; He and Krishnamurthy 2013) have identified complex nonlinear relationships between factors and stock returns in real markets. This discrepancy highlights the limitations of linear factor models in capturing actual market behavior. Consequently, a core issue in current factor model research is how to mine more effective factors that are applicable to real market behavior.

Recent advancements in machine learning (ML) have introduced a new perspective for factor model research (Kelly, Pruitt, and Su 2019; Uddin and Yu 2020; Gu, Kelly, and Xiu 2021). ML-based approaches can learn complex and nonlinear market patterns in a data-driven manner and construct

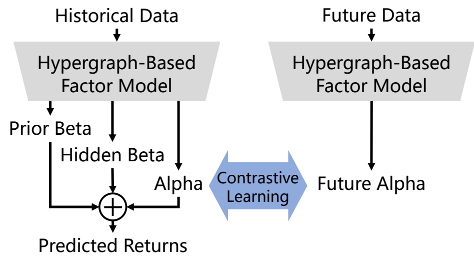

Figure 2: Brief illustration of FactorGCL.

more market-adaptive models (Li et al. 2024; Zou et al. 2022; Cui et al. 2023). However, the low signal-to-noise ratio in stock market data may complicate the learning process for ML-based factor models. Most current ML-based methods extract factors from market data without effectively leveraging prior human experience, which may lead to overfitting of the extracted factors to market noise rather than capturing effective patterns. This limitation presents a significant obstacle to the application of ML in factor models.

To address this obstacle, we propose a hypergraph-based factor model with temporal residual contrastive learning (FactorGCL) that supplements human-designed prior factors with data-driven hidden factors, thereby enhancing the effectiveness of factor models in predicting stock returns. Our model utilizes a hypergraph, a generalized graph structure, to better capture high-order relationships among stock returns and factors. Specifically, stocks are treated as nodes in the hypergraph, factors are represented as hyperedges, and the mining of hidden factors is framed as a hyperedge generation task. As shown in Figure 2, FactorGCL employs a cascading residual hypergraph architecture where stock returns are decomposed into three components: prior beta, hidden beta, and individual alpha. Each component is extracted from the residuals after removing the influence of the previous component. Additionally, we propose a temporal residual contrastive learning method to guide the model's learning process by contrasting individual stock residuals across different time periods.

In summary, the contributions of our work are as follows:

- We propose FactorGCL, a novel factor model that utilizes a hypergraph structure to capture high-order nonlinear relationships among stock returns and factors. It employs a cascading residual hypergraph architecture to mine hidden factors, supplementing human-designed prior factors, thereby enhancing the prediction of stock returns.
- We design a self-supervised learning method called temporal residual contrastive learning. This method enhances the model's ability to extract effective and comprehensive hidden factors by contrasting stock-specific residuals across different time periods to better guide the mining of hidden factors.
- We conduct extensive experiments on real stock market data. The results demonstrate that our method not only surpasses existing state-of-the-art baselines in stock trend prediction but also mines effective hidden factors for predicting stock returns.

Related Work

Factor Model

Factor models are widely utilized in stock investments. The original factor model, the capital asset pricing model (CAPM) (Treynor 1961; Sharpe 1964; Lintner 1975), attributes differences in stock returns to varying exposures to a single market factor. Later, in a seminal work (Eugene and French 1992), it was observed that firm value and size also contribute to explaining expected stock returns, and proposed the Fama-French three-factor model. With the advancement of machine learning, some machine learning-based factor models have emerged. (Levin 1995) proposed a nonlinear factor model based on neural networks to model possible interactions between different factors. (Gu, Kelly, and Xiu 2021) proposed a latent dynamic factor model using a conditional autoencoder network to capture non-linearity in return dynamics, demonstrating that the nonlinear factor model outperforms other leading linear methods. Furthermore, (Duan et al. 2022) introduced a probabilistic factor model based on variational autoencoders to better extract effective factors from the market data with high noise levels.

Hypergraph Neural Network

Hypergraphs have proven to be an efficient approach for modeling high-order correlations among data. (Zhou, Huang, and Schölkopf 2006) first introduced hypergraph learning as a propagation process on hypergraph structures. (Feng et al. 2019) further advanced this concept by developing the hypergraph convolutional neural network using deep learning methods for data representation learning. Hypergraphs have also been widely applied in the field of stock return prediction(Li et al. 2022; Han et al. 2023; Su et al. 2024), (Sawhney et al. 2020) initially applied the hypergraph neural network to learn stock price evolution based on stock relationships. Subsequently, (Sawhney et al. 2021) improved hypergraph neural network for stock trend prediction by incorporating ranking loss. (Xu et al. 2021) designed a concept-oriented graph framework to mine hidden concepts for stock trend forecasting. Additionally, (Xia et al. 2024) developed a dynamic hypergraph for stock selection problem using a transformer-based pretraining mechanism.

Contrastive Learning

This work is also related to contrastive learning , a promising class of self-supervised methods that leverage the semantic dependencies of sample pairs to capture the essence of data (Chen and He 2021; Lin et al. 2022; Li et al. 2020; Caron et al. 2020). Contrastive learning has been widely used in various applications. For instance, (Oord, Li, and Vinyals 2018) introduced Contrastive Predictive Coding to learn useful representations for predicting future data. (Liu et al. 2022) proposed a unsupervised deep graph structure learning method based on contrastive learning. In the financial domain, (Hou et al. 2021) introduced a contrastive learning method for multi-granularity stock data and used it as a regularization term to improve stock trend prediction tasks.

Preliminaries

In this section, we first introduce the basic concepts of hypergraph convolutional neural networks, and then formally describe the research problem.

Hypergraph Convolutional Neural Network

A hypergraph generalizes a graph by allowing an edge, termed a hyperedge, to connect two or more nodes. This structure enables the hypergraph to capture the group-wise correlations beyond pair-wise connections. Formally, a hypergraph is defined as $\mathcal{G} = (\mathcal{V}, \mathcal{E}, W)$, where V and E denote the sets of vertices and hyperedges, respectively, and W is a diagonal matrix assigning weights to the hyperedges. The pair $(\mathcal{V}, \mathcal{E})$ in a hypergraph can be represented by an incidence matrix $H \in \mathbb{R}^{|\mathcal{V}| \times |\mathcal{E}|}$, where $H^{(i,j)}$ indicates the connection between the i-th vertex $\mathcal{V}^{(i)}$ and the j-th hyperedge $\mathcal{E}^{(j)}$, defined as: $H^{(i,j)} = \begin{cases} 1, & \text{if } \mathcal{V}^{(i)} \in \mathcal{E}^{(j)} \\ 0, & \text{if } \mathcal{V}^{(i)} \notin \mathcal{E}^{(j)} \end{cases}$

The hypergraph convolutional neural network (HyperGCN) (Feng et al. 2019) extends graph convolutional networks to hypergraphs, enabling the capture of high-order relationships inherent in hypergraph structures. The core of HyperGCN involves a message propagation rule that updates node features by aggregating information from their connected hyperedges, which are influenced by all nodes connected by those hyperedges. Formally, the node features at the $(l+1)$-th layer of HyperGCN, $e^{(l+1)} \in \mathbb{R}^{|\mathcal{V}| \times d_{l+1}}$, are computed using the formula:

$$e^{(l+1)} = \sigma(D_n^{-1/2} H W D_e^{-1} H^T D_n^{-1/2} e^{(l)} w) \tag{1}$$

where σ is a non-linear activation function, $D_n \in \mathbb{R}^{|\mathcal{V}| \times |\mathcal{V}|}$ and $D_e \in \mathbb{R}^{|\mathcal{E}| \times |\mathcal{E}|}$ are diagonal matrices representing node degrees and hyperedge degrees, respectively. $W \in \mathbb{R}^{|\mathcal{E}| \times |\mathcal{E}|}$ is a diagonal matrix representing hyperedge weights, $w \in \mathbb{R}^{d_l \times d_{l+1}}$ is a learnable weight matrix, with d_l representing the dimensions of node features at the l-th layer.

Problem Formulation

Given N stocks in cross-section of the stock market, and a set of K factors, the traditional linear factor model calculates expected stock returns as a linear combination of factors weighted by factor exposures:

$$y_t = \sum_{k=1}^{K} \beta_t^{(k)} z_t^{(k)} + \alpha_t \tag{2}$$

where $y_t = \frac{\text{price}_{t+\Delta t} - \text{price}_t}{\text{price}_t} \in \mathbb{R}^N$ denotes the future returns of N stocks at trading day t, $\beta_t \in \mathbb{R}^{N \times K}$ is the factor exposure matrix of stocks at trading day t, $\beta_t^{(k)} \in \mathbb{R}^N$ represents the k-th factor exposure of stocks at trading day t, $z_t \in \mathbb{R}^K$ is the vector of K factor returns, and $\alpha_t \in \mathbb{R}^N$ denotes the idiosyncratic returns of the stocks.

(Levin 1995) extends the linear factor model to a nonlinear version, where the stock returns are calculated using a nonlinear function h of factor returns and factor exposures:

$$y_t = h(\beta_t, z_t) + \alpha_t \tag{3}$$

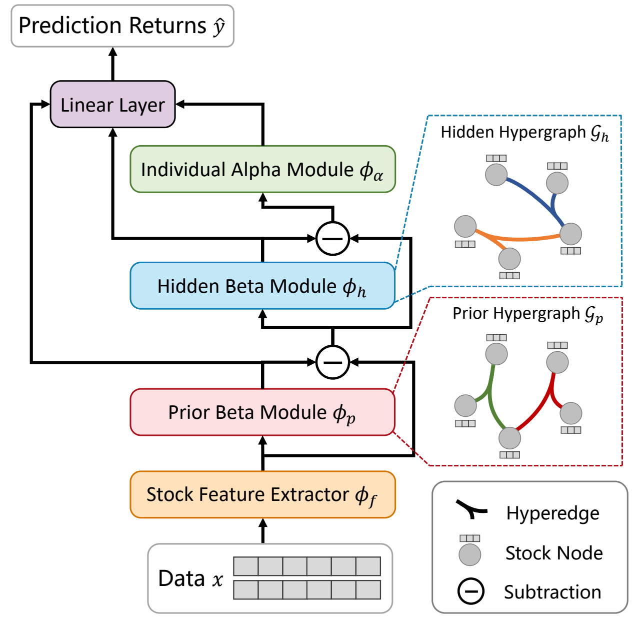

Figure 3: Overview of the cascading residual hypergraph architecture in FactorGCL. Stock returns are decomposed into prior beta, hidden beta, and individual alpha components. Each component is extracted from the residuals after removing the influence of the preceding component.

Typically, factor returns and stock individual returns are estimated using factor exposures and historical stock returns. For example, in a linear factor model, factor returns z are estimated using the slopes from the linear regression on historical data, while individual returns α are estimated using the residuals. Therefore, we set $z_t = f_z(\beta_t, x_t)$ and $\alpha_t = f_\alpha(\beta_t, x_t)$, where $x_t \in \mathbb{R}^{N \times T \times D}$ represents the historical data of stocks at trading day t, with T being the length of historical data and D being the feature dimension of each stock's data.

Finally, the task in this work is to learn a nonlinear factor model based on given factor exposures β_t and historical stock market data x_t, for predicting future stock returns.

$$\hat{y}_t = h(\beta_t, z_t) + \alpha_t = f_\beta(\beta_t, x_t) + f_\alpha(\beta_t, x_t) \tag{4}$$

In the following sections, we simplify our notation by omitting the time subscript t. Unless otherwise specified, all references to w and b pertain to the weights and biases of linear layers, respectively, and will not be further elaborated.

Methodology

This section introduces the design of FactorGCL. We first design a cascading residual hypergraph architecture for our model, which extracts prior beta, hidden beta, and individual alpha components to predict stock returns. Next, we propose a temporal residual contrastive learning method to guide the model in extracting effective and comprehensive hidden factors.

Cascading Residual Hypergraph Architecture

As previously mentioned, we utilize hypergraphs to construct a nonlinear factor model. Inspired by (Xu et al. 2021), we design a cascading residual hypergraph architecture to better extract hidden factors. As illustrated in Figure 3, we decompose the predicted stock returns into three components: prior beta, hidden beta, and individual alpha. Specifically, we first extract stock features from the raw data and then use the prior beta module to extract the representations of prior factors. Next, we mine hidden factors from the residuals after removing the prior beta information using the hidden beta module. Finally, we extract individual alpha from the residuals after removing both prior and hidden beta information. The final prediction of our model is obtained by summing these three components. The detailed design of our architecture is as follows.

Feature Extractor Given the raw sequential market data $x \in \mathbb{R}^{N \times T \times D}$, the feature extractor ϕ_{feat} encodes the stock temporal feature $e_s \in \mathbb{R}^{N \times H}$ to capture rich temporal information. This process is defined as $e_s = \phi_{\text{feat}}(x)$, where H represents the dimension of the feature embeddings. To capture long-term dependencies in sequential data, we utilize a gated recurrent unit with a batch normalization as the feature extractor, using the hidden state at the last time step as the stock feature embeddings.

Prior Beta Module To leverage expert knowledge from given K prior factors $\beta \in \mathbb{R}^{N \times K}$, we employ a hypergraph convolutional neural network (HyperGCN) to model the nonlinear relationships among stock returns and these factors. Specifically, we represent stocks as nodes in the hypergraph and factors as hyperedges. Stocks exposed to the same factor are connected by the same hyperedge, with the incidence matrix representing factor exposures. We posit that the information propagation mechanism in the HyperGCN effectively captures the nonlinear influence of factors on stocks. This process, illustrated in Figure 4, comprises the following steps:

- **Message extraction**: Applies a transformation matrix to the each node features to extract expressive information.
- **Message aggregation**: Aggregates the information from stock nodes connected by the same hyperedge, representing the shared information of the corresponding factor.
- **Message sharing**: Integrates the node embeddings with the shared factor information as the influence of factors on stocks.

Formally, given the stock feature embeddings e_s output by the feature extractor, we build a hypergraph $\mathcal{G}_p$ with node features e_s and incidence matrix β. We then calculate the prior beta embeddings by applying the HyperGCN to the hypergraph $\mathcal{G}_p$, expressed as:

$$\begin{aligned} e_p &= \phi_{\text{prior}}(e_s, \beta) \\ &= \sigma(D_n^{-1/2} \beta W D_e^{-1} \beta^T D_n^{-1/2} e_s w_p) \end{aligned} \tag{5}$$

where σ is the LeakyReLU activation function, D_n and D_e are the degree matrices of the nodes and hyperedges, respectively, and $W = I$ is the identity matrix. The resulting prior beta embeddings $e_p \in \mathbb{R}^{N \times H}$ represent the influence of prior factors on stocks.

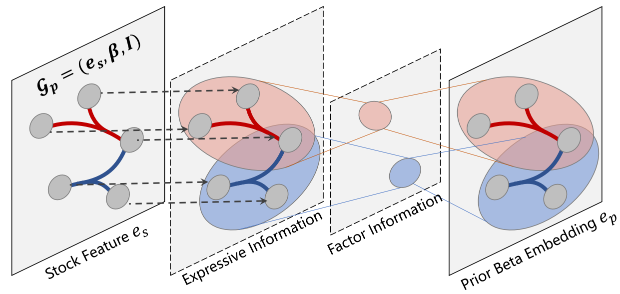

Figure 4: Illustration of the information propagation process in the HyperGCN. The HyperGCN can model the nonlinear influence of factors on stocks by aggregating information from stock nodes connected by the same hyperedge.

Hidden Beta Module As previously mentioned, factors based on human prior knowledge may not adequately capture stock returns. To address this, we designed a hidden beta module to extract hidden factors that supplement these prior factors. Specifically, we regard the extraction of hidden factors as a hyperedge generation task: after removing the prior factor information, the hidden beta module generates new hyperedges from the residual embeddings and constructs a new hypergraph to model the nonlinear influence of these hidden factors on stocks.

Formally, we first calculate the residual embeddings by subtracting the prior factor embeddings from the stock feature embeddings, i.e., $e_r = e_s - e_p$. Next, we construct M learnable vectors $\{c^{(i)}\}_{i=1}^{M}$, where $c^{(i)} \in \mathbb{R}^H$, referred to as hidden factor prototypes. Hidden factors are mined by calculating the similarity between the residual embeddings and the hidden factor prototypes:$\beta_h^{(i,j)} = \text{Sigmoid}(e_r^{(i)} \cdot {c^{(j)}}^T)$ where $\beta_h \in \mathbb{R}^{N \times M}$ is the hidden factor exposure matrix.

Similar to the prior beta module, we construct a hypergraph $\mathcal{G}_h$ with node features e_r and incidence matrix β_h, and then extract the influence of hidden factors on stocks by applying the HyperGCN to the hypergraph $\mathcal{G}_h$:

$$e_h = \phi_{\text{hidden}}(e_r, \beta_h) = \text{HyperGCN}(e_r, \beta_h) \tag{6}$$

where $e_h \in \mathbb{R}^{N \times H}$ is the hidden beta embeddings. Note that we generate "soft" hyperedges β_h in the hidden beta module, with values ranging between [0, 1], enhancing the flexibility of the hidden factors.

Individual Alpha Module In addition to the influence of prior and hidden factors, the idiosyncratic information of the stock itself, or alpha, also significantly impacts stock returns. The individual alpha module handles the residual embeddings after removing the prior and hidden factor embeddings to capture the stock-specific information. We calculate the individual alpha embeddings $e_\alpha \in \mathbb{R}^{N \times H}$ by applying a linear layer with a LeakyReLU activation function:

$$e_\alpha = \text{LeakyReLU}(w_\alpha(e_s - e_p - e_h) + b_\alpha) \tag{7}$$

Prediction We obtain the model's prediction by performing a linear mapping on the embeddings output by the prior beta module, hidden beta module, and individual alpha module, and then summing them up. Additionally, we design a multi-label prediction that requires the model to predict stock returns over multiple forward periods. This approach aims to enhance the robustness and reliability of our model by ensuring its predictive power extends across different future time frames.

$$\hat{y}^{(l)} = w_{o1}^{(l)} e_p + w_{o2}^{(l)} e_h + w_{o3}^{(l)} e_\alpha + b_{\text{o}}^{(l)} \tag{8}$$

where $\hat{y}^{(l)} \in \mathbb{R}^N$ represents the predicted stock returns of the l-th forward prediction period.

Temporal Residual Contrastive Learning

As previously mentioned, data-driven factor models face the challenge of a low signal-to-noise ratio in market data. Specifically, the hidden factors mined through such models encounter two main issues:

- **Effectiveness**: The hidden factors extracted from historical data should remain consistently effective in the future. However, factors extracted by data-driven approaches are prone to overfitting market noise, thus lacking effectiveness in predicting future stock returns.
- **Comprehensiveness**: The hidden factors should supplement prior factors to provide a comprehensive description of stock returns. Nonetheless, market noise complicates factor mining, making the model more likely to extract simplistic factors while neglecting others, thereby failing to adequately represent stock returns.

To address these issues, we have developed a self-supervised contrastive learning method for FactorGCL, termed temporal residual contrastive learning. The motivation behind this method is as follows: for a factor model with effective and comprehensive factors, after removing all factor information, the residual, represented by the alpha embeddings e_α, should contain only idiosyncratic information unique to each stock, independent of other stocks. Based on this intuition, we draw inspiration from (Oord, Li, and Vinyals 2018) and design a cross-temporal contrastive learning approach at the stock node level. As illustrated in the Figure 5, given the same prior and hidden factors, we use our model to calculate alpha embeddings based on both past and future market data. We then treat the past and future alpha embeddings of the same stock as positive pairs, and the embeddings of different stocks as negative pairs. By training the model with a cascading residual hypergraph architecture to extract temporally consistent alpha embeddings through a contrastive learning objective function, our model can be guided to mine hidden factors that are both effective and comprehensive.

Formally, given future data $x' \in \mathbb{R}^{N \times T' \times D}$, with T' being the length of future data, we use the prior factor exposure β and hidden factor exposure β_h extracted from historical data to calculate the future alpha embedding e'_α:

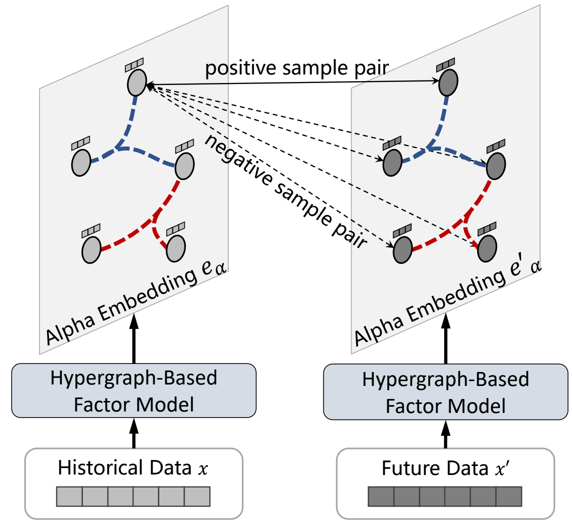

Figure 5: Illustration of the temporal residual contrastive learning method. The model contrasts the past and future alpha embeddings of the same stock as positive pairs, and the embeddings of different stocks as negative pairs.

$$e'_s = \phi'_{\text{feat}}(x') \tag{9}$$

$$e'_\alpha = e'_s - \phi'_{\text{prior}}(e'_s, \beta) - \phi'_{\text{hidden}}(e'_r, \beta_h) \tag{10}$$

where ϕ'_{feat}, ϕ'_{prior}, and ϕ'_{hidden} represent the feature extractor, prior beta module, and hidden beta module for the future data, respectively. Note that ϕ'_{prior} and ϕ'_{hidden} share parameters with ϕ_{prior} and ϕ_{hidden}, respectively.

In this context, we employ the InfoNCE loss function from (Oord, Li, and Vinyals 2018) as the contrastive learning loss function, formulated as follows:

$$\mathcal{L}_{\text{CL}} = -\frac{1}{N}\sum_{i=1}^{N} \log \frac{\exp\left(\text{sim}(p(e_\alpha^{(i)}), p({e'}_\alpha^{(i)}))/\tau\right)}{\sum_{j=1}^{N} \exp\left(\text{sim}(p(e_\alpha^{(i)}), p({e'}_\alpha^{(j)}))/\tau\right)} \tag{11}$$

where $p(x)$ = is a 2-layer MLP with LeakyReLU activation functions, and sim(x, y) represents the cosine similarity function, τ is the temperature parameter.

Objective Function Our objective function consists of two parts. The first part is the mean squared error (MSE) over multiple forward periods, which aims to minimize the prediction error. The second part is the contrastive learning loss, which guides the model to mine hidden factors that are both effective and comprehensive. The overall objective function is:

$$\mathcal{L}_{\text{mse}} = \frac{1}{N \cdot L}\sum_{l=1}^{L}\sum_{i=1}^{N}(\hat{y}^{(i,l)} - y^{(i,l)})^2 \tag{12}$$

$$\mathcal{L} = \mathcal{L}_{\text{mse}} + \gamma \mathcal{L}_{\text{CL}} \tag{13}$$

where L is the number of forward prediction periods, $y^{(i,l)}$ is the true return of the i-th stock at the l-th forward period, and γ is a hyperparameter that balances the contributions of the mean squared error loss and the contrastive learning loss.

Methods	Description	$\Delta t = 1$		$\Delta t = 5$		$\Delta t = 10$		$\Delta t = 20$	
		IC	ICIR	IC	ICIR	IC	ICIR	IC	ICIR
MLP	Multi-layer perceptron model	0.0579	0.6032	0.0650	0.7043	0.0674	0.7155	0.0716	0.8029
GRU (Cho et al. 2014)	RNN model based on GRU	0.0637	0.7045	0.0813	0.9277	0.0829	0.9378	0.0861	1.0083
TCN (Bai, Kolter, and Koltun 2018)	Temporal convolutional network	0.0596	0.6271	0.0719	0.8018	0.0704	0.7880	0.0688	0.7986
Transformer (Vaswani 2017)	Time-series Transformer	0.0617	0.6764	0.0748	0.8292	0.0739	0.8523	0.0723	0.9102
ALSTM (Qin et al. 2017)	Attention-based LSTM	0.0646	0.7320	0.0813	0.9597	0.0818	0.9687	0.0815	1.0091
SFM (Zhang, Aggarwal, and Qi 2017)	Discrete Fourier transform	0.0621	0.6758	0.0789	0.8425	0.0821	0.8624	0.0853	0.9354
GAT (Veličković et al. 2017)	Graph attention network	0.0538	0.5197	0.0667	0.6743	0.0678	0.7093	0.0669	0.7650
HIST (Xu et al. 2021)	Mining hidden concepts	0.0571	0.6059	0.0705	0.7954	0.0703	0.8036	0.0622	0.7656
HyperGCN (Bai, Zhang, and Torr 2021)	Hypergraph convolutional	0.0554	0.5899	0.0665	0.7359	0.0688	0.7502	0.0708	0.7552
STHAN-SR (Sawhney et al. 2021)	Spatio-temporal hypergraph	0.0597	0.6500	0.0765	0.9162	0.0791	0.9724	0.0812	1.0201
FactorVAE (Duan et al. 2022)	A factor model based on VAE	0.0646	0.6699	0.0807	0.8680	0.0756	0.7496	0.0722	0.6526
CI-STHPAN (Xia et al. 2024)	Spatial-temporal pre-training	0.0554	0.5460	0.0707	0.7570	0.0727	0.7703	0.0711	0.7123
FactorGCL	Our proposed model	**0.0684**	**0.7487**	**0.0885**	**0.9787**	**0.0915**	**1.0327**	**0.0929**	**1.0350**

Table 1: The stock returns prediction performance of all compared methods on the test dataset; the higher, the better.

Experiments

In this section, we present a series of experiments to demonstrate the effectiveness of our proposed method in real-world stock markets. Our discussion is structured around the following key research questions:

- **RQ1:** How does our method compare to existing stock trend prediction methods in terms of performance?
- **RQ2:** What impact does each module in our model have on its overall performance?
- **RQ3:** How does varying the number of hidden factors affect the model's performance?
- **RQ4:** Can our method achieve higher investment profits in simulated investment scenarios?

Experiment Settings

We conduct our experiments on the China A-shares market, utilizing a dataset spanning from 01/01/2014 to 06/30/2023. This dataset includes 5028 stocks, excluding suspended or otherwise abnormal stocks, and is constructed from a sequence of market data comprising day-level price-volume data (*high*, *open*, *low*, *close*, *volume-weighted average price*, and *trading volume*). In detail, the cross-sectional standardizated future stock returns with multiple periods ($\Delta t = 1, 5, 10, 20$) are used as labels, and the future return is calculated by the formula $y_t = \frac{\text{price}_{t+\Delta t+1} - \text{price}_{t+1}}{\text{price}_{t+1}}$, where price_t is the volume-weighted average price at trading day t. The length of historical squence data x is $T = 60$, and the length of future data x' is $T' = 20$. We adopt secondary industry factors as the prior factors (83 industries, if a stock belongs to the industry, the value of corresponding factor exposure is 1, otherwise 0).

We follow the temporal order to split the dataset into training set, validation set and test set, where the time length is *5 years* : *1 year* : *2 years*, and adopt a rolling method for training and testing. The overall test period is from 01/01/2020 to 06/30/2023. The other details about the experiment are provided in the supplementary material[1].

[1]https://tinyurl.com/bdhffkch

Main Results

In the experiment, we compare our proposed model with competitive baselines on the stock trend prediction task. In order to evaluate the performance of the compared methods, we adopt the information coefficient (IC) as metric, which are the widely-used evaluation metrics in finance. Besides, we also report the information ratio of information coefficient (ICIR) to evaluate the stability of prediction.

Table 1 summarizes the performances of all compared methods on the test dataset. Our method achieves the highest IC and ICIR among all the compared methods. From the experimental results, we have the following observations:

- FactorGCL can achieve better results than existing stock trend prediction methods like ALSTM, SFM and STHAN-SR, which illustrates the effectiveness of the proposed method for stock trend prediction.
- Moreover, compared with some baselines which can also extract the hidden relations from market data, like HIST, FactorVAE and CI-STHPAN, our model can mine the hidden factors more effectively and comprehensively, to achieve higher prediction performance.

Ablation Study

To verify the effectiveness of different modules in our framework, we build four variants of proposed model by removing the prior beta module, hidden beta module, individual alpha module, and contrastive learning loss, respectively. Table 2 lists the IC of these variants on the test dataset. The results show that each module in our model contributes to the overall performance of our model.

	$IC_{\Delta t=1}$	$IC_{\Delta t=5}$	$IC_{\Delta t=10}$	$IC_{\Delta t=20}$
-wo Prior	0.0654	0.0842	0.0904	0.0906
-wo Hidden	0.0649	0.0819	0.0828	0.0851
-wo Alpha&CL	0.0618	0.0808	0.0857	0.0856
-wo CL	0.0661	0.0856	0.0899	0.0912
FactorGCL	**0.0684**	**0.0885**	**0.0915**	**0.0929**

Table 2: The ablation study results on the test dataset

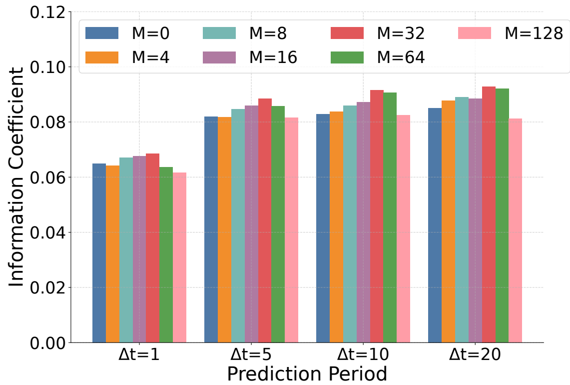

Figure 6: The performances of FactorGCL with different numbers of hidden factors.

We also conducted an experiment to assess the impact of varying the number of hidden factors M on the model's performance. In this experiment, we adjusted the number of hidden factors produced by the hidden beta module while keeping all other components of the model constant. The results, shown in Figure 6, demonstrate an improvement in model performance as the number of hidden factors increases, indicating that hidden factors play a crucial role in enhancing the model's predictive power. However, beyond a certain point, further increases in hidden factors lead to a decline in performance, suggesting that an excessive number of hidden factors can cause overfitting and degrade the model's efficacy, which is consistent with our intuition.

Investment Simulation

To further evaluate the profitability of our method in the real stock market, we conduct an investment simulation. Specifically, we adopt a simple stock selection strategy, referred to as the *TopK* strategy. This strategy involves investing in the *TopK* stocks with the highest predicted scores each trading day and selling them after holding for Δt days, where Δt represents the model's prediction period. In the simulation backtest, we select stocks from the CSI 300 and CSI 500 indexes, which are representative of large-cap and mid to small-cap stocks, respectively, providing a balanced and comprehensive evaluation of our investment strategy across different market segments. We use the equally weighted CSI300 and CSI500 portfolio as the benchmark, and set $TopK = 30$, $\Delta t = 10$, and the transaction cost to 0.3%.

Methods	CSI300			CSI500		
	AR	IR	RoMaD	AR	IR	RoMaD
MLP	0.034	0.957	0.180	-0.001	-0.015	-0.003
GRU	0.083	2.281	1.071	0.043	1.137	0.401
TCN	0.042	1.121	0.418	0.010	0.261	0.067
Transformer	0.058	1.675	0.742	0.040	1.082	0.396
ALSTM	0.078	2.248	0.891	0.046	1.323	0.413
SFM	0.095	2.913	1.122	0.025	0.743	0.185
GAT	0.004	0.100	0.037	-0.020	-0.515	-0.117
HIST	0.048	1.369	0.592	-0.022	-0.620	-0.172
HyperGCN	0.015	0.454	0.107	0.007	0.200	0.069
STHAN-SR	0.047	1.376	0.544	0.033	0.906	0.165
FactorVAE	0.092	2.446	0.740	0.050	1.271	0.411
CI-STHPAN	0.044	1.271	0.251	0.028	0.797	0.252
FactorGCL	**0.169**	**4.208**	**2.978**	**0.087**	**2.092**	**0.737**

Table 3: The investment simulation results on CSI300 and CSI500 stocks; the higher, the better.

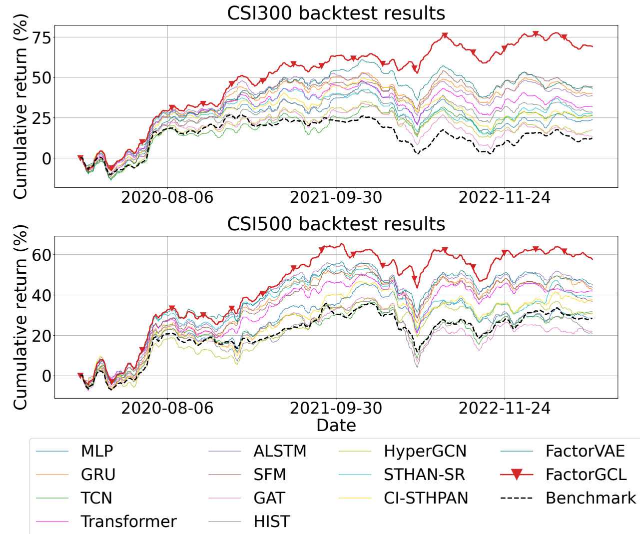

Figure 7: The results of investment simulation.

We present the cumulative return (CR) curves of all compared methods in Figure 7 and report the annualized return (AR), information ratio (IR), and return over maximum drawdown (RoMaD) of cumulative excess return (CER) relative to the benchmark in Table 3, and the meaning of these metrics can be found in the supplementary material. The investment simulation results show that our method achieves the best performance across all metrics, indicating that our model can achieve profitable investments in the real market.

Conclusion

In this paper, we propose a novel hypergraph-based factor model with temporal residual contrastive learning (FactorGCL), which leverages both human-designed orior factors and data-driven hidden factors to predict stock returns. Specifically, our model follows a cascading residual hypergraph architecture, in which the hidden factors are extracted from the residual information after removing the prior factor information. To enhance the effectiveness and comprehensiveness of the hidden factors, we design a temporal residual contrastive learning method that contrasts stock-specific residuals embeddings over different time periods. Our extensive experiments demonstrate that our proposed FactorGCL outperforms existing state-of-the-art baselines in terms of predictive accuracy and investment profitability, and the ablation study further verifies our method can effectively extract hidden factors to improve the model's performance. In the future, we plan to apply the hypergraph-based factor model to risk factor mining and portfolio optimization.

Acknowledgements

The authors are supported in part by the National Natural Science Foundation of China Grant 62161146004

References

Almeida, C.; and Freire, G. 2023. Which (Nonlinear) Factor Models? *Available at SSRN 4421179*.

Bai, S.; Kolter, J. Z.; and Koltun, V. 2018. An empirical evaluation of generic convolutional and recurrent networks for sequence modeling. arXiv. *arXiv preprint arXiv:1803.01271*, 10.

Bai, S.; Zhang, F.; and Torr, P. H. 2021. Hypergraph convolution and hypergraph attention. *Pattern Recognition*, 110: 107637.

Bansal, R.; and Yaron, A. 2004. Risks for the long run: A potential resolution of asset pricing puzzles. *The journal of Finance*, 59(4): 1481–1509.

Caron, M.; Misra, I.; Mairal, J.; Goyal, P.; Bojanowski, P.; and Joulin, A. 2020. Unsupervised learning of visual features by contrasting cluster assignments. *Advances in neural information processing systems*, 33: 9912–9924.

Chen, X.; and He, K. 2021. Exploring simple siamese representation learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 15750–15758.

Cho, K.; Van Merriënboer, B.; Gulcehre, C.; Bahdanau, D.; Bougares, F.; Schwenk, H.; and Bengio, Y. 2014. Learning phrase representations using RNN encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*.

Cui, C.; Li, X.; Zhang, C.; Guan, W.; and Wang, M. 2023. Temporal-relational hypergraph tri-attention networks for stock trend prediction. *Pattern Recognition*, 143: 109759.

Daniel, K.; Hirshleifer, D.; and Sun, L. 2020. Short-and long-horizon behavioral factors. *The review of financial studies*, 33(4): 1673–1736.

Duan, Y.; Wang, L.; Zhang, Q.; and Li, J. 2022. Factorvae: A probabilistic dynamic factor model based on variational autoencoder for predicting cross-sectional stock returns. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, 4468–4476.

Eugene, F.; and French, K. 1992. The cross-section of expected stock returns. *Journal of Finance*, 47(2): 427–465.

Fama, E. F.; and French, K. R. 2021. *Multifactor explanations of asset pricing anomalies*. University of Chicago Press.

Feng, Y.; You, H.; Zhang, Z.; Ji, R.; and Gao, Y. 2019. Hypergraph neural networks. In *Proceedings of the AAAI conference on artificial intelligence*, volume 33, 3558–3565.

Gu, S.; Kelly, B.; and Xiu, D. 2021. Autoencoder asset pricing models. *Journal of Econometrics*, 222(1): 429–450.

Han, H.; Xie, L.; Chen, S.; and Xu, H. 2023. Stock trend prediction based on industry relationships driven hypergraph attention networks. *Applied Intelligence*, 53(23): 29448–29464.

He, Z.; and Krishnamurthy, A. 2013. Intermediary asset pricing. *American Economic Review*, 103(2): 732–70.

Hou, M.; Xu, C.; Liu, Y.; Liu, W.; Bian, J.; Wu, L.; Li, Z.; Chen, E.; and Liu, T.-Y. 2021. Stock trend prediction with multi-granularity data: A contrastive learning approach with adaptive fusion. In *Proceedings of the 30th ACM International Conference on Information & Knowledge Management*, 700–709.

Kelly, B. T.; Pruitt, S.; and Su, Y. 2019. Characteristics are covariances: A unified model of risk and return. *Journal of Financial Economics*, 134(3): 501–524.

Levin, A. 1995. Stock selection via nonlinear multi-factor models. *Advances in Neural Information Processing Systems*, 8.

Li, J.; Zhou, P.; Xiong, C.; and Hoi, S. C. 2020. Prototypical contrastive learning of unsupervised representations. *arXiv preprint arXiv:2005.04966*.

Li, X.; Cui, C.; Cao, D.; Du, J.; and Zhang, C. 2022. Hypergraph-based reinforcement learning for stock portfolio selection. In *ICASSP 2022-2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 4028–4032. IEEE.

Li, X.; Li, Z.; Shi, C.; Xu, Y.; Du, Q.; Tan, M.; Huang, J.; and Lin, W. 2024. AlphaFin: Benchmarking Financial Analysis with Retrieval-Augmented Stock-Chain Framework. *arXiv preprint arXiv:2403.12582*.

Lin, S.; Liu, C.; Zhou, P.; Hu, Z.-Y.; Wang, S.; Zhao, R.; Zheng, Y.; Lin, L.; Xing, E.; and Liang, X. 2022. Prototypical graph contrastive learning. *IEEE transactions on neural networks and learning systems*, 35(2): 2747–2758.

Lintner, J. 1975. The valuation of risk assets and the selection of risky investments in stock portfolios and capital budgets. In *Stochastic optimization models in finance*, 131–155. Elsevier.

Liu, Y.; Zheng, Y.; Zhang, D.; Chen, H.; Peng, H.; and Pan, S. 2022. Towards unsupervised deep graph structure learning. In *Proceedings of the ACM Web Conference 2022*, 1392–1403.

Oord, A. v. d.; Li, Y.; and Vinyals, O. 2018. Representation learning with contrastive predictive coding. *arXiv preprint arXiv:1807.03748*.

Qin, Y.; Song, D.; Chen, H.; Cheng, W.; Jiang, G.; and Cottrell, G. 2017. A dual-stage attention-based recurrent neural network for time series prediction. *arXiv preprint arXiv:1704.02971*.

Sawhney, R.; Agarwal, S.; Wadhwa, A.; Derr, T.; and Shah, R. R. 2021. Stock selection via spatiotemporal hypergraph attention network: A learning to rank approach. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, 497–504.

Sawhney, R.; Agarwal, S.; Wadhwa, A.; and Shah, R. R. 2020. Spatiotemporal hypergraph convolution network for stock movement forecasting. In *2020 IEEE International Conference on Data Mining (ICDM)*, 482–491. IEEE.

Sharpe, W. F. 1964. Capital asset prices: A theory of market equilibrium under conditions of risk. *The journal of finance*, 19(3): 425–442.

Su, H.; Wang, X.; Qin, Y.; and Chen, Q. 2024. Attention based adaptive spatial–temporal hypergraph convolutional networks for stock price trend prediction. *Expert Systems with Applications*, 238: 121899.

Treynor, J. L. 1961. *Toward a theory of market value of risky assets*.

Uddin, A.; and Yu, D. 2020. Latent factor model for asset pricing. *Journal of Behavioral and Experimental Finance*, 27: 100353.

Vaswani, A. 2017. Attention is all you need. *arXiv preprint arXiv:1706.03762*.

Veličković, P.; Cucurull, G.; Casanova, A.; Romero, A.; Lio, P.; and Bengio, Y. 2017. Graph attention networks. *arXiv preprint arXiv:1710.10903*.

Xia, H.; Ao, H.; Li, L.; Liu, Y.; Liu, S.; Ye, G.; and Chai, H. 2024. CI-STHPAN: Pre-trained Attention Network for Stock Selection with Channel-Independent Spatio-Temporal Hypergraph. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, 9187–9195.

Xu, W.; Liu, W.; Wang, L.; Xia, Y.; Bian, J.; Yin, J.; and Liu, T.-Y. 2021. Hist: A graph-based framework for stock trend forecasting via mining concept-oriented shared information. *arXiv preprint arXiv:2110.13716*.

Zhang, L.; Aggarwal, C.; and Qi, G.-J. 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.

Zhou, D.; Huang, J.; and Schölkopf, B. 2006. Learning with hypergraphs: Clustering, classification, and embedding. *Advances in neural information processing systems*, 19.

Zou, J.; Cao, H.; Liu, L.; Lin, Y.; Abbasnejad, E.; and Shi, J. Q. 2022. Astock: A new dataset and automated stock trading based on stock-specific news analyzing model. *arXiv preprint arXiv:2206.06606*.