RSAP-DFM: Regime-Shifting Adaptive Posterior Dynamic Factor Model for Stock Returns Prediction

Quanzhou Xiang, **Zhan Chen**, **Qi Sun** and **Rujun Jiang***
Fudan University
{qzxiang22, chenz22, sunqi22}@m.fudan.edu.cn, rjjiang@fudan.edu.cn

Abstract

As the latest development of asset pricing research, how to use machine learning to improve the performance of factor models has become a topic of concern in recent years. The variability of the instantaneous macro environment brings great difficulties to quantitative investment, so the extended factor model must learn how to self-adapt to extract the macro pattern from the massive stock volume and price information, and how to continuously map the extracted macro pattern to the stock investment is also an open question. To this end, we propose the first continuous regime-based dynamic factor model, RSAP-DFM, which adaptively extracts continuous macroeconomic information and completes the dynamic explicit mapping of stock returns for the first time through dual regime shifting, while the adversarial posterior factors effectively correct the mapping deviation of prior factors. In addition, our model integrates an innovative two-stage optimization algorithm and normally distributed sampling, which further enhances the robustness of the model. Performance on three real stock datasets validates the validity of our model, which exceeds any previous methods available.

1 Introduction

Asset pricing, a central topic in modern financial research, seeks to explain the cross-sectional differences in the expected returns of different assets. Influenced by the Capital Asset Pricing Model (CAPM) [Sharpe, 1964], the Fama-French three-factor model ushers in the era of factor modeling. This approach conceptualizes stock excess returns as combinations of multiple factor-based returns, with these factors symbolizing distinct sources of excess returns. However, the traditional static factor exposure is now being reevaluated in favor of dynamic models, acknowledging time-varying influences on investment performance [Stock and Watson, 2011]. Unlike linear factor models, machine learning can use multi-layer nonlinear networks, offering a more robust framework for capturing complex financial relationships.

Given the above advantages, the application of machine learning to the prediction of cross-sectional expected excess returns becomes the focus of academic attention. Existing studies transfer machine learning methods such as tree model [Chen and Guestrin, 2016], time series model [Sutskever *et al.*, 2014], graph model [Velickovic *et al.*, 2017], and attention model [Ding *et al.*, 2020] to stock prediction problem. Although the performance is improved, interpretability remains a challenge. To further describe the interdependence between intertemporal stock features, researchers develop some machine learning models specifically for financial markets [Lin *et al.*, 2021a; Xu *et al.*, 2021b]. Compared to these model-free approaches, machine learning technology also offers an entirely new perspective on traditional dynamic factor models (linear relationship), which dynamically learn factor exposures and factors returns in different periods [Kelly *et al.*, 2019; Uddin and Yu, 2020; Gu *et al.*, 2021; Duan *et al.*, 2022].

However, we believe that two unresolved issues remain. Firstly, in the current complex and volatile economic environment, the key to the success of the investment model is whether it can judge the macroeconomic state in time and integrate it into the investment decision. Existing machine learning and dynamic factor models struggle to encapsulate this characteristic effectively. [Wei *et al.*, 2023] introduces an innovative concept of hierarchical market states, yet the approach of broadly categorizing economic features into discrete market states and employing black-box integration appears somewhat imprecise. Secondly, traditional data-driven factor construction is polluted by the low signal-to-noise ratio of stock data, making predicting returns accurately difficult.

In this paper, we propose a **R**egime-**S**hifting **A**daptive **P**osterior **D**ynamic **F**actor **M**odel for stock returns prediction called **RSAP-DFM**. In our approach, we reshape the relationship between factor construction, factor return, and factor exposure. We leverage a multi-head attention mechanism to dynamically generate and sample factors from normal distributions. These factors are crucial components within our dynamic factor model (DFM), which is instrumental in predicting stock returns. To enhance the precision of current factor state identification, we have introduced a novel regime recognition method that is applied distinctively to factor returns and factor loadings. Specifically, we employ a "jumping encoder" to capture the influence of macroeconomic states

*Corresponding Author

derived from historical stock time series data on factor returns. Additionally, a "loading encoder" characterizes the influence of these macro states on factor loadings and idiosyncratic returns. Furthermore, we have devised an innovative bilevel optimization algorithm for constructing posterior factors through adversarial learning, which enhances the robustness of the factor constructions and optimizes their predictive returns. Our findings substantiate the superior performance of RSAP-DFM compared to alternative baseline methods. The contributions of our paper are as follows:

1. We propose a stock returns prediction framework named RSAP-DFM, which discards the artificial factor construction process by constructing factors sampled from the normal distribution and posterior factors constructed by adversarial learning. To the best of our knowledge, we are the first to propose gradient-based posterior factor construction and optimize it in a bilevel form.

2. To the best of our knowledge, in a neural-network training framework, we propose a dual regime shifting structure and apply it in DFM, which is the first to describe explicitly, rather than in a black box, how macroeconomic state affects stock returns; furthermore, we are also the first to perform regime shifting in continuous rather than discrete intervals.

3. We conduct a comprehensive array of experiments employing real stock market data. Experiments on datasets from the A-share market demonstrate our achievement of an unprecedented state-of-the-art (SOTA) performance, thus surpassing prior benchmarks.

2 Related Works

2.1 Deep Learning in Technical Analysis

Since Alexnet [Krizhevsky *et al.*, 2012] proves the power of neural networks, the development and application of deep learning have surpassed imagination. In stock prediction, technical analysis, as a core area, is particularly suitable for the application of deep learning models. Technical analysis uses only numerical characteristics of stocks and markets as data. [Selvin *et al.*, 2017] uses RNN [Rumelhart *et al.*, 1986], LSTM [Hochreiter and Schmidhuber, 1997], and CNN-sliding window for stock price prediction. Using LSTM as the backbone like [Nelson *et al.*, 2017], [Zhang *et al.*, 2017] also proposes a high-frequency trading prediction model using only historical data.

Based on traditional neural network models, more advanced neural networks have been developed for stock forecasting. The improvement of classical deep learning models specific to the stock market becomes an essential direction of technical analysis, such as adversarial LSTM [Qin *et al.*, 2017], multi-scale Gaussian prior Transformer [Ding *et al.*, 2020], and adaptive RNN [Du *et al.*, 2021]. How to better analyze the volume and price information and correlation relationship of the stock market becomes another major direction for predicting returns. Existing studies use hypergraph ranking method [Sawhney *et al.*, 2021], instance-wise graph-based method [Xu *et al.*, 2021a], optimal transport [Lin *et al.*, 2021b], and adaptive long-short pattern transformer [Wang *et al.*, 2022] to deeply characterize the stock information and relationship. However, research addressing regime switching via deep learning methodologies remains limited. [Mari and Mari, 2023] only focuses on energy commodity prices, while [Wei *et al.*, 2023] learns the hierarchical latent space by using a moving-average online learning algorithm to identify discrete market institutions. Different from any regime switching approach, we believe that market regime shifts in continuous intervals and maps explicitly.

2.2 Factor Model

Following the traditional definition, the factor model here only refers to a model in which stock returns are a linear combination of exposure returns on each factor. Traditional factor models in finance have progressed from the single market factor in CAPM [Sharpe, 1964] through Ross's development of the Arbitrage Pricing Theory [Ross, 2013] and the expansion to multi-factor models like the Fama-French three-factor model [Fama and French, 1992] and the Carhart four-factor model [Carhart, 1997]. The latest factor model has developed hundreds of factors, forming the famous "factor zoo".

In dynamic factor models, factors and factor exposures vary over time and are typically derived from individual characteristics. Dynamic factor models are originally proposed by Geweke [Geweke, 1977] as a time-series extension of cross-sectional factor models; for a detailed explanation and collection of traditional dynamic factor models, see [Stock and Watson, 2011]. However, the dynamic factor model has limitations in that it relies on complex statistical techniques for factor construction, while machine learning can obscure these methods as a black box. Unlike the linear assumption of Instrumented Principal Components Analysis (IPCA) [Kelly *et al.*, 2019], [Gu *et al.*, 2021] models factor exposures as flexible nonlinear functions of covariates. Furthermore, [Lin *et al.*, 2021a] proposes a risk factor model to better explain the variance of stock returns, and [Wei *et al.*, 2022] proposes a deep multi-factor model to build a dynamic and multi-relational stock graph in a hierarchical structure. The latest research is [Duan *et al.*, 2022], which considers using variational autoencoders to model noise based on [Gu *et al.*, 2021]. However, existing factor models are limited to wrapping each step of the traditional factor model in machine learning rather than incorporating new ideas into model design.

3 Preliminaries

In this section, we briefly introduce the dynamic factor model as a backbone, then define **regime shifting** and introduce our problem.

To broadly convey this concept, we consistently utilize the notation and problem definition outlined below. In the complete set Ω, there are a total of T subsets $\{\boldsymbol{x}_t, \boldsymbol{r}_t\}$ sorted in chronological order: $\Omega = \{\{\boldsymbol{x}_1, \boldsymbol{r}_1\}, \ldots, \{\boldsymbol{x}_T, \boldsymbol{r}_T\}\}$ where $t = 1, \ldots, T$ is the total number of trading days in the data set. $\boldsymbol{x}_t \in \mathbb{R}^{N_t \times B \times F}$ denotes F features (such as price, volume, text-data) of N_t stocks in past B time-steps, and $\boldsymbol{r}_t \in \mathbb{R}^{N_t}$ denotes the future returns of N_t cross-sectional stocks. The goal of this type of problem is to predict cross-sectional future returns through current characteristics, which can be summarized as follows:

$$\hat{\boldsymbol{r}}_t = f(\boldsymbol{x}_t). \tag{1}$$

3.1 Dynamic Factor Model

To fulfill the objective mentioned above, it is imperative to provide the reader with a comprehensive understanding of the dynamic factor model. A typical representation of the dynamic factor model is formulated as follows:

$$\hat{\boldsymbol{r}}_t = \boldsymbol{\alpha}_t + \sum_{k=1}^{K} \boldsymbol{\beta}_t^{(k)} \boldsymbol{\lambda}_t^{(k)} + \boldsymbol{\varepsilon}_t, \tag{2}$$

where $\boldsymbol{r}_t = (\boldsymbol{y}_{t+1} - \boldsymbol{y}_t)/\boldsymbol{y}_t \in \mathbb{R}^{N_t}$ denotes the future 1-day returns of N_t stocks at time step t, and $\boldsymbol{y}_t \in \mathbb{R}^{N_t}$ denotes the price at time step t, which can be represented by a variety of prices, such as the opening price, the closing price, or the volume weighted average price (VWAP). $\boldsymbol{\alpha}_t \in \mathbb{R}^{N_t}$ denotes the vector of stock idiosyncratic returns, $\boldsymbol{\beta}_t \in \mathbb{R}^{N_t \times K}$ denotes the factor exposure, $\boldsymbol{\lambda}_t \in \mathbb{R}^K$ denotes the return vector of K factors. $\boldsymbol{\varepsilon}_t$ denotes the idiosyncratic noises, with $\mathbb{E}(\boldsymbol{\varepsilon}_t) = 0$.

When it comes to applying machine learning to the dynamic factor model, the common practice is to fit an unknown prediction function f to a large amount of data and predict the intertemporal return using the historical data of the stock. The historical data of the stock determines all the parameters of f:

$$\hat{\boldsymbol{r}}_t = f(\boldsymbol{x}_t) = \alpha(\boldsymbol{x}_t) + \beta(\boldsymbol{x}_t)\lambda(\boldsymbol{x}_t). \tag{3}$$

3.2 Our Problem

Definition 1. *The market environment, also known as the regime, changing within the continuous interval $[a, b]$ is defined as **regime shifting**.*

Different from the regime switching which changes in the discrete interval $\{a, a_1, ...b_{n-1}, b\}$, we believe that the regime change is in a continuous interval rather than several fixed discrete regimes. This dimension helps illuminate the complex correlation between economic dynamics and stock return.

Compared with the traditional dynamic factor model, we introduce our new dynamic factor model by dual dynamic *regime shifting*:

$$\hat{\boldsymbol{r}} = f(\boldsymbol{x}) = \alpha(\boldsymbol{x}, \boldsymbol{m}) + \beta(\boldsymbol{x}, \boldsymbol{m})\lambda(\boldsymbol{x}, \boldsymbol{m}), \tag{4}$$

where $\boldsymbol{m}$ denotes *regime shifting* features. In RSAP-DFM, $\boldsymbol{m}$ can be adaptively extracted from the cross-sectional stock features. To avoid notational confusion, we ignore the subscript t in the following pages because our goal is to predict cross-sectional returns, and each prediction will involve only a subset of Ω.

4 Methodology

In this section, we elaborate on our Regime-Shifting Adaptive Posterior Dynamic Factor Model (RSAP-DFM) framework. Figure 1 is the illustration of our RSAP-DFM framework. In this section, we first show how the feature extractor works for stock hidden states and macro regime embeddings and present a dual-encoder to handle dual regime-shifting features (**4.1**). Then, we describe our prior factor encoder based on a multi-head attention mechanism to encode factors with regime-shifting jumping (**4.2**). With regime-shifting loading, we describe our dual dynamic factor model for stock prediction, which combines neural networks to coordinate α, β, and factors (**4.3**). After that, we introduce factor posterior following the idea of adversarial learning to further modify factor selection and factor return (**4.4**). At last, we introduce dual tasks as a bilevel form and present an innovative two-stage optimization method for training (**4.5**).

4.1 Feature Embedding Extractor

Stock Feature Construction

Future stock returns can be predicted by features constructed from stock time series data. We utilize a GRU model to extract hidden stock features from $\boldsymbol{x}$:

$$\begin{aligned} \boldsymbol{z}_t &= \sigma\left(\boldsymbol{W}_z \boldsymbol{x} + \boldsymbol{U}_z \boldsymbol{h}_{t-1} + \boldsymbol{b}_z\right), \\ \boldsymbol{r}_t &= \sigma\left(\boldsymbol{W}_r \boldsymbol{x} + \boldsymbol{U}_r \boldsymbol{h}_{t-1} + \boldsymbol{b}_r\right), \\ \tilde{\boldsymbol{h}}_t &= \tanh\left(\boldsymbol{W}_h \boldsymbol{x} + \boldsymbol{U}_h\left(\boldsymbol{r}_t \odot \boldsymbol{h}_{t-1}\right) + \boldsymbol{b}_h\right), \\ \boldsymbol{h}_t &= \boldsymbol{z}_t \odot \boldsymbol{h}_{t-1} + (1 - \boldsymbol{z}_t) \odot \tilde{\boldsymbol{h}}_t, \end{aligned} \tag{5}$$

where $\odot$ is Hadamard product, $\boldsymbol{W}_*$ and $\boldsymbol{U}_*$ are trainable weights and biases. In the following section, we use $\boldsymbol{e}$ to refer to $\boldsymbol{h}_t$.

Dual-Regime-Shifting Encoder Structure

Regime shifting in the dynamic factor model, in our perspective, assumes a dual role. Firstly, it influences the overall market's returns, thereby impacting factor returns. Secondly, it exerts varying degrees of influence on returns of individual stocks, consequently altering the factor exposure of stocks.

To avoid providing additional information and thus making the model incomparable, we dynamically extract macroeconomic features from the stock hidden features $\boldsymbol{e}$:

$$\boldsymbol{m} = \phi_{\text{DM}}(\boldsymbol{e}), \tag{6}$$

where $\boldsymbol{m} \in \mathbb{R}^M$ denotes M regime shifting features.

Regime Shifting Jumping Encoder. When we choose macroeconomic features $\boldsymbol{m}$, capturing their temporal patterns is not needed; we just need to know the current regime and characterize how the regime affects the return of each factor. Therefore, after extracting relatively low-dimensional features, the regime shifting jumping encoder ϕ_{RJ} must have components to further project into K-dimensional space:

$$\begin{aligned} \boldsymbol{h}^{\text{rj}} &= \text{LeakyReLU}(\boldsymbol{w}^{\text{rj}} \boldsymbol{m} + \boldsymbol{b}^{\text{rj}}), \\ \boldsymbol{e}^{\text{rj}} &= \phi_{\text{pj}}^{\text{rj}}(\boldsymbol{h}^{\text{rj}}), \end{aligned} \tag{7}$$

where $\boldsymbol{e}^{\text{rj}} \in \mathbb{R}^K$, K is the number of factors.

Regime Shifting Loading Encoder. Similar to ϕ_{RJ}, our regime shifting loading encoder ϕ_{RL} also adopts a similar architecture. Note that the purpose of introducing macroeconomic features into the model is to characterize regime shifting in terms of factor loading, so ϕ_{RL} will output $\boldsymbol{l}_\alpha$ and $\boldsymbol{l}_\beta$. But unlike ϕ_{RJ}, ϕ_{RL} shows the different effects of regime shifting on the factor loadings of different stocks, so ϕ_{RL} needs to introduce the individual features of the stock:

$$\begin{aligned} \boldsymbol{m}^{\text{rl}} &= \text{LeakyReLU}(\boldsymbol{w}_1^{\text{rl}} \boldsymbol{m} + \boldsymbol{b}_1^{\text{rl}}), \\ \boldsymbol{h}^{\text{rl}} &= \text{LeakyReLU}(\boldsymbol{w}_2^{\text{rl}}(\boldsymbol{m}^{\text{rl}}, \boldsymbol{e}) + \boldsymbol{b}_2^{\text{rl}}), \\ \boldsymbol{l}_\alpha &= \phi_{\text{pj}}^{\alpha}(\boldsymbol{h}^{\text{rl}}), \quad \boldsymbol{l}_\beta = \phi_{\text{pj}}^{\beta}(\boldsymbol{h}^{\text{rl}}). \end{aligned} \tag{8}$$

where $\boldsymbol{l}_\alpha \in \mathbb{R}^N$ denotes regime shifting on the α loading, $\boldsymbol{l}_\beta \in \mathbb{R}^{N \times K}$ denotes regime shifting on the β exposure.

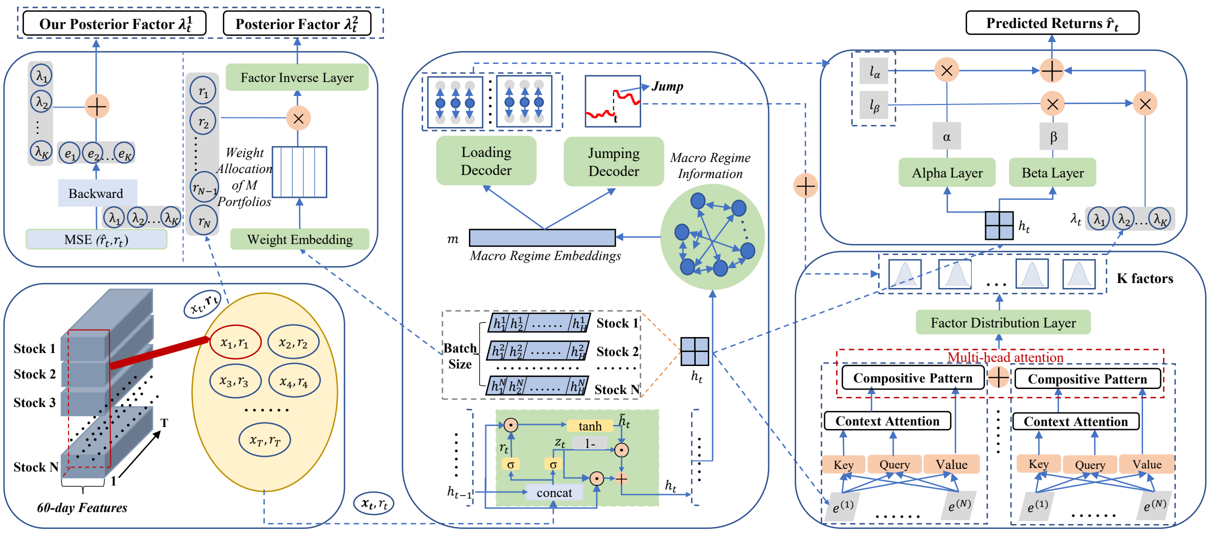


Figure 1: Our RSAP-DFM Framework

4.2 Multi-Head Attention-Based Factor Encoder

To enhance the model's resilience, we incorporate a variational encoder that introduces stochasticity into factor returns, thus rendering them as random variables conforming to a normal distribution. Prior factor encoder ϕ_{prior} extracts prior factors $\boldsymbol{\lambda}_{\text{prior}}$ from hidden features of stocks without revealing future information.

We divide the "factor mining" process into two perspectives: factor construction and factor return. In traditional factor investing, each factor reflects a risk premium in the market, and the factor construction is usually constructed manually, while the factor return is obtained by using the cross-sectional order of the factor values to build a long-short combination and there is no relationship between the two.

The ϕ_{prior} we design incorporates both factor construction and factor return: a multi-headed attention mechanism characterizes factor construction and partial factor return; a regime shifting factor encoder characterizes the other part of factor returns from an unbiased global perspective:

$$\begin{aligned}[\boldsymbol{\mu}_{\text{prior}}, \boldsymbol{\sigma}_{\text{prior}}] &= \phi_{\text{prior}}(\boldsymbol{e}, \boldsymbol{m}) \\ &:= \pi_{\text{prior}}(\phi_{\text{att}}(\boldsymbol{e}), \phi_{\text{RJ}}(\boldsymbol{m})).\end{aligned} \tag{9}$$

Formally, ϕ_{att} stitches multiple attention heads together:

$$\begin{aligned}\boldsymbol{k}^{(i)} &= \boldsymbol{w}_{\text{key}}\boldsymbol{e}^{(i)}, \boldsymbol{v}^{(i)} = \boldsymbol{w}_{\text{value}}\boldsymbol{e}^{(i)}, \\ \boldsymbol{a}_{\text{att}}^{(i)} &= \frac{\max\left(0, \frac{\boldsymbol{q}\boldsymbol{k}^{(i)}}{\|\boldsymbol{q}\|_2 \cdot \|\boldsymbol{k}^{(i)}\|_2}\right)}{\sum_{i=1}^{N} \max\left(0, \frac{\boldsymbol{q}\boldsymbol{k}^{(i)}}{\|\boldsymbol{q}\|_2 \cdot \|\boldsymbol{k}^{(i)}\|_2}\right)}, \\ \boldsymbol{h}_{\text{att}} &= \varphi_{\text{att}}(\boldsymbol{e}) = \sum_{i=1}^{N} \boldsymbol{a}_{\text{att}}^{(i)}\boldsymbol{v}^{(i)}, \\ \boldsymbol{h}_{\text{multi}} &= \text{Concat}([\boldsymbol{h}_{\text{att}_1}, ..., \boldsymbol{h}_{\text{att}_K}]).\end{aligned} \tag{10}$$

Then we merge factor construction with factor return through π_{prior} to form an end-to-end factor coding process without the hand-made construction part:

$$\begin{aligned}&[\boldsymbol{\mu}_{\text{prior}}, \boldsymbol{\sigma}_{\text{prior}}] = \pi_{\text{prior}}(\boldsymbol{h}_{\text{multi}}, \boldsymbol{e}^{\text{rj}}), \\ &\boldsymbol{\lambda}_{\text{prior}} \sim \mathcal{N}(\boldsymbol{\mu}_{\text{prior}}, \text{diag}(\boldsymbol{\sigma}_{\text{prior}}^2)).\end{aligned} \tag{11}$$

4.3 Dual Dynamic Factor Model

Now that we have explained in detail how to output the optimal prior factors of a dynamic factor model, let us take another look at our problem (4), then reconstruct the model based on the known prior factors:

$$\hat{\boldsymbol{r}} = \phi_{\text{DDF}}(\boldsymbol{e}, \boldsymbol{m}, \boldsymbol{\lambda}) = \alpha(\boldsymbol{e}, \boldsymbol{m}) + \beta(\boldsymbol{e}, \boldsymbol{m})\boldsymbol{\lambda}. \tag{12}$$

Alpha Layer. In the Alpha layer, we generate idiosyncratic returns denoted as $\boldsymbol{\alpha}$, which are derived from latent features $\boldsymbol{e}$. It's posited that $\boldsymbol{\alpha}$ follows a Gaussian distribution, characterized as $\boldsymbol{\alpha} \sim \mathcal{N}\left(\boldsymbol{\mu}_\alpha, \text{diag}\left(\boldsymbol{\sigma}_\alpha^2\right)\right)$. Here, the mean $\boldsymbol{\mu}_\alpha \in \mathbb{R}^N$ and the standard deviation $\boldsymbol{\sigma}_\alpha \in \mathbb{R}^N$ are produced by a distribution network $\pi_\alpha := [\boldsymbol{\mu}_\alpha, \boldsymbol{\sigma}_\alpha]$:

$$\begin{aligned}\boldsymbol{h}_\alpha^{(i)} &= \text{LeakyReLU}\left(\boldsymbol{w}_\alpha \boldsymbol{e}^{(i)} + \boldsymbol{b}_\alpha\right), \\ \boldsymbol{\mu}_\alpha^{(i)} &= \boldsymbol{w}_{\alpha_\mu}\boldsymbol{h}_\alpha^{(i)} + \boldsymbol{b}_{\alpha_\mu}, \\ \boldsymbol{\sigma}_\alpha^{(i)} &= \text{Softplus}\left(\boldsymbol{w}_{\alpha_\sigma}\boldsymbol{h}_\alpha^{(i)} + \boldsymbol{b}_{\alpha_\sigma}\right),\end{aligned} \tag{13}$$

where $\boldsymbol{h}_\alpha^{(i)} \in \mathbb{R}^H$ denotes the hidden space.

Beta Layer. Beta Layer is tasked with calculating factor exposure $\boldsymbol{\beta} \in \mathbb{R}^{N \times K}$ which is a linear mapping from the latent features $\boldsymbol{e}$:

$$\boldsymbol{\beta}^{(i)} = \varphi_\beta(\boldsymbol{e}^{(i)}) = \boldsymbol{w}_\beta \boldsymbol{e}^{(i)} + \boldsymbol{b}_\beta. \tag{14}$$

Regime Shifting. Here, we present the dual dynamic factor model. We believe that regime shifting can be reflected in factor exposure and stock idiosyncratic returns, where macroeconomic information is explicitly mapped into dynamic factor model. Since market information represented by regime shifting is also dynamic, we name the model the **dual dynamic factor model**. We take the form of factor loadings (or "element-wise multiplication"):

$$\phi_{\text{DDF}}(\boldsymbol{e}, \boldsymbol{m}, \boldsymbol{\lambda}) = \boldsymbol{l}_\alpha \alpha(\boldsymbol{e}) + \boldsymbol{l}_\beta \beta(\boldsymbol{e}) \boldsymbol{\lambda}. \tag{15}$$

Consequently, the output of the dual dynamic factor model, denoted as $\hat{\boldsymbol{r}} \in \mathbb{R}^N$, is also governed by a Gaussian distribution, specifically $\hat{\boldsymbol{r}}^{(i)} \sim \mathcal{N}(\boldsymbol{\mu}_r^{(i)}, \boldsymbol{\sigma}_r^{(i)})$, where

$$\begin{aligned} \boldsymbol{\mu}_r^{(i)} &= \boldsymbol{l}_\alpha^{(i)} \boldsymbol{\mu}_\alpha^{(i)} + \sum_{k=1}^{K} \boldsymbol{l}_\beta^{(i,k)} \boldsymbol{\beta}^{(i,k)} \boldsymbol{\mu}_{\text{prior}}^{(k)}, \\ \boldsymbol{\sigma}_r^{(i)} &= \sqrt{\left(\boldsymbol{l}_\alpha^{(i)2} \boldsymbol{\sigma}_\alpha^{(i)2} + \sum_{k=1}^{K} \boldsymbol{l}_\beta^{(i,k)2} \boldsymbol{\beta}^{(i,k)2} \boldsymbol{\sigma}_{\text{prior}}^{(k)2} \right)}. \end{aligned} \tag{16}$$

4.4 Adaptive Factor Posterior Module

The gradient-based factor posterior absorbs the idea of adversarial learning [Goodfellow *et al.*, 2014] and adds an adaptive perturbation to the data in a batch to form adversarial factors $\boldsymbol{\lambda}_G \in \mathbb{R}^K$ during each iteration training. We construct a perturbation such that applying it causes the most significant change in the prediction compared to the perturbation of the same size and approximate the predicted returns formed by $\boldsymbol{\lambda}_G$ and future returns, thus enhancing the robustness of prior factors.

By calculating the MSE loss between predicted returns $\hat{\boldsymbol{r}}$ and returns $\boldsymbol{r}$, and calculating the gradient of the loss with respect to $\boldsymbol{\lambda}_{\text{prior}}$, the adaptive perturbation is constructed:

$$\boldsymbol{\lambda}_G = \boldsymbol{\lambda}_{\text{prior}} + \epsilon \frac{\boldsymbol{g}^\star}{\|\boldsymbol{g}^\star\|}, \quad \boldsymbol{g}^\star = \frac{\partial l\left(\boldsymbol{r}, \hat{\boldsymbol{r}}\right)}{\partial \boldsymbol{\lambda}_{\text{prior}}}. \tag{17}$$

4.5 Our Algorithm

Our basic idea is to construct the model training as a bilevel optimization form by introducing a posterior factor to construct an adaptive auxiliary training task to improve the performance of the main task. The basic principle of our algorithm is that both the main task and the auxiliary task can share parameters. Following the thought of traditional finance, it is a natural choice to separate factor construction and factor model optimization. We split the model parameters into two parts so that there are two sets of parameters that can be learned alternately:

$$\begin{aligned} &\min_{\theta_D} \mathcal{L}_{\mathcal{M}}\left(\boldsymbol{x}, \boldsymbol{r}; \theta_C^*, \theta_D\right); \quad \text{s.t.}, \\ &\theta_C^* = \underset{\theta_C}{\operatorname{argmin}} \frac{1}{|\Omega_{\text{tr}}|} \sum_{(\boldsymbol{x},\boldsymbol{r})\sim\Omega_{\text{tr}}} \mathbb{E}\, \ell_A(\boldsymbol{x}, \boldsymbol{r}; \theta_C, \theta_D). \end{aligned} \tag{18}$$

where Ω_{tr} denotes the training part of Ω, θ_C denotes the parameters of factor construction part, θ_D denotes the parameters of dual dynamic factor model part, $\mathcal{L}_{\mathcal{M}}$ denotes the loss function of the main task,ℓ_A denotes the loss function of the auxiliary task on data $(\boldsymbol{x}, \boldsymbol{r})$.

Intuitively, we hope that this auxiliary task can promote the performance of our model on the main task. Here, we choose gradient-based posterior factors to participate in the construction of auxiliary tasks to construct more robust factors so as to minimize the loss of the model on the main task.

Algorithm 1 Optimization Algorithm

Input: Training epochs E; update iterations I; training dataset $\Omega_{\text{tr}} = \{\Omega^1_{\text{Batch}}, ..., \Omega^{N_{train}}_{\text{Batch}}\}$;
Parameter: Model parameters θ_C,θ_D;
Output: Trained model parameters θ_C,θ_D;

```
for e ← 1 : E do
    Randomly Sample Ω^a_Batch from Ω, batch size B_1 equals to stock num of Ω^a_Batch;
    Randomly Sample Ω^b_Batch from Ω, batch size B_2 equals to stock num of Ω^b_Batch;
    for i ← 1 : I do
        For each (x, r) ∈ Ω^a_Batch, calculate the loss of auxiliary task:
            L_A = Σ_{(x,r)~Ω^a_Batch} ℓ_A(x, r; θ_C, θ_D);
        Update θ_C: θ_C ← θ_C − (η_1/B_1)∇_{θ_C} L_A;
        For each (x, r) ∈ Ω^b_Batch, calculate the loss of main task:
            L_M = Σ_{(x,r)~Ω^b_Batch} ℓ_M(x, r; θ_C, θ_D);
        Update θ_D: θ_D ← θ_D − (η_2/B_2)∇_{θ_D} L_M;
    end for
end for
return θ_C,θ_D
```

Our approach is shown in Algorithm 1. The training process consists of E epochs: in the inner layer, we fix θ_D and optimize θ_C on the auxiliary task; On the outer layer, optimize θ_D on the main task by fixing θ_C. Such algorithm setting can be generalized, but in this experiment, the main task loss and auxiliary task loss are respectively:

$$\begin{aligned} \ell_M &= \mathbf{MSE}\left(\hat{\boldsymbol{r}}, \boldsymbol{r}\right), \\ \ell_A &= \mathbf{MSE}\left(\phi_{\text{DDF}}(\boldsymbol{e}, \boldsymbol{m}, \boldsymbol{\lambda}_G), \boldsymbol{r}\right) + \mathbf{MSE}\left(\hat{\boldsymbol{r}}, \boldsymbol{r}\right). \end{aligned} \tag{19}$$

5 Experiments

In this section, we compare our framework with other major related works and verify the effectiveness, state-of-the-art, and robustness of our framework through multiple experiments. The following five questions are explored in the following experiments:

Question 1: Does our model outperform current state-of-the-art models in stock prediction?

Question 2: Can bilevel optimization promote the performance of our model?

Question 3: Is dual regime-shifting architecture useful for improving performance?

Question 4: Is the posterior factor we designed better than the previous methods?

Question 5: Can our model's superior prediction performance generate superior investment performance?

5.1 Experiment Setting

Setting Details. We evaluate the performance of our RSAP-DFM model on China's A-share market, which is split into a training period (2008.01-2014.12), a validation period (2015.01-2016.02), and a test period (2017.01-2020.08). Our stock features come from Alpha360 on the Qlib platform

Baselines	
XGBoost[Chen and Guestrin, 2016]	A scalable end-to-end tree boosting method.
LightGBM[Ke *et al.*, 2017]	A boosting decision tree, with gradient-based one-side sampling and exclusive feature bundling.
MLP	A non-linear factor model with multilayer perceptrons, with an LSTM model to extract hidden features from temporal features.
Transformer[Ding *et al.*, 2020]	A stock return prediction framework with the transformer architecture.
ALSTM[Qin *et al.*, 2017]	A variant of LSTM, which adds an attention mechanism to automatically learn and focus on the information at critical moments.
GATs[Velickovic *et al.*, 2017]	A graph attention network that treats each stock as a node to specify different weights to different nodes by stacking layers.
AdaRNN[Du *et al.*, 2021]	An adaptive RNN which aims to solve the temporal covariate shift problem of time series.
IGMTF[Xu *et al.*, 2021a]	An instance-based graph framework to make predictions using the interdependencies of different variables at different time stamps.
TRA[Lin *et al.*, 2021b]	A learning algorithm based on optimal transport for learning multiple stock trading patterns.
TCTS[Wu *et al.*, 2021]	A sequence learning model combined with a learnable scheduler that adaptively selects auxiliary tasks for training.
FactorVAE[Duan *et al.*, 2022]	A dynamic factor model with the variational autoencoder and prior-posterior learning.

Table 1: Baselines compared to RSAP-DFM framework

	CSI100				CSI300				CSI500			
Model	IC	ICIR	RANK IC	RANK ICIR	IC	ICIR	RANK IC	RANK ICIR	IC	ICIR	RANK IC	RANK ICIR
XGBoost	0.0517	0.3096	0.0547	0.3744	0.0517	0.4642	0.0596	0.5696	0.0517	0.4642	0.0596	0.5696
LightGBM	0.0502	0.3211	0.0595	0.3858	0.0560	0.4703	0.0654	0.5608	0.0555	0.5286	0.0667	0.6554
MLP	0.0413	0.2466	0.0427	0.2676	0.0358	0.2587	0.0427	0.3246	0.0353	0.2754	0.0460	0.4076
Transformer	0.0475	0.2668	0.0578	0.3435	0.0415	0.3094	0.0564	0.428	0.0396	0.3359	0.0533	0.4624
ALSTM	0.0574	0.3340	0.0655	0.3866	0.0658	0.5394	0.0763	0.6231	0.0586	0.5919	0.0671	0.6737
GATS	0.0583	0.3526	0.0678	0.4208	0.0717	0.5655	0.0804	0.6520	0.0680	0.6628	0.0798	0.7931
AdaRNN	0.0657	0.3865	0.0697	0.4222	0.0640	0.5077	0.0769	0.6148	0.0705	0.7104	0.0833	0.8694
IGMTF	0.0622	0.3678	0.0708	0.4352	0.0669	0.5250	0.0765	0.6196	0.0653	0.6025	0.0771	0.7387
TRA	0.0674	0.4157	0.0755	0.4784	0.0638	0.5207	0.0731	0.6084	0.0688	0.7046	0.0788	0.8214
TCTS	0.0593	0.3429	0.0632	0.3738	0.0763	0.6633	0.0806	0.6828	0.0718	0.7294	0.0834	**0.8881**
FactorVAE	0.0487	0.3084	0.0518	0.3321	0.0528	0.4417	0.0627	0.5350	0.0606	0.5894	0.0736	0.7491
RSAP-DFM	**0.0768**	**0.4260**	**0.0870**	**0.4852**	**0.0855**	**0.6194**	**0.0961**	**0.6849**	**0.0854**	**0.7317**	**0.1025**	0.8770

Table 2: Overall prediction performance on CSI100, CSI300 and CSI500

[Yang *et al.*, 2020]. This dataset provides six basic transaction information for stocks over the past 60 days, including $opening - high - low - closing$ prices, volume weighted average price (VWAP), and trading volume. We set the label and forecast target as the daily stock return, defined as $\boldsymbol{r}_t = (\boldsymbol{y}_{t+1} - \boldsymbol{y}_t)/\boldsymbol{y}_t$, $\boldsymbol{y}_t$ denotes opening price here because we believe this better captures the hidden information of the stock's history and eliminates the interference of unexpected situations the next day. Since ST stocks are not included, we set the batch size to a float equal to the number of stocks included in the day. We adopt the Pytorch framework to implement our model and perform all experiments on the NVIDIA GEFORCE RTX 2080 GPU.

Baselines. We compare our model with the baseline models in Table 1, which represent the latest related research.

5.2 Overall Performance Evaluation

In this experiment, we verify the performance evaluation of our RSAP-DFM. To show the superiority of our model's prediction performance, we use three significant datasets on the A-share market (CSI100, CSI300, CSI500) and four indicators commonly used in the academic community to describe comprehensively: the information coefficient (IC), the information ratio of IC (ICIR), the rank information coefficient (Rank IC), the information ratio of RankIC (RANK ICIR). Table 2 presents our main experimental results. Compared with the baseline models, RSAP-DFM achieves all-round leading prediction performance on all three datasets. Surprisingly, the performance on the different datasets validates the high robustness of RSAP-DFM. Such results fully demonstrate the effectiveness of our method (**Q1**).

5.3 Ablation Study

Bilevel Optimization and Auxiliary Task Can Promote Model Performance

End-to-end optimization (E2E) via neural networks is widely employed. In the context of our experiments, we seek to validate the efficacy of our novel approach, which centers on auxiliary task-based bilevel optimization and its capacity to enhance model performance. We formulate an end-to-end optimization strategy as the baseline against which we compare our proposed method. Furthermore, we propose a model that removes a posteriori auxiliary task (RS-DFM) as a baseline, that is, optimizes for central task performance only. As depicted in Table 3, our bilevel optimization approach consistently demonstrates its effectiveness in enhancing model performance across diverse factor model configurations. In the entire Chinese stock market, our method consistently outperforms conventional end-to-end optimization techniques (**Q2**).

Dual Regime-Shifting Architecture Drives Our Model Better

In this experiment, we will explain the necessity of our model components. Therefore, we construct three models that modify some components: DFM represents the dynamic factor model part, RSAP-DFM(wo-Jump) represents the removal of the regime-shifting jumping encoder component, and RSAP-DFM(wo-Load) represents the removal of the regime-shifting loading encoder component. These models help analyze the effectiveness of RSAP-DFM components.

As shown in Table 3, the performance of our model is better in almost all metrics than that of the model after changing the components. The following conclusions can be drawn: 1) Compared with models based on only one region shifting

	Model	Method	IC	ICIR	RANK IC	RANK ICIR
CSI100	RS-DFM	E2E	0.0697	0.3713	0.0802	0.4359
	RSAP-DFM	E2E	0.0761	**0.4426**	0.0824	0.4800
	DFM	E2E	0.0752	0.4134	0.0833	0.4735
	RSAP-DFM(wo-Jump)	Bilevel	0.0743	0.4020	0.0844	0.4782
	RSAP-DFM(wo-Load)	Bilevel	0.0738	0.4056	0.0861	0.4852
	RSAP-DFM	Bilevel	**0.0768**	0.4260	**0.0870**	**0.4852**
CSI300	RS-DFM	E2E	0.0791	0.5604	0.0945	0.6556
	RSAP-DFM	E2E	0.0767	0.5698	0.0922	0.6605
	DFM	E2E	0.0744	0.4941	0.0901	0.5996
	RSAP-DFM(wo-Jump)	Bilevel	0.0766	0.5502	0.0907	0.6490
	RSAP-DFM(wo-Load)	Bilevel	0.0773	0.5390	0.0919	0.6321
	RSAP-DFM	Bilevel	**0.0855**	**0.6194**	**0.0961**	**0.6849**
CSI500	RS-DFM	E2E	0.0761	0.6703	0.0957	0.8285
	RSAP-DFM	E2E	0.0743	0.6641	0.0903	0.7790
	DFM	E2E	0.0826	**0.7540**	0.0967	0.8762
	RSAP-DFM(wo-Jump)	Bilevel	0.0767	0.6555	0.0970	0.8453
	RSAP-DFM(wo-Load)	Bilevel	0.0796	0.6932	0.0948	0.8081
	RSAP-DFM	Bilevel	**0.0854**	0.7317	**0.1025**	**0.8770**

Table 3: Ablation Study

method, RSAP-DFM based on dual regime shifting can better capture macro hidden information. 2) RSAP-DFM better encodes market information extracted from stock features and dynamically injects it into the dual dynamic factor model, providing the model with more robustness and accuracy (**Q3**).

5.4 Gradient-Based Posterior Factor Construction Is Better

Considering the large number of stocks in the real market, we follow the approach by [Gu *et al.*, 2021; Duan *et al.*, 2022] and use the portfolio returns instead of the returns of individual stocks, thus building return-based posterior factors.

The steps of this method are as follows: First, construct P portfolios, and the stock weights of each portfolio are dynamically redistributed according to the stocks' latent features $\boldsymbol{e}$. Second, introduce $\boldsymbol{r}$ and calculate the return of the portfolios $\boldsymbol{R}$. Third, dynamically map $\boldsymbol{R}$ to K Gaussian distributions through a neural network. Last, extract posterior factors λ_R from each of K Gaussian distributions:

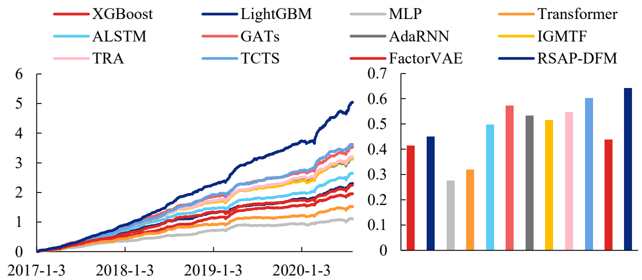
$$\begin{aligned}
&\boldsymbol{w}_p = \text{Softmax}(\text{Linear}(\boldsymbol{e}, \text{P})),\\
&\boldsymbol{R} = \boldsymbol{w}_\text{p}^\text{T}\boldsymbol{r},\\
&\boldsymbol{\mu}_\text{post} = \boldsymbol{w}_\text{post}\boldsymbol{R} + \boldsymbol{b}_\text{post},\\
&\boldsymbol{\sigma}_\text{post} = \log(1+\exp(\boldsymbol{w}_\text{post}\boldsymbol{R} + \boldsymbol{b}_\text{post})),\\
&\boldsymbol{\lambda}_R \sim \mathcal{N}(\boldsymbol{\mu}_\text{post}, \text{diag}(\boldsymbol{\sigma}^2_\text{post})),
\end{aligned} \tag{20}$$

where $\sum_{i=1}^N \boldsymbol{w}_\text{p}^{(i,j)} = 1$, $\boldsymbol{w}_\text{p} \in \mathbb{R}^{N\times P}$ denotes stock weight of P portfolios, $\boldsymbol{w}_\text{p}^{(i,j)}$ denotes the weight of i-th stock in j-th portfolio, $\boldsymbol{R} \in \mathbb{R}^P$ denotes the returns of P portfolios.

Based on $\boldsymbol{\lambda}_G$ and $\boldsymbol{\lambda}_R$, we use three other auxiliary tasks as baselines to verify the superiority of our auxiliary tasks:

$$\begin{aligned}
\ell_A^1 &= \text{KL}(P_{\phi_\text{prior}}(\lambda|\boldsymbol{x}), P_{\phi_\text{post}}(\boldsymbol{\lambda}_R|\boldsymbol{x},\boldsymbol{r})),\\
\ell_A^2 &= \text{KL}(P_{\phi_\text{prior}}(\lambda|\boldsymbol{x}), P_{\phi_\text{post}}(\boldsymbol{\lambda}_R|\boldsymbol{x},\boldsymbol{r}))\\
&\quad + \text{MSE}\left(\phi_\text{DDF}(\boldsymbol{e},\boldsymbol{m},\boldsymbol{\lambda}_R),\boldsymbol{r}\right),\\
\ell_A^3 &= \text{KL}(P_{\phi_\text{prior}}(\lambda|\boldsymbol{x}), P_{\phi_\text{post}}(\boldsymbol{\lambda}_R|\boldsymbol{x},\boldsymbol{r}))\\
&\quad + \text{MSE}\left(\phi_\text{DDF}(\boldsymbol{e},\boldsymbol{m},\boldsymbol{\lambda}_R),\boldsymbol{r}\right) + \text{MSE}\left(\phi_\text{DDF}(\boldsymbol{e},\boldsymbol{m},\boldsymbol{\lambda}),\boldsymbol{r}\right).
\end{aligned} \tag{21}$$

Table 4 presents a comparative prediction analysis of the four models. Compared with the existing and popular posterior factor construction methods, our new gradient-based posterior factor can steadily bring superior performance (**Q4**).



(a) Cumulative Returns (b) Sharpe Ratios

Figure 2: Average Investment Performance

DataSet	Model	IC	ICIR	RANK IC	RANK ICIR
CSI100	RSAP-DFM-ℓ^1	0.0672	0.3788	0.0734	0.4203
	RSAP-DFM-ℓ^2	0.0737	0.3980	0.0843	0.4634
	RSAP-DFM-ℓ^3	0.0645	0.3432	0.0802	0.4394
	RSAP-DFM	**0.0768**	**0.4260**	**0.0870**	**0.4852**
CSI300	RSAP-DFM-ℓ^1	0.0757	0.5440	0.0892	0.6466
	RSAP-DFM-ℓ^2	0.0736	0.5298	0.0882	0.6292
	RSAP-DFM-ℓ^3	0.0689	0.4741	0.0838	0.5820
	RSAP-DFM	**0.0855**	**0.6194**	**0.0961**	**0.6849**
CSI500	RSAP-DFM-ℓ^1	0.0685	0.5934	0.0878	0.7538
	RSAP-DFM-ℓ^2	0.0760	0.6486	0.0952	0.8133
	RSAP-DFM-ℓ^3	0.0839	**0.7613**	0.1008	**0.9271**
	RSAP-DFM	**0.0854**	0.7317	**0.1025**	0.8770

Table 4: Posterior Comparison

5.5 Investment Performance

We adopt the most common investment strategy named *Long/Short Strategy* to demonstrate our profitability. We rank prediction scores from highest to lowest and divide them into five groups, where the first group is assigned to long positions and the fifth group to short positions, with adjustments made each trading day. Our model significantly outperforms the baselines in terms of average cumulative long-short return and Sharpe ratio across three datasets in Figure 2 (**Q5**).

6 Conclusion

In this paper, we present RSAP-DFM, a novel adaptive dual dynamic factor model with regime-shifting and bilevel optimization, which can adaptively extract macroeconomic information into the dynamic factor model in a regime-shifting manner to improve model performance. The explicit macroeconomic information mapping method improves the interpretability of our model and provides a new perspective for future research. Our model effectively exploits the hidden features of stock time-series data with the relationship reconstruction between factor construction, factor return and factor exposure. We have also verified the effectiveness of our model in the real stock market. In the future, we plan to further verify the universality of our proposed dual regime shifting method and apply it to other methods.

Acknowledgments

Rujun Jiang is partly supported by NSFC 12171100 and Natural Science Foundation of Shanghai 22ZR1405100.

References

[Carhart, 1997] Mark M Carhart. On persistence in mutual fund performance. *The Journal of finance*, 52(1):57–82, 1997.

[Chen and Guestrin, 2016] Tianqi Chen and Carlos Guestrin. Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*, pages 785–794, 2016.

[Ding *et al.*, 2020] Qianggang Ding, Sifan Wu, Hao Sun, Jiadong Guo, and Jian Guo. Hierarchical multi-scale gaussian transformer for stock movement prediction. In *IJCAI*, pages 4640–4646, 2020.

[Du *et al.*, 2021] Yuntao Du, Jindong Wang, Wenjie Feng, Sinno Pan, Tao Qin, Renjun Xu, and Chongjun Wang. Adarnn: Adaptive learning and forecasting of time series. In *Proceedings of the 30th ACM international conference on information & knowledge management*, pages 402–411, 2021.

[Duan *et al.*, 2022] Yitong Duan, Lei Wang, Qizhong Zhang, and Jian Li. 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, pages 4468–4476, 2022.

[Fama and French, 1992] Eugene F Fama and Kenneth R French. The cross-section of expected stock returns. *the Journal of Finance*, 47(2):427–465, 1992.

[Geweke, 1977] John Geweke. The dynamic factor analysis of economic time series. *Latent variables in socio-economic models*, 1977.

[Goodfellow *et al.*, 2014] Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*, 2014.

[Gu *et al.*, 2021] Shihao Gu, Bryan Kelly, and Dacheng Xiu. Autoencoder asset pricing models. *Journal of Econometrics*, 222(1):429–450, 2021.

[Hochreiter and Schmidhuber, 1997] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[Ke *et al.*, 2017] Guolin Ke, Qi Meng, Thomas Finley, Taifeng Wang, Wei Chen, Weidong Ma, Qiwei Ye, and Tie-Yan Liu. Lightgbm: A highly efficient gradient boosting decision tree. *Advances in neural information processing systems*, 30, 2017.

[Kelly *et al.*, 2019] Bryan T Kelly, Seth Pruitt, and Yinan Su. Characteristics are covariances: A unified model of risk and return. *Journal of Financial Economics*, 134(3):501–524, 2019.

[Krizhevsky *et al.*, 2012] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems*, 25, 2012.

[Lin *et al.*, 2021a] Hengxu Lin, Dong Zhou, Weiqing Liu, and Jiang Bian. Deep risk model: a deep learning solution for mining latent risk factors to improve covariance matrix estimation. In *Proceedings of the Second ACM International Conference on AI in Finance*, pages 1–8, 2021.

[Lin *et al.*, 2021b] Hengxu Lin, Dong Zhou, Weiqing Liu, and Jiang Bian. Learning multiple stock trading patterns with temporal routing adaptor and optimal transport. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*, pages 1017–1026, 2021.

[Mari and Mari, 2023] Carlo Mari and Emiliano Mari. Deep learning based regime-switching models of energy commodity prices. *Energy Systems*, 14(4):913–934, 2023.

[Nelson *et al.*, 2017] David MQ Nelson, Adriano CM Pereira, and Renato A De Oliveira. Stock market's price movement prediction with lstm neural networks. In *2017 International joint conference on neural networks (IJCNN)*, pages 1419–1426. Ieee, 2017.

[Qin *et al.*, 2017] Yao Qin, Dongjin Song, Haifeng Chen, Wei Cheng, Guofei Jiang, and Garrison W. Cottrell. A dual-stage attention-based recurrent neural network for time series prediction. In *IJCAI*, pages 2627–2633, 2017.

[Ross, 2013] Stephen A Ross. The arbitrage theory of capital asset pricing. In *Handbook of the fundamentals of financial decision making: Part I*, pages 11–30. World Scientific, 2013.

[Rumelhart *et al.*, 1986] David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams. Learning representations by back-propagating errors. *nature*, 323(6088):533–536, 1986.

[Sawhney *et al.*, 2021] Ramit Sawhney, Shivam Agarwal, Arnav Wadhwa, Tyler Derr, and Rajiv Ratn Shah. Stock selection via spatiotemporal hypergraph attention network: A learning to rank approach. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 497–504, 2021.

[Selvin *et al.*, 2017] Sreelekshmy Selvin, R Vinayakumar, EA Gopalakrishnan, Vijay Krishna Menon, and KP Soman. Stock price prediction using lstm, rnn and cnn-sliding window model. In *2017 international conference on advances in computing, communications and informatics (icacci)*, pages 1643–1647. IEEE, 2017.

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

[Stock and Watson, 2011] James H Stock and Mark W Watson. Dynamic factor models. 2011.

[Sutskever *et al.*, 2014] Ilya Sutskever, Oriol Vinyals, and Quoc V Le. Sequence to sequence learning with neural networks. *Advances in neural information processing systems*, 27, 2014.

[Uddin and Yu, 2020] Ajim Uddin and Dantong Yu. Latent factor model for asset pricing. *Journal of Behavioral and Experimental Finance*, 27:100353, 2020.

[Velickovic *et al.*, 2017] Petar Velickovic, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, Yoshua Bengio, et al. Graph attention networks. *stat*, 1050(20):10–48550, 2017.

[Wang *et al.*, 2022] Heyuan Wang, Tengjiao Wang, Shun Li, Jiayi Zheng, Shijie Guan, and Wei Chen. Adaptive long-short pattern transformer for stock investment selection. In *IJCAI*, pages 3970–3977, 2022.

[Wei *et al.*, 2022] Zikai Wei, Bo Dai, and Dahua Lin. Factor investing with a deep multi-factor model. *arXiv preprint arXiv:2210.12462*, 2022.

[Wei *et al.*, 2023] Zikai Wei, Anyi Rao, Bo Dai, and Dahua Lin. Hirevae: An online and adaptive factor model based on hierarchical and regime-switch vae. In *IJCAI*, page 4903–4911, 2023.

[Wu *et al.*, 2021] Xueqing Wu, Lewen Wang, Yingce Xia, Weiqing Liu, Lijun Wu, Shufang Xie, Tao Qin, and Tie-Yan Liu. Temporally correlated task scheduling for sequence learning. In *International Conference on Machine Learning*, pages 11274–11284. PMLR, 2021.

[Xu *et al.*, 2021a] Wentao Xu, Weiqing Liu, Jiang Bian, Jian Yin, and Tie-Yan Liu. Instance-wise graph-based framework for multivariate time series forecasting. *arXiv preprint arXiv:2109.06489*, 2021.

[Xu *et al.*, 2021b] Wentao Xu, Weiqing Liu, Lewen Wang, Yingce Xia, Jiang Bian, Jian Yin, and Tie-Yan Liu. Hist: A graph-based framework for stock trend forecasting via mining concept-oriented shared information. *arXiv preprint arXiv:2110.13716*, 2021.

[Yang *et al.*, 2020] Xiao Yang, Weiqing Liu, Dong Zhou, Jiang Bian, and Tie-Yan Liu. Qlib: An ai-oriented quantitative investment platform. *arXiv preprint arXiv:2009.11189*, 2020.

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