FactorVAE: A Probabilistic Dynamic Factor Model Based on Variational Autoencoder for Predicting Cross-Sectional Stock Returns

Yitong Duan, Lei Wang, Qizhong Zhang, Jian Li

Tsinghua University
{dyt19, wanglei20, zhangqz18, lijian83}@mails.tsinghua.edu.cn

Abstract

As an asset pricing model in economics and finance, factor model has been widely used in quantitative investment. Towards building more effective factor models, recent years have witnessed the paradigm shift from linear models to more flexible nonlinear data-driven machine learning models. However, due to low signal-to-noise ratio of the financial data, it is quite challenging to learn effective factor models. In this paper, we propose a novel factor model, FactorVAE, as a probabilistic model with inherent randomness for noise modeling. Essentially, our model integrates the dynamic factor model (DFM) with the variational autoencoder (VAE) in machine learning, and we propose a prior-posterior learning method based on VAE, which can effectively guide the learning of model by approximating an optimal posterior factor model with future information. Particularly, considering that risk modeling is important for the noisy stock data, FactorVAE can estimate the variances from the distribution over the latent space of VAE, in addition to predicting returns. The experiments on the real stock market data demonstrate the effectiveness of FactorVAE, which outperforms various baseline methods.

Introduction

Stock investors attempt to predict cross-sectional stock returns to construct their stock portfolios that outperform the average performance of the market consistently. As a widely employed cross-section analysis method in economics and finance, factor model, with a very profound influence in academia and industry, has shown the capacity to predict the returns of cross-sectional stocks (Daniel, Hirshleifer, and Sun 2020; Fama and French 2021). Hence, establishing an effective factor model for the real market is of great importance in stock investment.

Factor models explain market phenomena and asset returns by various factors, which can be fundamental, technical, macroeconomic, and so on. Specifically, in factor models, stock returns are described by factors and corresponding exposure to factors (which means the impact of factors over stocks), and computed by the linear combination of factors. According to whether the factor exposure varies with time, factor models fall under two categories: static models and dynamic models, and recent research (Fama and French 2020) indicates that dynamic models with time-varying factor exposure achieve better asset pricing performance than static methods, so dynamic models become increasingly popular. However, the traditional dynamic models adopt the factors designed by practical experience (for example, the *momentum* factor is designed based on the observation that the stocks with higher returns in the past will also perform better in the future), which may introduce model bias because of the inconsistency between prior knowledge and real market.

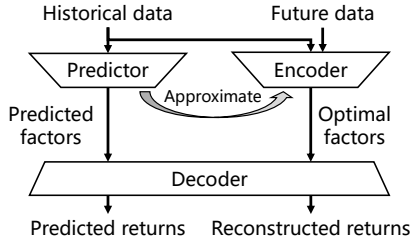

Figure 1: Brief illustration of FactorVAE

The recent advancement of machine learning (ML) offers a new data-driven perspective to dynamic factor models (Karolyi and Van Nieuwerburgh 2020). Due to the superior capacity to capture complex patterns from the market data, some ML solutions (Kelly, Pruitt, and Su 2019; Uddin and Yu 2020; Gu, Kelly, and Xiu 2021) can automatically extract latent factors from the market data, which are more effective and practical than traditional methods in the real market. Nonetheless, existing ML solutions may suffer from a vital issue, i.e., the low signal-to-noise ratio of stock data. The largely noisy data will interfere with the learning of ML-based models, and result in poor effectiveness of latent factors extracted by models. Such an issue indeed places a barrier for obtaining an effective factor model for predicting cross-sectional returns.

To break this barrier, we propose a novel probabilistic dynamic factor model based on variational autoencoder (VAE), called FactorVAE, to bridge the gap between the noisy data and effective factors. Essentially, we regard factors as the latent random variables in VAE, to model the noise in data by the distribution over the latent space of VAE, and then

introduce a prior-posterior learning method to guide the extraction of effective factors for predicting cross-sectional returns. More concretely, as shown in Figure 1, we first adopt an encoder-decoder architecture, with access to future stock returns, to extract optimal factors for reconstructing returns, and then train a predictor, only given observable historical data, to predict factors to approximate the optimal factors. In prediction phase, only the predictor and the decoder would be utilized without any future information leakage. Also note that, our model calculates stock returns by the factors with randomness, which derives a probabilistic model for estimating risk in addition to predicting returns.

The contributions of our paper are as follows:

- We propose FactorVAE as a dynamic factor model to extract effective factors from noisy market data, and we design a prior-posterior learning method based on VAE to further guide the learning of model in the highly noisy market data.
- To the best of our knowledge, we are the first to treat factors as the latent random variables in VAE, which enhances the capacity of modeling noisy data, and derives a probabilistic model for risk estimation.
- We conduct extensive experiments on the real stock market data, and the results show that our model surpasses not only other dynamic factor models, but also ML-based prediction models on cross-sectional returns prediction.

Related Work

Factor Model

Factor models can be classified into two categories, static models and dynamic models. In static factor models, the factor exposure of stock is time-invariant. The original static factor model is the capital asset pricing model (CAPM) (Treynor 1961; Sharpe 1964; Lintner 1975), which proposes the *market* factor and attributes the difference between stock returns to the different exposure to the *market* factor. Later in a seminal work, observing that the firm value and size contribute to explaining the difference in stock expected returns, (Eugene and French 1992) expands the CAPM by adding the *size* and *value* risk factor, and proposes the famous Fama-French three-factor model.

In dynamic factor models, factor exposure varies with time, and is usually calculated from the firm or asset characteristics (such as market capitalization, book-to-market ratio, asset liquidity) . (Kelly, Pruitt, and Su 2019) introduces the instrumented principal components analysis into factor model, in which factors exposure depends on the observable asset characteristics partially and has a linear relationship with them. Further, (Gu, Kelly, and Xiu 2021) proposes a latent dynamic factor asset pricing model with a conditional autoencoder network, to model the non-linearity in the return dynamics (Bansal and Yaron 2004; He and Krishnamurthy 2013), and shows that the non-linear factor model achieve better performance than other leading linear methods. Nevertheless, these works face the challenges of learning from the highly noisy market data, neglecting noise may result in the poor effectiveness of models. Hence our work dedicates to address the problem of learning from noisy data.

Stock Prediction with Machine Learning

During recent years, many studies on predicting stock returns based on machine learning have emerged. Depending on the type of data, these methods are mainly classified into two areas: alternative methods and technical methods. Alternative methods predict stock returns based on diversified alternative data, such as news texts (Hu et al. 2018), social media information (Xu and Cohen 2018) and knowledge graphs (Cheng et al. 2020). (Chen et al. 2019) incorporates the fine-grained new events into stock movement prediction, and (Chen, Wei, and Huang 2018) constructs a financial knowledge graph based on raw news texts for stock price prediction. Unlike alternative methods, technical methods only focus on the market data (mainly stock price and volume and derived features). Among them, (Qin et al. 2017) proposes a dual-stage attention-based recurrent neural network to capture long-term temporal dependencies in stock prediction. (Zhang, Aggarwal, and Qi 2017) proposes a variant of LSTM, which decomposes the hidden states of memory cells into multiple frequency components to capture the trading patterns. (Zhang et al. 2020) proposes an ensemble framework based on sample reweighting and feature selection for financial market prediction. (Ding et al. 2020) adopts a method based on Transformer (Vaswani et al. 2017) to tackle the stock movement prediction task, and shows the power of mining long-term financial time series.

Variational Autoencoder

This work is also related to variational autoencoder (VAE) (Kingma and Welling 2013), VAE is a mainstream family of deep generative models, which describes high-dimensional observation by probability distribution in low-dimensional latent space, and its variants have been widely used in various applications (Miao, Yu, and Blunsom 2016; Vahdat and Kautz 2020). (Chung et al. 2015) first proposes a generative model for sequential data by combining a recurrent neural network with the elements of VAE, and (Fraccaro et al. 2016) further improves the sequential neural generative model by integrating a deterministic recurrent neural network with a state space model. In financial applications, (Luo et al. 2018) proposes a stochastic volatility models based on (Chung et al. 2015; Fraccaro et al. 2016) , to better estimate temporal dynamics of stock volatility. (Xu and Cohen 2018) presents a VAE-based model jointly exploiting social media text and price signals for stock movement prediction.

Preliminaries

In this section, we first formally define the notations and describe the problem, and then introduce variational autoencoder briefly.

Problem Formulation

In this paper, we use lower-case letters (*e.g.*, h) to denote vectors or matrices, and capital letters (*e.g.*, N) to denote scalars, if not otherwise specified. In addition, all w and b without definition represent to the weight and bias of linear layers, which will not be described later for simplicity.

First, we formally introduce the dynamic factor model (DFM). According to (Ng, Engle, and Rothschild 1992), the general functional form of DFM is formulated as

$$y_s = \alpha_s + \sum_{k=1}^{K} \beta_s^{(k)} z_s^{(k)} + \epsilon_s \tag{1}$$

where $y_s = \frac{\text{price}_{s+1} - \text{price}_s}{\text{price}_s} \in \mathbb{R}^{N_s}$ denotes future returns of N_s stocks in cross-section at time step s, $\alpha_s \in \mathbb{R}^{N_s}$ is the vector of stock idiosyncratic returns, $\beta_s \in \mathbb{R}^{N_s \times K}$ is factor exposure matrix. $z_s \in \mathbb{R}^K$ is the vector of K factors, ϵ_s is the idiosyncratic noises with zero mean.

The formulation of the task is to learn a dynamic factor model with parameter Θ, for predicting future cross-sectional returns from historical data.

$$\hat{y}_s = f(x_s; \Theta) = \alpha(x_s) + \beta(x_s) z(x_s) \tag{2}$$

where $x_s \in \mathbb{R}^{N_s \times T \times C}$ is the historical stock characteristics (such as volatility, liquidity) of past T time-steps, N_s is the number of stocks in cross-section at time step s (we only consider the stocks that exist in cross-section at all T time steps), C is the number of characteristics.

We formally define the problem as:

Input: A set of samples $\{(x_s, y_s)\}$, where $x_s \in \mathbb{R}^{N_s \times T \times C}$ is the sequential characteristics of stocks, and $y_s \in \mathbb{R}^{N_s}$ is the future returns of cross-sectional stocks.

Output: A dynamic factor model as Equation 2, which outputs the prediction returns $\hat{y}_s$.

Variational Autoencoder

As a generative model based on dimension reduction, variational autoencoder (VAE) (Kingma and Welling 2013) follows an encoder-decoder architecture, and generates the high-dimensional data from low-dimensional latent space, as shown in Figure 2.

We assume that observation data x can be generated from a latent random variable $\mathbf{z}$, and use an encoder with parameter ϕ to describe the posterior distribution of z given x, denoted as $q_\phi(z|x)$, then use a decoder with parameter θ to generate the reconstructed observation data x' from z_{post}, where z_{post} is sampled from $q_\phi(z|x)$. Meanwhile, $q_\phi(z|x)$ is enforced to close to a given prior distribution $p(z)$ (such as a standard Gaussian distribution). In the generation phase, VAE generate a new observation from the z_{prior}, where z_{prior} is sampled from $p(z)$. Formally, the objective function of

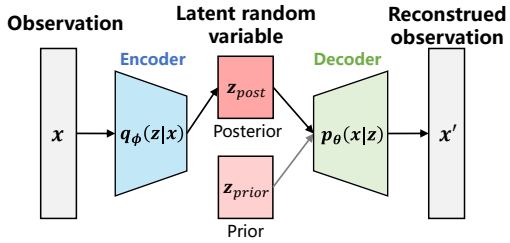

Figure 2: The architecture of variational autoencoder

VAE is

$$\max_{\theta,\phi} \{\mathbb{E}_{z \sim q_\phi(z|x)} [\ln p_\theta(x|z)] - \text{KL}[q_\phi(z|x) \| p(z)]\} \tag{3}$$

where the first term is to reduce the reconstruction error (by maximizing the expected log-likelihood), and the second term is to keep the posterior distribution close to the prior distribution (by minimizing the Kullback–Leibler divergence (KLD) between $q_\phi(z \mid x)$ and $p(z)$).

Methodology

In this section, we establish FactorVAE to extract effective factors from the noisy market data. First, we obtain optimal factors by an encoder-decoder architecture with access to future data, and then train a factor predictor according a prior-posterior learning method, which extracts factors to approximate the optimal factors. The overall framework of model is summarized in Figure 3.

Encoder-Decoder Architecture

Our model follows the encoder-decoder architecture of VAE, to learn an optimal factor model, which can reconstruct the cross-sectional stock returns by several factors well. As shown in Figure 3, with access to future stock returns, the encoder plays a role as an oracle, which can extract optimal factors from future data, called posterior factors, and then the decoder reconstructs future stock returns by the posterior factors. Specially, the factors in our model are regarded as the latent variables in VAE, with the capacity of modeling noisy data.

Concretely, this architecture contains three components: feature extractor, factor encoder and factor decoder.

Feature Extractor Feature extractor extracts stocks latent features e from the historical sequential characteristics x, formulated as $e = \phi_{\text{feat}}(x)$. In order to capture the temporal dependence in sequences, we adopt the Gate Recurrent Unit (GRU), a variant of RNN (Chung et al. 2014). At time-step t, it performs as:

$$\begin{aligned} h_{\text{proj}}^{(i,t)} &= \text{LeakyReLU}\left(w_{\text{proj}} x^{(i,t)} + b_{\text{proj}}\right) \\ h_{\text{gru}}^{(i,t)} &= \text{GRU}\left(h_{\text{proj}}^{(i,t)}, h_{\text{gru}}^{(i,t-1)}\right) \end{aligned} \tag{4}$$

where $x^{(i,t)} \in \mathbb{R}^C$ is the characteristics of *i-th* stock at time step t, $h_{\text{proj}}, h_{\text{gru}}^{(i,t)} \in \mathbb{R}^H$ are the hidden states with dimension H, and $\text{LeakyReLU}(x) = \begin{cases} x, & \text{if } x \geq 0 \\ \zeta x, & \text{otherwise} \end{cases}$, ζ is negative slope.

Finally, we use the hidden state of GRU at last time step T as the latent features of stocks, i.e., $e = h_{\text{gru}}^{(T)}$.

Factor Encoder Factor encoder extracts posterior factors $\mathbf{z}_{\text{post}}$ from the future stock returns y and the latent features e

$$\begin{aligned} [\mu_{\text{post}}, \sigma_{\text{post}}] &= \phi_{\text{enc}}(y, e) \\ \mathbf{z}_{\text{post}} &\sim \mathcal{N}\left(\mu_{\text{post}}, \text{diag}\left(\sigma_{\text{post}}^2\right)\right) \end{aligned} \tag{5}$$

where $\mathbf{z}_{\text{post}}$ is a random vector following the independent Gaussian distribution, which can be described by the mean

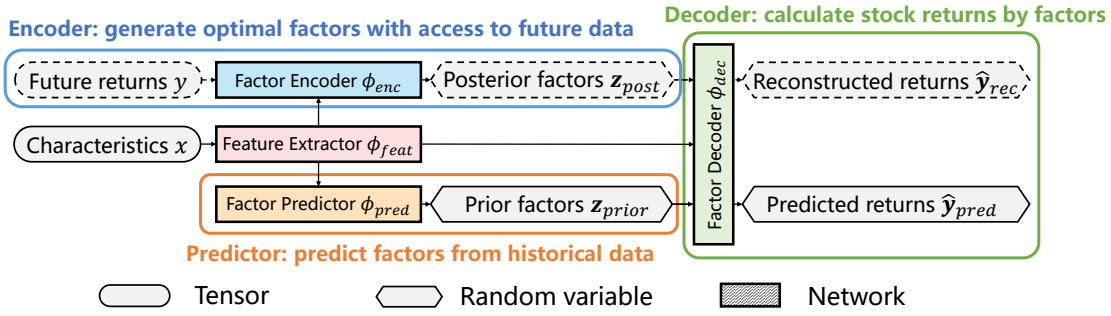

Figure 3: The overall framework of FactorVAE. All the modules with dotted lines, involving the future data, are only used in the training phase, and would be removed in the test phase or in prediction.

$\mu_{\text{post}} \in \mathbb{R}^K$ and the standard deviation (std) $\sigma_{\text{post}} \in \mathbb{R}^K$, K is the number of factors.

The architecture of ϕ_{enc} is shown in Figure 4(a). Because the number of individual stocks in cross-section is large and varies with time, instead of using stock returns y directly, we construct a set of portfolios inspired by (Gu, Kelly, and Xiu 2021), these portfolios are dynamically re-weighted on the basis of stock latent features, i.e., $y_p = y \cdot \varphi_p(e) = y \cdot a_{\text{p}}$, where $a_{\text{p}} \in \mathbb{R}^M$ denotes the weight of M portfolios. Formally,

$$a_p^{(i,j)} = \frac{\exp\left(w_p e^{(i)} + b_p\right)^{(j)}}{\sum_{i=1}^{N} \exp\left(w_p e^{(i)} + b_p\right)^{(j)}} \tag{6}$$

$$y_{\text{p}}^{(j)} = \sum_{i=1}^{N} y^{(i)} a_{\text{p}}^{(i,j)} \tag{7}$$

where $a_p^{(i,j)}$ denotes the weight of *i-th* stock in *j-th* portfolio and meets $\sum_{i=1}^{N} a_{\text{p}}^{(i,j)} = 1$, $y_{\text{p}} \in \mathbb{R}^M$ is the vector of portfolio returns. The main advantages of constructing portfolios lie in: 1) reducing the input dimension and avoiding too many parameters. 2) robust to the missing stocks in cross-section and thus suitable for the market (see Experiment 2).

And then the mean and the std of posterior factors are output by a mapping layer $[\mu_{\text{post}}, \sigma_{\text{post}}] = \varphi_{\text{map}}(y_{\text{p}})$, that is

$$\begin{aligned} \mu_{\text{post}} &= w_{\text{post}_\mu} y_{\text{p}} + b_{\text{post}_\mu} \\ \sigma_{\text{post}} &= \text{Softplus}(w_{\text{post}_\sigma} y_{\text{p}} + b_{\text{post}_\sigma}) \end{aligned} \tag{8}$$

where $\text{Softplus}(x) = \log(1 + \exp(x))$

Factor Decoder Factor decoder uses factors $\mathbf{z}$ and the latent feature e to calculate stock returns $\hat{\mathbf{y}}$

$$\hat{\mathbf{y}} = \phi_{\text{dec}}(\mathbf{z}, e) = \boldsymbol{\alpha} + \beta \mathbf{z} \tag{9}$$

Essentially, the decoder network ϕ_{dec} consists of alpha layer and beta layer, as shown in Figure 4(b).

Alpha layer outputs idiosyncratic returns $\boldsymbol{\alpha}$ from the latent features e. We assume that $\boldsymbol{\alpha}$ is a Gaussian random vector described by $\boldsymbol{\alpha} \sim \mathcal{N}\left(\mu_\alpha, \text{diag}\left(\sigma_\alpha^2\right)\right)$, where the mean $\mu_\alpha \in \mathbb{R}^N$ and the std $\sigma_\alpha \in \mathbb{R}^N$ are output by a distribution

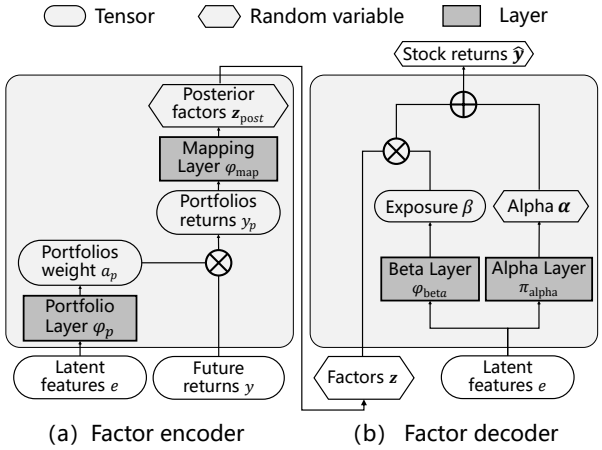

Figure 4: The encoder-decoder architecture of FactorVAE. (a) Factor encoder extracts posterior factors with access to future data, (b) Factor decoder reconstructs stock returns from the posterior factors

network π_{alpha}, i.e., $[\mu_\alpha, \sigma_\alpha] = \pi_{\text{alpha}}(e)$. Specifically,

$$\begin{aligned} h_\alpha^{(i)} &= \text{LeakyReLU}\left(w_\alpha e^{(i)} + b_\alpha\right) \\ \mu_\alpha^{(i)} &= w_{\alpha_\mu} h_\alpha^{(i)} + b_{\alpha_\mu} \\ \sigma_\alpha^{(i)} &= \text{Softplus}(w_{\alpha_\sigma} h_\alpha^{(i)} + b_{\alpha_\sigma}) \end{aligned} \tag{10}$$

where $h_\alpha^{(i)} \in \mathbb{R}^H$ is the hidden state.

Beta layer calculates factor exposure $\beta \in \mathbb{R}^{N \times K}$ from the latent features e by linear mapping. Formally,

$$\beta^{(i)} = \varphi_{\text{beta}}(e^{(i)}) = w_\beta e^{(i)} + b_\beta \tag{11}$$

Note that $\boldsymbol{\alpha}$ and $\mathbf{z}$ are both follow independent Gaussian distribution, and thus the output of decoder $\hat{\mathbf{y}}^{(i)} \sim \mathcal{N}\left(\mu_y^{(i)}, {\sigma_y^{(i)}}^2\right)$, where

$$\begin{aligned} \mu_y^{(i)} &= \mu_\alpha^{(i)} + \sum_{k=1}^{K} \beta^{(i,k)} \mu_z^{(k)} \\ \sigma_y^{(i)} &= \left({\sigma_\alpha^{(i)}}^2 + \sum_{k=1}^{K} {\beta^{(i,k)}}^2 {\sigma_z^{(k)}}^2\right)^{\frac{1}{2}} \end{aligned} \tag{12}$$

where $\mu_z, \sigma_z \in \mathbb{R}^K$ are the mean and the std of factors respectively.

Prior-Posterior Learning

As mentioned before, our goal is to bridge the gap between the noisy market data and an effective factor model for predicting returns. The model trained in an end-to-end method may not extract effective factors from the noisy data. Therefore, we propose a prior-posterior learning method based on VAE to fulfill this goal: train a factor predictor only given the historical observation data, which predicts factors to approximate the optimal posterior factors above, called prior factors. Then we use the factor decoder to calculate the stock returns by the prior factors without any future information leakage, as the predicted returns of model.

Factor Predictor Factor predictor extracts prior factors $\mathbf{z}_{\text{prior}}$ from the stock latent features e:

$$\begin{aligned} [\mu_{\text{prior}}, \sigma_{\text{prior}}] &= \phi_{\text{pred}}(e) \\ \mathbf{z}_{\text{prior}} &\sim \mathcal{N}\left(\mu_{\text{prior}}, \text{diag}\left(\sigma^2_{\text{prior}}\right)\right) \end{aligned} \tag{13}$$

where $\mathbf{z}_{\text{prior}}$ is a Gaussian random vector, described by the mean $\mu_{\text{prior}} \in \mathbb{R}^K$ and the std $\sigma_{\text{prior}} \in \mathbb{R}^K$.

Considering that a factor usually represents a certain type of risk premium in the market (such as the *size* factor focuses on the risk premium of small-cap stocks), we design a muti-head global attention mechanism to integrate the diverse global representations of the market in parallel, and extract factors from them to represent diverse risk premium of market, as shown in Figure 5. Formally, a single-head attention performs as

$$\begin{aligned} k^{(i)} &= w_{\text{key}} e^{(i)}, v^{(i)} = w_{\text{value}} e^{(i)} \\ a^{(i)}_{att} &= \frac{\max\left(0, \frac{q {k^{(i)}}^T}{\|q\|_2 \cdot \left\|k^{(i)}\right\|_2}\right)}{\sum_{i=1}^{N} \max\left(0, \frac{q {k^{(i)}}^T}{\|q\|_2 \cdot \left\|k^{(i)}\right\|_2}\right)} \\ h_{\text{att}} &= \varphi_{\text{att}}(e) = \sum_{i=1}^{N} a^{(i)}_{\text{att}} v^{(i)} \end{aligned} \tag{14}$$

where query token $q \in \mathbb{R}^H$ is a learnable parameter, and $h_{\text{att}} \in \mathbb{R}^H$ is the global representation of market. The muti-head attention concatenates K independent heads together

$$h_{\text{muti}} = \text{Concat}\left([\varphi_{\text{att}_1}(e), \ldots, \varphi_{\text{att}_K}(e)]\right) \tag{15}$$

where $h_{\text{muti}} \in \mathbb{R}^{K \times H}$ is the muti-global representation.

And then we use a distribution network π_{prior} to predict the mean μ_{prior} and the std σ_{prior} of prior factors $\mathbf{z}_{\text{prior}}$, similar to Equation 10

$$[\mu_{\text{prior}}, \sigma_{\text{prior}}] = \pi_{\text{prior}}(h_{\text{muti}}) \tag{16}$$

Objective Function Our objective consists of two parts, the first part is to train an optimal posterior factor model, and the second part is to effectively guide the leaning of factor predictor by the posterior factors. Thus, the loss function of model is

$$\begin{aligned} L(x, y) = &-\frac{1}{N} \sum_{i=1}^{N} \log P_{\phi_{\text{dec}}}\left(\widehat{\boldsymbol{y}}^{(i)}_{\text{rec}} = y^{(i)} | x, \mathbf{z}_{\text{post}}\right) \\ &+ \gamma \cdot \text{KL}\left(P_{\phi_{\text{enc}}}(z|x, y), P_{\phi_{\text{pred}}}(z|x)\right) \end{aligned} \tag{17}$$

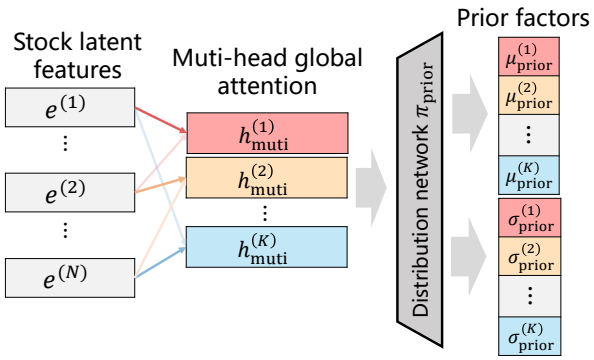

Figure 5: Illustration of factor predictor. The muti-head global attention mechanism integrates diverse global representations from the stock latent features, and the distribution network outputs the distribution of prior factors from the global representations.

where the first loss term is the negative log likelihood, to reduce the reconstruction error of posterior factor model, and $\hat{\mathbf{y}}^{(i)}_{\text{rec}} = \boldsymbol{\alpha}^{(i)} + \beta^{(i)} \mathbf{z}_{\text{post}}$ is the reconstructed return of *i-th* stock. The second loss term is the Kullback–Leibler divergence (KLD) between the distribution of prior and posterior factors, for enforcing the prior factors to approximate to the posterior factors, γ is the weight of KLD loss.

Prediction In prediction phase, the model predicts stock returns only by the predictor and the decoder, without encoder or any future information leakage. Formally,

$$\hat{\mathbf{y}}_{\text{pred}} = \phi_{\text{dec}}(\mathbf{z}_{\text{prior}}, x) = \boldsymbol{\alpha} + \beta \mathbf{z}_{\text{prior}} \tag{18}$$

where

$$\begin{aligned} e &= \phi_{\text{feat}}(x) \\ \boldsymbol{\alpha} &\sim \pi_{\text{alpha}}(e) \\ \beta &= \varphi_{\text{beta}}(e) \\ \mathbf{z}_{\text{prior}} &\sim \phi_{\text{pred}}(e) \end{aligned} \tag{19}$$

According to Equation 12, the predicted return follow Gaussian distribution $\hat{\mathbf{y}}^{(i)}_{\text{pred}} \sim \mathcal{N}\left(\mu^{(i)}_{\text{pred}}, {\sigma^{(i)}_{\text{pred}}}^2\right)$, where the mean $\mu^{(i)}_{\text{pred}}$ represents the expected return of *i-th* stock, and the std $\sigma^{(i)}_{\text{pred}}$ can be used to estimate risk, which helps the investment in the real market (see Experiment 3).

Experiments

In this section, we evaluate the proposed model on the real-world stock market data, and demonstrate the effectiveness of our model by various experiments. To lead our discussion, we first raise the following research questions:

- **RQ1:** Does the prior-posterior learning method guide the learning of model effectively?
- **RQ2:** Is our model robust to the stocks that have never been learned in the training phase?
- **RQ3:** How does the risk estimate of our model help stock investment?

Experiment Settings

Dataset We conduct the experiments over China A-shares market, the raw data is collected from Yahoo Finance, containing the day-level price-volume data of all stocks except the suspension or other abnormal stocks. We adopt *Alpha158* dataset from Qlib platform[1] (Yang et al. 2020) containing 158 technical features extracted from price-volume data, and we use 20 features selected by Qlib for the time series model. The sequence length is $T = 20$, and the daily stock return is computed by $y^{(i)} = \frac{p^{(i)}_{t+2} - p^{(i)}_{t+1}}{p^{(i)}_{t+1}}$, where t is the prediction day, and $p^{(i)}_{t+1}$ denotes the close price of *i-th* stock on the next trading day of the prediction day.

Finally, we split data into training (3432 stocks, from 01/01/2010 to 12/31/2017), validation (3450 stocks, from 01/01/2018 to 12/31/2018) and test (3923 stocks, from 01/01/2019 to 12/31/2020) datasets. The increase stock number is due to the issuance of new stocks.

Baselines We compare our model with other dynamic factor models and ML-based prediction models:

- **Linear** is a linear dynamic factor model.
- **CA** (Gu, Kelly, and Xiu 2021) is a dynamic factor model with a conditional autoencoder to extract latent factors.
- **GRU** is a neural network with a GRU layer (Chung et al. 2014) and a linear prediction layer.
- **ALSTM** (Qin et al. 2017) is a variant of LSTM, which adds an attention layer into the LSTM model to aggregate information attentively.
- **GAT** (Veličković et al. 2017) is a graph attention network, which treats stocks as the nodes on graph and predicts without knowing the graph structure upfront.
- **Trans** (Ding et al. 2020) is a neural network adopting Transformer architecture for stock returns prediction.
- **SFM** (Zhang, Aggarwal, and Qi 2017) is a RNN that uses discrete fourier transform to decompose the hidden states and capture the multi-frequency trading patterns.

Experiment 1: Cross-Sectional Returns Prediction

In this experiment, we train models to predict the future returns of cross-sectional stocks. In order to evaluate the performance of the compared methods, we adopt the rank information coefficient (*Rank IC*) as a metric, which is the dominate ranking metric in finance. Formally,

$$\textit{Rank IC}_s = \frac{1}{N_s} \frac{\left(r_{\hat{y}_s} - mean\left(r_{\hat{y}_s}\right)\right)^T \left(r_{y_t} - mean\left(r_{y_s}\right)\right)}{std\left(r_{\hat{y}_s}\right) \cdot std\left(r_{y_s}\right)} \tag{20}$$

$$\textit{Rank IC} = \frac{1}{T_{\text{test}}} \sum_{s=1}^{T_{\text{test}}} \textit{Rank IC}_s \tag{21}$$

where T_{test} is the number of trading days in test range, $r_{\hat{y}_s}, r_{y_s}$ are the predicted and true ranks of stocks in cross-section on *s-th* trading day respectively. In addition, to evaluate the stability of prediction, we also report the information ratio of *Rank IC* (*Rank ICIR*), which is calculated by dividing the average by the standard deviation of *Rank IC*$_s$.

[1] https://github.com/microsoft/qlib

Category	Method	Rank IC	Rank ICIR
ML-based prediction model	GRU	0.032(0.002)	0.398(0.031)
	ALSTM	0.031(0.004)	0.360(0.019)
	GAT	0.034(0.002)	0.390(0.032)
	Trans	0.033(0.003)	0.417(0.032)
	SFM	0.037(0.001)	0.456(0.004)
Dynamic factor model	Linear	0.022(0.002)	0.333(0.033)
	CA	0.039(0.002)	0.442(0.036)
	FactorVAE-prior	0.042(0.003)	0.384(0.033)
	FactorVAE	**0.055(0.004)**	**0.568(0.044)**

Table 1: Cross-sectional returns prediction performance of the compared methods on test dataset; the higher, the better. We report the mean and the standard deviation values of the results with 5 random seeds.

Note that FactorVAE is a probabilistic model and the predicted returns $\hat{\mathbf{y}}^{(i)}_{\text{pred}} \sim \mathcal{N}\left(\mu^{(i)}_{\text{pred}}, {\sigma^{(i)}_{\text{pred}}}^2\right)$, so we adopt the mean μ_{pred} as the predicted value.

Thorough comparison of performance on the test dataset are summarized in Table 1, from which we have the following observations:

- FactorVAE has the best performance among all the compared methods, which illustrates the effectiveness of the proposed method.
- FactorVAE-prior is a variant of our model without the prior-posterior learning method, which is trained to predict returns by prior factors directly. As we can see from the results, without the guide of posterior factors, it is hard to learn an effective factor model from the real market data, which shows that the prior-posterior learning method is critical to our model (**RQ1**).

Experiment 2: Robustness

In this experiment, we evaluate the robustness of models to the missing stocks in training dataset. Specifically, we randomly remove m stocks $\mathcal{S} = \{s^{(i_1)}, \ldots, s^{(i_m)}\}$ from the training dataset, and use the new training dataset $\mathcal{D}_{train} = \{(x_s, y_s)\}_{s \notin \mathcal{S}}$ to train models. Then we predict the stock returns on test dataset $\mathcal{D}_{test}$, and select the predicted returns of these m stocks $\{\hat{y}^{(i)}_{pred}\}_{i \in \mathcal{S}}$. Finally, we evaluate the performance of models on missing stocks by calculate the *Rank IC* and *Rank ICIR* of stock set $\mathcal{S}$ on test dataset.

Table 2 lists the results on different number of missing stocks sampled from 5 random seeds, and we observe that:

- FactorVAE is superior to other baseline methods on all m, which shows that our model is more robust to the stocks that have never been learned before (**RQ2**), and thus suitable for the situation in the real market (e.g., predict the return of newly issued stocks).
- FactorVAE-port is a variant which replaces the portfolio layer φ_p in factor encoder with the portfolio construction used in the baseline model CA. In particular, the portfolios in this variant are constructed by $y_{\text{p}} = (e^T e)^{-1} e^T y$

Methods	m=50		m=100		m=200	
	Rank IC	Rank ICIR	Rank IC	Rank ICIR	Rank IC	Rank ICIR
GRU	0.031(0.005)	0.184(0.029)	0.030(0.004)	0.234(0.030)	0.031(0.004)	0.282(0.032)
ALSTM	0.027(0.004)	0.162(0.022)	0.028(0.007)	0.210(0.045)	0.026(0.005)	0.237(0.041)
GAT	0.029(0.008)	0.166(0.043)	0.023(0.011)	0.176(0.085)	0.025(0.009)	0.215(0.071)
Trans	0.034(0.007)	0.201(0.040)	0.034(0.006)	0.259(0.043)	0.033(0.003)	0.302(0.023)
SFM	0.037(0.007)	0.220(0.042)	0.038(0.004)	0.294(0.035)	0.038(0.003)	0.342(0.036)
linear	0.018(0.005)	0.138(0.075)	0.018(0.005)	0.147(0.044)	0.018(0.004)	0.176(0.042)
CA	0.038(0.008)	0.215(0.046)	0.039(0.004)	0.284(0.034)	0.039(0.003)	0.328(0.027)
FactorVAE-port	0.043(0.005)	0.241(0.022)	0.039(0.003)	0.272(0.005)	0.041(0.004)	0.328(0.011)
FactorVAE	**0.053(0.007)**	**0.299(0.039)**	**0.056(0.002)**	**0.384(0.044)**	**0.050(0.008)**	**0.399(0.063)**

Table 2: The robustness of the compared methods.

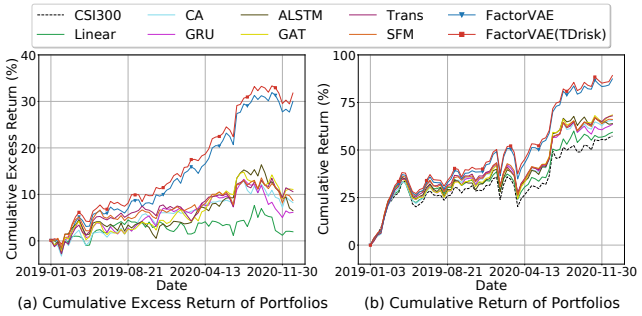

Figure 6: The performance of portfolios on the test period.

Method	AR(↑)	SR(↑)	MDD(↓)
GRU	2.28%	0.31	9.08%
ALSTM	2.20%	0.27	12.19%
GAT	4.49%	0.56	7.20%
Trans	4.79%	0.62	5.01%
SFM	3.33%	0.42	7.32%
Linear	0.01%	0.02	8.02%
CA	3.62%	0.47	7.00%
FactorVAE	15.32%	1.92	**4.47%**
FactorVAE(TDrisk)	**16.32%**	**2.09**	4.50%

Table 3: The portfolio performance relative to the benchmark. ↑ means the larger the better while ↓ means the smaller the better.

(using the same notations as above). The comparison results illustrate the effectiveness of constructing dynamically re-weighted portfolios.

Experiment 3: Portfolio Investment

In this experiment, we construct portfolios based on the prediction of models, and compare their performance through backtest. We adopt the *TopK-Drop* strategy[2], to maintain a portfolio on each trading day. Formally, on trading day t, *TopK-Drop* constructs an equal weight portfolio of k stocks $\mathcal{P}_t = \{s_t^{(i_1)}, \ldots, s_t^{(i_k)}\}$, which are selected according to the ranking of predicted returns, under the turnover constraint that the number of intersection stocks $\mathcal{P}_t \cap \mathcal{P}_{t-1} \geq k - n$. We set $k = 50$ and $n = 5$ in the experiment.

As a widely used benchmark in China A-shares market, CSI300 index consists of the 300 largest and most liquid A-share stocks, aiming to reflect the overall performance of market. Therefore, we choose CSI300 index as the benchmark, and select 50 stocks from CSI300 stocks on each trading day to construct portfolios. In the experiment, we use a strict backtest procedure to simulate the real market, in which we take into account the trading fee, stock suspension and price limit in A-share market.

The portfolio performance of the compared methods are show in Figure 6, where (a) shows the cumulative excess returns (relative to the CSI300 index) of methods, and (b) shows the cumulative returns (in terms of absolute returns) of methods. We also report the performance of portfolios by measuring the annualized return (AR), Sharpe ratio (SR), and maximum drawdown (MDD) of cumulative excess returns, which are summarized in Table 3. From the backtest results, it can be observed that:

- The portfolios based on FactorVAE outperform all the compared portfolios, which indicates our model can achieve a profitable investment in the real market.
- *TDrisk* is a variant of *TopK-Drop* considering risk aversion, which selects k stocks according to the risk-adjusted returns $\mu_{\text{pred}}^{(i)} - \eta\sigma_{\text{pred}}^{(i)}$, where η is risk aversion weight. Combined with *TDrisk*, our model further increases the AR and SR of portfolio, which shows the effectiveness of risk estimation in investment (**RQ3**).

Conclusion

In this paper, we show how to learn an effective factor model for predicting cross-sectional stock returns. Specifically, in view of the low signal-to-noise ratio of stock data, we propose a probabilistic dynamic factor model based on variational autoencoder (VAE). By treating factors as the latent random variables in VAE, the proposed model with inherent randomness can model the noisy data and estimate stock risk. In order to extract effective factors from noisy market data, we propose a prior-posterior learning method, which can guide the learning of model effectively. The experiment results over the real stock market data have demonstrated the effectiveness of our model. In the future, we plan to explore more portfolio strategies based on our model.

[2]https://qlib.readthedocs.io/en/latest/component/strategy.html

Acknowledgements

The research is supported in part by the National Natural Science Foundation of China Grant 62161146004, Turing AI Institute of Nanjing and Xi'an Institute for Interdisciplinary Information Core Technology.

References

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.

Chen, D.; Zou, Y.; Harimoto, K.; Bao, R.; Ren, X.; and Sun, X. 2019. Incorporating fine-grained events in stock movement prediction. *arXiv preprint arXiv:1910.05078*.

Chen, Y.; Wei, Z.; and Huang, X. 2018. Incorporating corporation relationship via graph convolutional neural networks for stock price prediction. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*, 1655–1658.

Cheng, D.; Yang, F.; Wang, X.; Zhang, Y.; and Zhang, L. 2020. Knowledge graph-based event embedding framework for financial quantitative investments. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, 2221–2230.

Chung, J.; Gulcehre, C.; Cho, K.; and Bengio, Y. 2014. Empirical evaluation of gated recurrent neural networks on sequence modeling. *arXiv preprint arXiv:1412.3555*.

Chung, J.; Kastner, K.; Dinh, L.; Goel, K.; Courville, A. C.; and Bengio, Y. 2015. A recurrent latent variable model for sequential data. *Advances in neural information processing systems*, 28: 2980–2988.

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

Ding, Q.; Wu, S.; Sun, H.; Guo, J.; and Guo, J. 2020. Hierarchical Multi-Scale Gaussian Transformer for Stock Movement Prediction. In *IJCAI*, 4640–4646.

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. 2020. Comparing cross-section and time-series factor models. *The Review of Financial Studies*, 33(5): 1891–1926.

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

Fraccaro, M.; Sønderby, S. K.; Paquet, U.; and Winther, O. 2016. Sequential neural models with stochastic layers. *arXiv preprint arXiv:1605.07571*.

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

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

Hu, Z.; Liu, W.; Bian, J.; Liu, X.; and Liu, T.-Y. 2018. Listening to chaotic whispers: A deep learning framework for news-oriented stock trend prediction. In *Proceedings of the eleventh ACM international conference on web search and data mining*, 261–269.

Karolyi, G. A.; and Van Nieuwerburgh, S. 2020. New methods for the cross-section of returns. *The Review of Financial Studies*, 33(5): 1879–1890.

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.

Kingma, D. P.; and Welling, M. 2013. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*.

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.

Luo, R.; Zhang, W.; Xu, X.; and Wang, J. 2018. A neural stochastic volatility model. In *Thirty-second AAAI conference on artificial intelligence*.

Miao, Y.; Yu, L.; and Blunsom, P. 2016. Neural variational inference for text processing. In *International conference on machine learning*, 1727–1736. PMLR.

Ng, V.; Engle, R. F.; and Rothschild, M. 1992. A multi-dynamic-factor model for stock returns. *Journal of Econometrics*, 52(1-2): 245–266.

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*.

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

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.

Vahdat, A.; and Kautz, J. 2020. Nvae: A deep hierarchical variational autoencoder. *arXiv preprint arXiv:2007.03898*.

Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A. N.; Kaiser, Ł.; and Polosukhin, I. 2017. Attention is all you need. In *Advances in neural information processing systems*, 5998–6008.

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

Xu, Y.; and Cohen, S. B. 2018. Stock movement prediction from tweets and historical prices. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 1970–1979.

Yang, X.; Liu, W.; Zhou, D.; Bian, J.; and Liu, T.-Y. 2020. Qlib: An AI-oriented Quantitative Investment Platform. *arXiv preprint arXiv:2009.11189*.

Zhang, C.; Li, Y.; Chen, X.; Jin, Y.; Tang, P.; and Li, J. 2020. DoubleEnsemble: A New Ensemble Method Based on Sample Reweighting and Feature Selection for Financial Data Analysis. In *2020 IEEE International Conference on Data Mining (ICDM)*, 781–790. IEEE.

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.