Learning Universal Stock Price Patterns through Return-Volatility Normalization for Accurate Stock Price Prediction

Anonymous Author(s)

Abstract

How can we learn universal stock price patterns shared by all stocks to accurately predict future stock prices? Stock price prediction technology has garnered significant attention in both academia and industry due to its ability to recognize complex patterns. Most stock price prediction techniques aim to identify universal price patterns shared by all stocks from price data to predict future prices. However, it has been a challenging task to discover universal price patterns since price data reflect unique characteristics of individual stocks.

In this paper, we propose RVN-UP (Return-Volatility Normalization for Learning Universal Stock Price Patterns), a novel framework for stock price prediction. The goal of RVN-UP is to accurately predict stock prices through learning universal price patterns. RVN-UP achieves this goal by 1) normalizing price features to remove individual stock characteristics, including return, volatility, and price scale, and 2) a stock price modeling approach based on GBM that integrates universal pattern with individual stock characteristics. Also, RVN-UP considers short-term patterns and long-term trends in stock prices simultaneously. RVN-UP achieves improvement of up to 0.0222 in information coefficient and 41.66%p in annualized rate of return compared to the previous state-of-the-art models on real-world datasets. Additionally, through universal pattern learning, RVN-UP successfully recognizes price patterns of stocks not seen in the training set, unlike existing methods.

1 Introduction

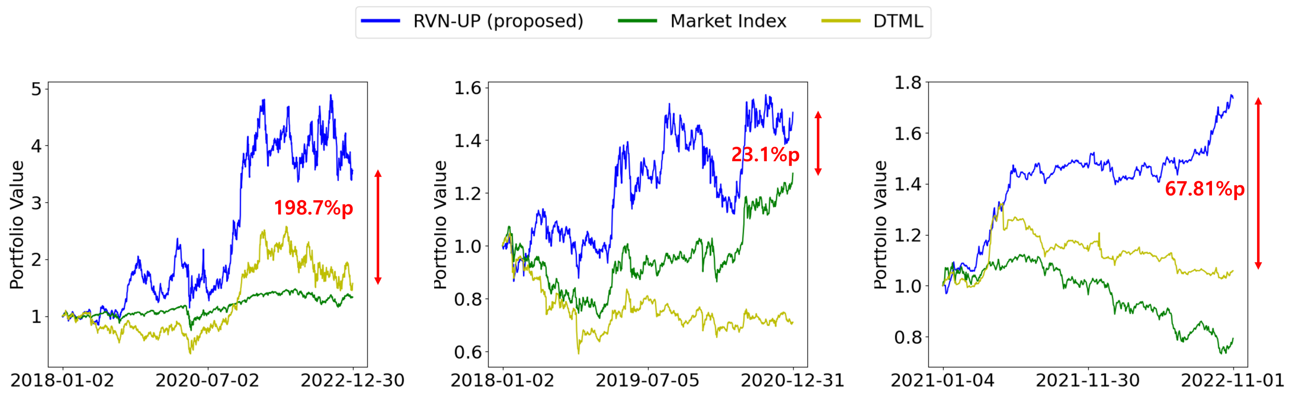
How can we learn universal stock price patterns for accurate stock price prediction? Accurate stock price prediction leads to maximized investment returns. Additionally, with recent advancements in data mining and machine learning technologies, system trading has been widely used, making accurate stock price prediction methods a topic of great interest.

Many stock price prediction techniques aim to discover universal patterns shared by all stocks in price data. However, identifying universal stock price patterns is challenging since stock price data contain unique characteristics of individual stocks. Finding universal patterns is challenging in many pattern recognition tasks due to individual sample characteristics. For example, in speech recognition, characteristics of sound samples such as speaker's voice tone make it difficult to identify universal patterns (fundamental phonemes) in speech. In this case, it is necessary to remove such characteristics from each sample to accurately recognize fundamental phonemes [4, 15]. Similarly, in stock price prediction, it is essential to remove individual stock characteristics from each sample to recognize universal stock price patterns. Previous works on stock prediction have considered only price scale as a characteristic of individual stocks for data normalization [13, 3, 16]. However, normalizing price scale does not sufficiently eliminate individual stock characteristics. Therefore, existing methods struggle to learn universal patterns in stock prices since they fail to remove characteristics of individual stocks.

In this paper, we propose RVN-UP (Return-Volatility Normalization for Learning Universal Stock Price Patterns), a method for learning universal stock price patterns by removing characteristics of each individual stock for accurate stock price prediction. The main ideas of RVN-UP are as follows. First, RVN-UP learns universal stock price patterns by removing individual stock characteristics: return, volatility, and price scale from stock price. Second, RVN-UP presents a modeling approach that integrates stock characteristics and universal price patterns based on geometric Brownian motion (GBM). We propose a neural network to replace the Brownian motion part of GBM, addressing the challenge of predicting future stock prices due to the GBM's assumption that future prices are independent of past prices.

Our contributions are summarized as follows:

- **Learning Universal Stock Price Patterns.** RVN-UP learns universal stock price patterns by removing characteristics of individual stocks, such as return and volatility.
- **Integration of Short-Term Patterns and Long-Term Trends.** For accurate prediction, both long-term trends and short-term patterns must be considered together. RVN-UP identifies short-term patterns through a neural network and integrates them with long-term trends represented by return and volatility.
- **Experiments.** We demonstrate through experiments that RVN-UP improves information coefficient by up to 0.0222, and enhances annualized rate of return by up to 41.66%p compared to the state-of-the-art method. Additionally, we show



(a) U.S. stock market (b) CN stock market (c) KR stock market

Figure 1: Performance on U.S., Chinese, and Korean markets for stocks the models have not seen during training. RVN-UP identifies universal patterns even in previously unseen stocks, generating significant profits.

Table 1: Symbols and descriptions.

Symbol	Definition
S_t	stock price at time t
B_t	value of Brownian motion at time t
μ	return of a stock
σ	volatility of a stock
$S_t^c, S_t^o, S_t^h, S_t^l$	closing, opening, the highest, and the lowest prices at day t
$\epsilon_t^c, \epsilon_t^o, \epsilon_t^h, \epsilon_t^l$	error terms of closing, opening, the highest, and the lowest prices at day t
r	time interval between the opening and closing prices
u	time interval between the highest and closing prices
v	time interval between the lowest and closing prices
θ	parameters of the neural network inside RVN-UP
f_θ	neural network inside RVN-UP
$\mathbf{E}$	return-volatility normalized feature matrix
w	window size

that RVN-UP learns universal stock price patterns, improving information coefficient by up to 0.0484 and annualized rate of return by up to 58.63%p for stocks not seen during training (see Figure 1).

The rest of this paper is organized as follows. We introduce related research on stock price prediction in Section 2. We propose RVN-UP in Section 3. We present the experimental results on real-world stock datasets in Section 4 and conclude at Section 5. We summarize the definitions of the symbols in this paper in Table 1. The datasets used in our experiments are available at https://github.com/AnonymousCoder2357/RVN-UP.

2 Related Works

We introduce the related studies on stock price prediction. First, we present traditional stock price modeling methods, and then we discuss deep learning-based stock price prediction methods.

2.1 Traditional Stock Price Modeling Methods

In this subsection, we review several traditional methods used for stock price modeling. These methods primarily rely on statistical and mathematical foundations to capture the dynamics of stock prices over time.

Geometric Brownian motion (GBM) [2] is one of the most popular models for stock price dynamics. GBM is widely used in various financial tasks, such as option pricing and portfolio optimization, due to its suitability and simplicity. The GBM model assumes that the logarithm of stock prices follows a Brownian motion with drift, allowing for continuous price movements. In GBM, the stock price S_t at time t is modeled as:

$$S_t = S_0 e^{(\mu - \frac{1}{2}\sigma^2)t + \sigma B_t} \tag{2.1}$$

where μ, σ, and B_t are the return, volatility of the stock, and the value of Brownian motion at time t, respectively. Brownian motion (BM) B is a stochastic process with the following properties:

1. $B_0 = 0$
2. if $0 \le s_1 \le t_1 \le s_2 \le t_2$, then $B_{t_1} - B_{s_1}$ and $B_{t_2} - B_{s_2}$ are independent
3. $B_t - B_s \sim \mathcal{N}(0, t-s) \quad (\text{for } 0 \le s \le t)$
4. $t \to B_t$ is almost surely continuous

where B_t is the value of BM at time t.

S_t is expressed in terms of S_{t-1} when time is discretized with the following Lemma.

LEMMA 2.1. *When time is discretized, the stock price S_t at time $t \in \mathbb{N}$ is expressed as follows:*

$$S_t = S_{t-1} e^{\left(\mu - \frac{1}{2}\sigma^2\right) + \sigma \epsilon_t} \tag{2.2}$$

where the error term $\epsilon_t \sim \mathcal{N}(0, 1)$. ϵ_t and ϵ_s are independent for $s < t$ and $s \in \mathbb{N}$.

Proof. By Equation (2.1), the stock price S_{t-1} at time $t - 1$ is as follows:

$$S_{t-1} = S_0 e^{\left(\mu - \frac{1}{2}\sigma^2\right)(t-1) + \sigma B_{t-1}}. \tag{2.3}$$

By dividing Equation (2.1) by Equation (2.3), we obtain the following:

$$\frac{S_t}{S_{t-1}} = e^{\left(\mu - \frac{1}{2}\sigma^2\right) + \sigma (B_t - B_{t-1})}. \tag{2.4}$$

Let $\epsilon_t = B_t - B_{t-1}$. Then, ϵ_t and ϵ_s are independent for $s < t$ and $s \in \mathbb{N}$, and $\epsilon_t \sim \mathcal{N}(0, 1)$ due to properties 2 and 3 of BM, respectively. By replacing $B_t - B_{t-1}$ with ϵ_t in Equation (2.4), we obtain the result. □

The assumption that ϵ_t and ϵ_s are independent for $s < t$ and $t, s \in \mathbb{N}$ means that future stock prices cannot be predicted from past prices. Such assumption of independence makes GBM inappropriate for predicting future stock prices. To address this issue, attempts have been made to predict stock prices using modeling approaches such as Hamilton regime-switching model [5], fractional Brownian motion (fBM) [11], and ARIMA [1]. However, these methods struggle to capture complex stock price patterns due to their simplicity.

2.2 Deep Learning-Based Stock Price Prediction Methods With advancement of deep neural networks, attempts to predict stock prices using the time series processing capabilities of recurrent neural networks have emerged [12, 14]. The attention mechanism has been used to enhance the performance of RNN-based models [13, 7, 3]. GCN or Transformer architectures have been used to leverage the correlations between stocks [9, 16, 8]. There have been also methods [10, 17] to detect domain shifts that may occur during the evolution of stock time series, and adapt to various market conditions. However, these methods struggle to learn universal stock price patterns since they fail to remove the unique characteristics of individual stocks from price data. In this work, we aim to learn universal stock price patterns by removing stock characteristics to improve the accuracy of stock price prediction.

3 Proposed Method

We propose RVN-UP (<u>R</u>eturn-<u>V</u>olatility <u>N</u>ormalization for Learning <u>U</u>niversal Stock Price <u>P</u>atterns), a novel method for accurate stock price prediction by learning universal stock price patterns.

3.1 Overview We aim to accurately predict future stock prices by learning universal stock price patterns. The formal problem definition is as follows:

Given the historical prices S_t^o,S_t^h,S_t^l, and S_t^c for w days prior to day T where S_t^o,S_t^h,S_t^l, and S_t^c are opening, the highest, the lowest,and closing prices of day t, respectively,

Predict the closing price S_{T+1}^c for the next day.

To achieve accurate prediction, RVN-UP is designed to address the following challenges.

1. **Removing individual stock characteristics.** If stock price data contain individual stock characteristics, it becomes difficult to find universal stock price patterns shared by all stocks. However, existing methods learn from data that include individual stock characteristics. How can we remove individual stock characteristics from stock price data?
2. **Integrated stock price modeling.** The traditional stock price modeling method, geometric Brownian motion (GBM), cannot accurately predict stock prices due to the independence assumption that future prices are independent of past prices. How can we model stock prices by incorporating stock characteristics and universal price patterns?

We address the challenges with the following ideas.

1. **Return-volatility normalization (Section 3.2).** We normalize stock price data using the stock characteristics: return, volatility, and price scale in order to remove them from the data. As a result, we obtain the error terms, which constitute universal price features.
2. **Modeling price pattern (Section 3.3).** We propose a stock price modeling method based on GBM, in order to accurately predict the future stock price given universal price features. We replace the Brownian motion (BM) part of GBM with a neural network that captures universal stock price patterns, thereby eliminating the independence assumption. Consequently, we model stock prices by integrating stock characteristics and the universal price patterns.

3.2 Return-Volatility Normalization We propose a method to transform stock price data into universal price features. Specifically, we convert opening, the highest, the lowest, and closing prices into the error terms in Equation (2.2).

To learn universal stock price patterns shared by all stocks, individual stock characteristics must be removed from stock price data. To achieve this goal, we must first define the stock characteristics. In the widely used stock modeling method GBM [2], stock characteristics are

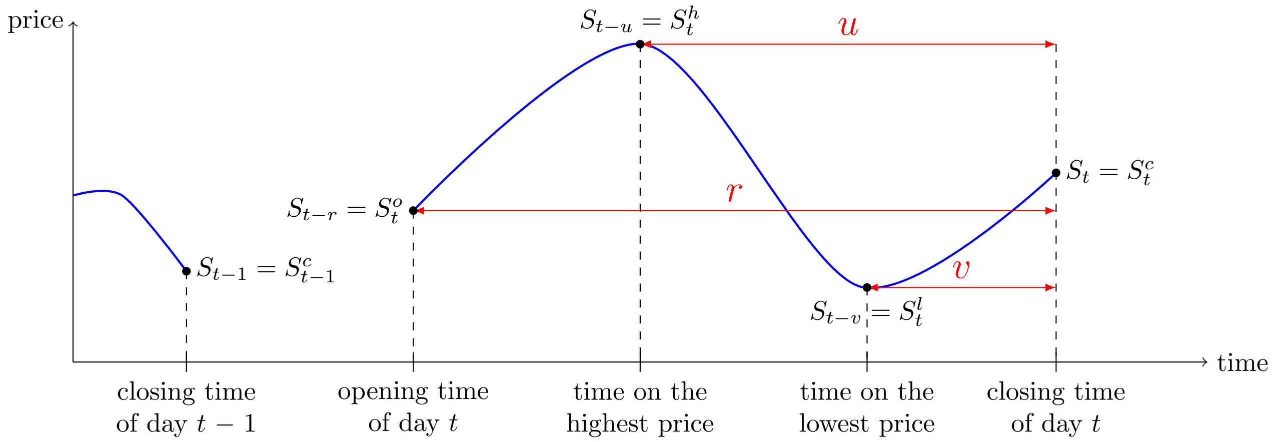


Figure 2: Opening, the highest, the lowest, and closing prices on a continuous time axis.

defined as return, volatility, and price scale. Observing Equation (2.2), the equation consists of the variables μ (return), σ (volatility), S_{t-1} (price scale), and ϵ_t (error term), among which μ, σ, and S_{t-1} represent stock characteristics. Note that 1) the error term represents values that do not reflect stock characteristics, and 2) the values of the error term over time form a universal price features that is not influenced by stock characteristics. Therefore, we normalize the stock price by return, volatility, and price scale to obtain the value of the error term and use it as a feature for learning.

Return and Volatility Estimating. We first estimate the return and volatility of the stock. We observe that $\log \frac{S_t^c}{S_{t-1}^c} = \mu - \frac{1}{2}\sigma^2 + \sigma\epsilon_t^c \sim \mathcal{N}(\mu - \frac{1}{2}\sigma^2, \sigma^2)$ by applying Equation (2.2) to closing prices. Thus, $\mu - \frac{1}{2}\sigma^2$ and σ are estimated as follows:

$$\widehat{\mu - \frac{1}{2}\sigma^2} = \mathbb{E}\left[\log \frac{S_t^c}{S_{t-1}^c}\right] \tag{3.5}$$

$$\hat{\sigma} = \sqrt{\mathrm{Var}\left(\log \frac{S_t^c}{S_{t-1}^c}\right)}. \tag{3.6}$$

Obtaining Error Term. We obtain the error term ϵ_t^c by normalizing the logarithmic daily return using the estimated return and volatility:

$$\epsilon_t^c = \frac{1}{\hat{\sigma}}\left(\log \frac{S_t^c}{S_{t-1}^c} - \left(\widehat{\mu - \frac{1}{2}\sigma^2}\right)\right). \tag{3.7}$$

The error term ϵ_t^c represents the error of the closing price on day t relative to the closing price on day $t-1$. Thus, we exploit ϵ_t^c as a feature for training in place of S_t^c, the closing price on day t.

A simple way to normalize the opening price is to apply Equation (3.7) to the opening price in the same manner. However, in this case, the error term represents the change from the opening price on day $t-1$ to the opening price on day t. Then, the relationship between the opening and closing prices cannot be captured. Therefore, we propose to place the opening and closing prices on a continuous time axis $[0, \infty)$ and calculate error terms that capture the relationship between them. The same approach is applied to the highest and lowest prices to capture their relationship with the closing price.

Let S_t represents the stock price at time t on a continuous time axis $[0, \infty)$. On this continuous time axis, $t \in \mathbb{N}$ represents the closing time of day t where we regard t as an integer. That is, $S_t = S_t^c$ for $t \in \mathbb{N}$. We place the opening, highest, and lowest prices at day t on this time axis. These prices are located between time $t-1$ and t. Let $0 \leq r, u, v \leq 1$ represent the time intervals between the opening, high, low prices, and the closing price, respectively. Then, on the continuous time axis, $S_{t-r} = S_t^o$, $S_{t-u} = S_t^h$, and $S_{t-v} = S_t^l$. Figure 2 visualizes the above logic.

By Equation (2.1), the opening price S_{t-r} is expressed in terms of the closing price S_t:

$$S_{t-r} = S_t e^{-(\mu - \frac{1}{2}\sigma^2)r - \sigma\epsilon_t^o} \tag{3.8}$$

where $\epsilon_t^o \sim \mathcal{N}(0, r)$ is the error term representing $B_t - B_{t-r}$. ϵ_t^o is determined if r is known since other variables ($S_{t-r}, S_t, \mu,$ and σ) are already known. However, determining the time interval between the opening and closing prices is not straightforward. We might consider setting r to 1 since the opening price is the immediate price following the previous day's closing price. On the other hand, due to price fluctuations during after-hours trading, the opening price may differ from the previous day's closing price, meaning that r cannot be set to 1. Therefore, we need to estimate r before normalizing the opening price. We estimate r based on the assumption that stock prices follow a geometric modeling with respect to time. Thus, we estimate r with the following Lemma.

LEMMA 3.1. *By minimizing a mean squared error, r is estimated as follows:*

$$\hat{r} = \frac{\mathbb{E}\left[\log \frac{S_{t-1}}{S_t} \log \frac{S_{t-r}}{S_t}\right]}{\mathbb{E}\left[\left(\log \frac{S_{t-1}}{S_t}\right)^2\right]}. \tag{3.9}$$

Proof. According to the geometric modeling of stock prices, we express the opening price on day t as follows:

$$S_{t-r} = S_t \left(\frac{S_{t-1}}{S_t}\right)^r. \tag{3.10}$$

By taking the log of Equation (3.10), we rewrite it as follows:

$$\log \frac{S_{t-r}}{S_t} = r \log \frac{S_{t-1}}{S_t}. \tag{3.11}$$

We need to minimize the following mean squared error:

$$\mathbb{E}\left[\left(r \log \frac{S_{t-1}}{S_t} - \log \frac{S_{t-r}}{S_t}\right)^2\right]. \tag{3.12}$$

Equation (3.12) attains its minimum at the point where the derivative with respect to r is equal to zero since it is convex with respect to r:

$$\begin{aligned} 0 &= \frac{d}{ds}\mathbb{E}\left[\left(r \log \frac{S_{t-1}}{S_t} - \log \frac{S_{t-r}}{S_t}\right)^2\right] && (3.13)\\ &= \mathbb{E}\left[2 \log \frac{S_{t-1}}{S_t}\left(r \log \frac{S_{t-1}}{S_t} - \log \frac{S_{t-r}}{S_t}\right)\right] && (3.14)\end{aligned}$$

Solving Equation (3.14), we obtain the result. ◻

As a result, we obtain ϵ_t^o by solving Equation (3.8):

$$\epsilon_t^o = \frac{1}{\hat{\sigma}}\left(\log \frac{S_t}{S_{t-r}} - \left(\widehat{\mu - \frac{1}{2}\sigma^2}\right)\hat{r}\right). \tag{3.15}$$

The highest price also cannot be normalized to the error term using the previous day's highest price, since it fails to capture the relationship with the closing price. Furthermore, it is also impossible to normalize it in the same way as the opening price as shown in Equation (3.15), since the time interval u between the closing and highest prices is unknown.

Thus, we use an approximation to extract the normalized highest price. By Equation (2.1), the highest price S_{t-u} is expressed in terms of the closing price S_t:

$$S_{t-u} = S_t e^{-\left(\mu - \frac{1}{2}\sigma^2\right)u - \sigma\epsilon_t^h} \tag{3.16}$$

where $\epsilon_t^h \sim \mathcal{N}(0, u)$ is the error term representing $B_t - B_{t-u}$. To calculate ϵ_t^h from Equation (3.16), we need the value of u, which is unknown. Thus, we approximate the drift $-\left(\mu - \frac{1}{2}\sigma^2\right)u$ as negligible. The justification for ignoring $-\left(\mu - \frac{1}{2}\sigma^2\right)u$ is based on the statistical observation that $P\left(\left|\left(\mu - \frac{1}{2}\sigma^2\right)u\right| < \rho\left|\sigma\epsilon_t^h\right|\right) \geq P\left(|Z| > \frac{0.1372}{\rho}\right)$ for $\rho > 0$ where $Z \sim \mathcal{N}(0,1)$. To present specific numbers, the probability that $\left|\left(\mu - \frac{1}{2}\sigma^2\right)u\right|$ is less than half, a third, and a quarter of $\left|\sigma\epsilon_t^h\right|$ is at least 78.38%, 68.06%, and 58.31%, respectively. The above claim follows from the following simple derivation:

Table 2: Mean of the ratio $\frac{\left|\mu - \frac{1}{2}\sigma^2\right|}{\sigma}$ for each dataset.

Dataset	$\mathbb{E}\left[\frac{\left\|\mu - \frac{1}{2}\sigma^2\right\|}{\sigma}\right]$
U.S.	0.1372
CN	0.1394
KR	0.1369

$$\begin{aligned} &P\left(\left|\left(\mu - \frac{1}{2}\sigma^2\right)u\right| < \rho\left|\sigma\epsilon_t^h\right|\right) && (3.17)\\ &= P\left(\left|\frac{1}{\sqrt{u}}\epsilon_t^h\right| > \frac{\sqrt{u}}{\rho}\frac{\left|\mu - \frac{1}{2}\sigma^2\right|}{\sigma}\right) && (3.18)\\ &\approx P\left(\left|\frac{1}{\sqrt{u}}\epsilon_t^h\right| > 0.1372\frac{\sqrt{u}}{\rho}\right) && (3.19)\\ &= P\left(|Z| > 0.1372\frac{\sqrt{u}}{\rho}\right) (\because \epsilon_t^h \sim \mathcal{N}(0,u)) && (3.20)\\ &\geq P\left(|Z| > \frac{0.1372}{\rho}\right) (\because 0 \leq \sqrt{u} \leq 1). && (3.21)\end{aligned}$$

$\frac{\left|\mu - \frac{1}{2}\sigma^2\right|}{\sigma}$ in Equation (3.18) is approximated to 0.1372 in Equation (3.19) by the mean of the ratio $\frac{\left|\mu - \frac{1}{2}\sigma^2\right|}{\sigma}$ of U.S. dataset in Table 2. The values for other datasets are similar, allowing for the same discussion. Therefore, we ignore the drift $-\left(\mu - \frac{1}{2}\sigma^2\right)u$ in Equation (3.16) and rewrite it as follows:

$$S_{t-u} \approx S_t e^{-\sigma\epsilon_t^h}. \tag{3.22}$$

As a result, we normalize the highest price as follows:

$$\epsilon_t^h = \frac{1}{\hat{\sigma}}\left(\log \frac{S_t}{S_{t-u}}\right). \tag{3.23}$$

We also normalize the lowest price in the same way:

$$\epsilon_t^l = \frac{1}{\hat{\sigma}}\left(\log \frac{S_t}{S_{t-v}}\right). \tag{3.24}$$

3.3 Modeling Universal Price Pattern

We transformed raw price features into universal price features, devoid of individual stock characteristics, through return-volatility normalization. Now we need to predict the future price from these universal price features. A naive approach to predict future prices is to train a neural network that maps the universal price features to the future price. However, this approach is inadequate since the future price includes individual stock characteristics; it is impossible to predict individual characteristics from features that do not include them. For example, consider a scenario where individual characteristics such as speaker's voice tone are removed from a sound sample, leaving only fundamental phoneme features for speech recognition. It is impossible to predict the voice tone from the fundamental phoneme features. Thus, we first predict the future error term that does not include individual stock characteristics, instead of the future stock price. After that, we combine the predicted feature with individual stock characteristics to ultimately predict the future stock price.

To achieve this, we replace the BM part of the GBM with a neural network. The neural network predicts the future error term from the past universal price features. Due to the independence assumption that future prices are independent of past prices (refer to property 2 of GBM in Section 2.1), GBM cannot predict future prices. However, by replacing the BM part with a neural network, the model gains predictive power. The neural network can be an existing time series prediction model or a stock price prediction model. We exploit DTML [16], a model that predicts stock prices based on the correlation between stocks, as the neural network. By replacing the BM part with a neural network, Equation (2.2) changes as follows:

$$S_t^c = S_{t-1}^c e^{\left(\mu-\frac{1}{2}\sigma^2\right)+\sigma f_\theta(\mathbf{E})} \tag{3.25}$$

where θ is the neural network parameters, f_θ is the neural network,

$$\mathbf{E} = \begin{bmatrix} \epsilon_{t-w}^c & \epsilon_{t-w+1}^c & \cdots & \epsilon_{t-1}^c \\ \epsilon_{t-w}^o & \epsilon_{t-w+1}^o & \cdots & \epsilon_{t-1}^o \\ \epsilon_{t-w}^h & \epsilon_{t-w+1}^h & \cdots & \epsilon_{t-1}^h \\ \epsilon_{t-w}^l & \epsilon_{t-w+1}^l & \cdots & \epsilon_{t-1}^l \end{bmatrix} \tag{3.26}$$

is the return-volatility normalized feature matrix, and w is the window size. By Equation (3.25), RVN-UP considers short-term pattern through the return-volatility normalized feature matrix $\mathbf{E}$, and long-term trend through the stock return μ and the volatility σ.

Model Training. Since we want $f_\theta(\mathbf{E})$ to predict ϵ_t, the neural network is trained to minimize the following mean squared error loss:

$$L(\theta) = \mathbb{E}\left[(f_\theta(\mathbf{E}) - \epsilon_t^c)^2\right]. \tag{3.27}$$

Algorithm 1 illustrates the whole training process.

4 Experiments

We present experimental results to answer the following research questions about our RVN-UP:

Q1. **Performance (Section 4.2).** Does RVN-UP outperform previous methods in stock price prediction?

Algorithm 1 Training Algorithm for RVN-UP

```
Input: stock price data
Output: optimal neural network parameter $\theta$
1: for each stock do
2:     $\widehat{\mu - \frac{1}{2}\sigma^2} \leftarrow \mathbb{E}\left[\log \frac{S_t}{S_{t-1}}\right]$
3:     $\hat{\sigma} \leftarrow \mathrm{Var}\left(\log \frac{S_t}{S_{t-1}}\right)$
4:     for $t = 1, 2, \ldots, T$ do
5:         $\epsilon_t \leftarrow \frac{1}{\hat{\sigma}}\left(\log \frac{S_t}{S_{t-1}} - \left(\widehat{\mu - \frac{1}{2}\sigma^2}\right)\right)$
6:         $\epsilon_t^o \leftarrow \frac{1}{\hat{\sigma}}\left(\log \frac{S_t}{S_{t-r}} - \left(\widehat{\mu - \frac{1}{2}\sigma^2}\right)\hat{r}\right)$
7:         $\epsilon_t^h \leftarrow \frac{1}{\hat{\sigma}}\left(\log \frac{S_t}{S_{t-u}}\right)$
8:         $\epsilon_t^l \leftarrow \frac{1}{\hat{\sigma}}\left(\log \frac{S_t}{S_{t-v}}\right)$
9:     end for
10: end for
11: Minimize $L(\theta)$ in Equation (3.27) using Adam [6]
```

Table 3: Summary of datasets.

Data	Stocks	Days	Dates
U.S.[1]	224	7659	1992-08-04 to 2022-12-30
CN[1]	171	2430	2011-01-06 to 2020-12-31
KR[1]	471	5391	2014-01-02 to 2022-11-01

[1] `https://github.com/AnonymousCoder2357/RVN-UP`

Q2. **Universal Price Pattern Learning (Section 4.3).** Does RVN-UP identify universal patterns to predict the future prices of stocks it has never encountered before?

Q3. **Hyperparameter Sensitivity (Section 4.4).** How sensitive is RVN-UP to hyperparameters?

Q4. **Ablation Study (Section 4.5).** Does each module of RVN-UP contribute to improving the performance for stock price prediction?

4.1 Experimental Setup

We present a dataset, competitors, and evaluation metrics for experiments on stock price prediction.

Datasets. We use three stock market datasets: U.S., CN, and KR markets. U.S., CN, and KR market datasets are collected from the stock markets of the United States, China, and Korea, respectively. The summary of the datasets is presented in Table 3. The goal is to predict the closing price on day $t+1$ given the price features up to day t.

Data Normalization. We generate an input

Table 4: An input feature vector ϵ_t at day t.

Feature	Calculation
ϵ_t^o	$\epsilon_t^o = S_t^o/S_t^c - 1$
ϵ_t^h	$\epsilon_t^h = S_t^h/S_t^c - 1$
ϵ_t^l	$\epsilon_t^l = S_t^l/S_t^c - 1$
ϵ_t^c	$\epsilon_t^c = S_t^c/S_{t-1}^c - 1$

Table 5: Stock prediction performance of RVN-UP and competitors measured by Information Coefficient (IC) and Annualized Rate of Return (ARR). RVN-UP shows the best performance across all datasets in all evaluation metrics, demonstrating up to 0.0222 higher IC and 41.66%p higher ARR compared to its competitors.

Method	U.S.		CN		KR	
	IC	ARR	IC	ARR	IC	ARR
MLP	0.0274	16.82%	0.0274	14.92%	0.0366	-1.23%
LSTM [12]	0.0460	-2.00%	0.0312	26.40%	0.0507	13.80%
ALSTM [3]	0.0441	-0.61%	0.0282	15.66%	0.0546	62.08%
DTML [16]	0.0349	18.20%	0.0148	2.55%	0.0133	-57.45%
RVN-UP (proposed)	**0.0673**	**26.10%**	**0.0498**	**68.06%**	**0.0768**	**95.51%**

Table 6: Stock prediction performance on unseen stocks of RVN-UP and competitors measured by Information Coefficient (IC) and Annualized Rate of Return (ARR). RVN-UP demonstrates the best performance on unseen stocks by learning universal patterns while other methods struggle to identify patterns in unseen stocks.

Method	U.S.		CN		KR	
	IC	ARR	IC	ARR	IC	ARR
MLP	0.0052	3.10%	0.0203	13.29%	0.0329	-17.86%
LSTM [12]	0.0542	15.96%	0.0207	9.19%	0.0414	-15.63%
ALSTM [3]	0.0470	5.08%	0.0313	11.96%	0.0437	3.73%
DTML [16]	0.0349	11.53%	0.0101	4.30%	0.0104	-38.88%
RVN-UP (proposed)	**0.0637**	**24.70%**	**0.0353**	**31.32%**	**0.0921**	**62.36%**

vector $\epsilon_{\mathbf{t}}$ at day t for competitors using widely used price ratio-based normalization method [3, 16]. In contrast, RVN-UP exploits the normalization technique described in Section 3.2. The normalization method for competitors is shown in Table 4. ϵ_t^o, ϵ_t^h, and ϵ_t^l indicate the relative values of the opening, highest, and lowest prices in comparison to closing price at day t, respectively. ϵ_t^c indicates the relative value of the closing price at day t in comparison to closing price at day $t-1$.

Competitors. We compare RVN-UP with the following competitors.

- **MLP** is the simplest baseline with two hidden layers.
- **LSTM** [12] is a recurrent neural network designed to capture long-term dependencies in time series data.
- **ALSTM** [3] enhances the LSTM model by incorporating an attention mechanism, allowing it to focus on the most relevant time steps for improved stock price prediction.
- **DTML** [16] is the previous state-of-the-art model for stock price prediction, leveraging temporal and inter-stock correlations with global market context.

We exploit DTML as the neural network in RVN-UP.

Evaluation Metrics. We conduct 10 experiments with different random seeds ranging from 0 to 9, and report the average across all cases. We evaluate the results of stock price prediction using Information Coefficient (IC). IC is the dominant metric in finance, which measures the linear relationship between the predicted prices and the actual prices. IC is calculated as:

$$IC = \frac{\text{cov}(\hat{y}, y)}{\text{std}(\hat{y})\text{std}(y)} \tag{4.28}$$

where $\hat{y}$ and y are the predicted price and the actual price, respectively.

We evaluate the portfolio's performance using the Annualized Rate of Return (ARR). The portfolio is rebalanced at the close of each day to allocate a weight of 0.2 to the top 5 stocks with the highest predicted returns.

Hyperparameters. We optimize the network parameters using the Adam [6] optimizer and apply early stopping based on the IC of the validation set. We conduct a hyperparameter search on learning rate, batch size, weight decay, and window size in $\{0.0001, 0.0005, 0.001, 0.005, 0.01\}$, $\{32, 64, 138\}$, $\{0.001, 0.005, 0.01, 0.05, 0.1\}$, and $\{8, 16, 32\}$, respectively.

4.2 Performance Table 5 compares the performance of RVN-UP and the baselines for stock price prediction across three datasets. RVN-UP achieves the state-of-the-art performance on all datasets through consistent improvement across all metrics compared to all other methods. RVN-UP learns universal price patterns shared by all stocks more effectively, by removing individual stock characteristics from price features.

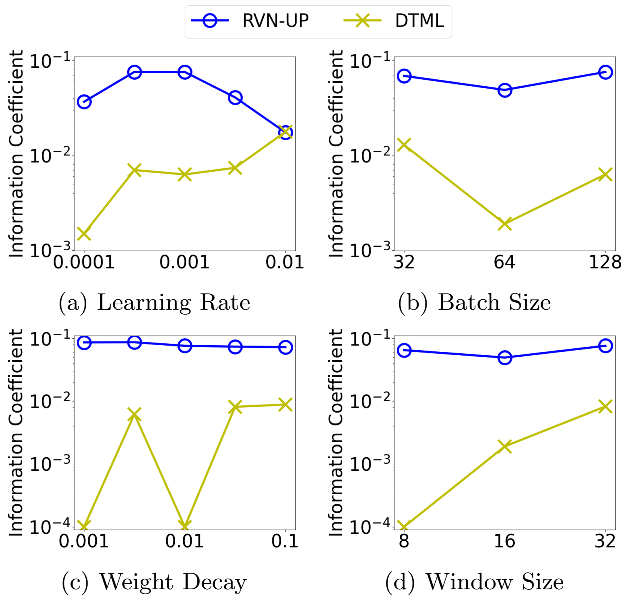


Figure 3: Comparison of hyperparameter sensitivity between RVN-UP and DTML.

4.3 Universal Price Pattern Learning Table 6 compares the performance of RVN-UP and the baselines on unseen stocks. We evaluate RVN-UP's ability to learn universal price patterns by testing its performance on stocks that were not included in the training set. We randomly split the stocks in the dataset into two halves, with one half used for training and the other half for testing. This approach is crucial for assessing how well RVN-UP generalizes to new, unseen data.

RVN-UP successfully identifies the price patterns of unseen stocks while the baselines struggle to recognize the price patterns of unfamiliar stocks. This success indicates that our approach to learning universal patterns is effective, as RVN-UP consistently identifies relevant price movements, even in unfamiliar stocks. The findings suggest that RVN-UP adapts more effectively to the diverse dynamics of stock markets by emphasizing learning of universal patterns.

4.4 Hyperparameter Sensitivity Figure 3 shows the comparison of hyperparameter sensitivity between RVN-UP and DTML. DTML is sensitive to hyperparameters due to its complexity. In contrast, the return-volatility normalization of RVN-UP reduces sensitivity to hyperparameters and minimizes the need for extensive hyperparameter tuning, achieving superior performance.

4.5 Ablation Study We analyze the performance of RVN-UP and its variants by removing each of the two primary modules, as shown in Table 7:

- RVN-UP-R: RVN-UP without the return-volatility normalization
- RVN-UP-N: RVN-UP without the neural network

Table 7: An ablation study of RVN-UP on U.S. dataset. RVN-UP-R and RVN-UP-N represent RVN-UP without the return-volatility normalization and neural network, respectively.

Method	IC	ARR
RVN-UP-R	0.0349	11.53%
RVN-UP-N	-0.0285	20.80%
RVN-UP	**0.0637**	**24.70%**

Each module contributes to improved prediction performance, with RVN-UP incorporating the two modules achieving the highest IC and ARR. Our observations suggest that the neural network module (NN) is more crucial than the return-volatility normalization (RVN), as NN serves the role of finding universal price patterns. RVN removes individual stock characteristics from the price data to generate universal price features.

5 Conclusion

We propose a novel framework RVN-UP for accurate stock price prediction. RVN-UP learns universal price patterns shared by all stocks to accurately predict stock prices. The main ideas of RVN-UP are: a) removing price features by normalizing them with individual stock characteristics: return, volatility, and price scale, and b) presenting a stock price modeling approach based on GBM to integrate universal pattern and individual stock characteristics. Additionally, RVN-UP integrates both short-term patterns and long-term trends in stock prices. RVN-UP achieves the state-of-the-art performance in stock price prediction on real-world datasets with improvement of up to 0.0222 in information coefficient and 41.66%p in annualized rate of return compared to the previous state-of-the-art models. Additionally, RVN-UP successfully recognizes price patterns of stocks unseen in the training set unlike existing methods.

References

[1] D. Asteriou and S. G. Hall, *Arima models and the box–jenkins methodology*, Applied Econometrics, 2 (2011), pp. 265–286.

[2] T. Bjork, *Arbitrage theory in continuous time*, (2019).

[3] F. Feng, H. Chen, X. He, J. Ding, M. Sun, and T.-S. Chua, *Enhancing stock movement prediction with adversarial training.*, in IJCAI, vol. 19, 2019, pp. 5843–5849.

[4] I. Gat, H. Aronowitz, W. Zhu, E. Morais, and R. Hoory, *Speaker normalization for self-supervised speech emotion recognition*, in ICASSP 2022-2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), IEEE, 2022, pp. 7342–7346.

[5] J. D. Hamilton, *Regime switching models*, Palgrave Macmillan UK, London, 2010, pp. 202–209.

[6] D. P. Kingma, *Adam: A method for stochastic optimization*, arXiv preprint arXiv:1412.6980, (2014).

[7] H. Li, Y. Shen, and Y. Zhu, *Stock price prediction using attention-based multi-input lstm*, in Asian conference on machine learning, PMLR, 2018, pp. 454–469.

[8] T. Li, Z. Liu, Y. Shen, X. Wang, H. Chen, and S. Huang, *Master: Market-guided stock transformer for stock price forecasting*, in Proceedings of the AAAI Conference on Artificial Intelligence, vol. 38, 2024, pp. 162–170.

[9] W. Li, R. Bao, K. Harimoto, D. Chen, J. Xu, and Q. Su, *Modeling the stock relation with graph network for overnight stock movement prediction*, in Proceedings of the twenty-ninth international conference on international joint conferences on artificial intelligence, 2021, pp. 4541–4547.

[10] H. Lin, D. Zhou, W. Liu, and J. 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, 2021, pp. 1017–1026.

[11] B. B. Mandelbrot and J. W. Van Ness, *Fractional brownian motions, fractional noises and applications*, SIAM review, 10 (1968), pp. 422–437.

[12] D. M. Nelson, A. C. Pereira, and R. A. De Oliveira, *Stock market's price movement prediction with lstm neural networks*, in 2017 International joint conference on neural networks (IJCNN), Ieee, 2017, pp. 1419–1426.

[13] Y. Qin, D. Song, H. Cheng, W. Cheng, G. Jiang, and G. W. Cottrell, *A dual-stage attention-based recurrent neural network for time series prediction*, in Proceedings of the 26th International Joint Conference on Artificial Intelligence, IJCAI'17, AAAI Press, 2017, p. 2627–2633.

[14] M. A. I. Sunny, M. M. S. Maswood, and A. G. Alharbi, *Deep learning-based stock price prediction using lstm and bi-directional lstm model*, in 2020 2nd novel intelligent and leading emerging sciences conference (NILES), IEEE, 2020, pp. 87–92.

[15] Y. Xu, J. Du, L.-R. Dai, and C.-H. Lee, *A regression approach to speech enhancement based on deep neural networks*, IEEE/ACM transactions on audio, speech, and language processing, 23 (2014), pp. 7–19.

[16] J. Yoo, Y. Soun, Y.-c. Park, and U. Kang, *Accurate multivariate stock movement prediction via data-axis transformer with multi-level contexts*, in Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining, 2021, pp. 2037–2045.

[17] D. Zhan, Y. Dai, Y. Dong, J. He, Z. Wang, and J. Anderson, *Meta-adaptive stock movement prediction with two-stage representation learning*, in Proceedings of the 2024 SIAM International Conference on Data Mining (SDM), SIAM, 2024, pp. 508–516.