Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence (IJCAI-20)
Special Track on AI in FinTech

Relation-Aware Transformer for Portfolio Policy Learning

Ke Xu[1,2*], **Yifan Zhang**[1,2*], **Deheng Ye**[3], **Peilin Zhao**[3†], **Mingkui Tan**[1†]
[1]South China University of Technology, Guangzhou, China
[2]Pazhou Lab, Guangzhou, China
[3]Tencent AI Lab, Shenzhen, China
{ms201721045770, sezyifan}@mail.scut.edu.cn, {dericye,masonzhao}@tencent.com,
mingkuitan@scut.edu.cn

Abstract

Portfolio selection is an important yet challenging task in AI for FinTech. One of the key issues is how to represent the non-stationary price series of assets in a portfolio, which is important for portfolio decisions. The existing methods, however, fall short of capturing: 1) the complicated sequential patterns for asset price series and 2) the price correlations among multiple assets. In this paper, under a deep reinforcement learning paradigm for portfolio selection, we propose a novel Relation-aware Transformer (RAT) to handle these aspects. Specifically, being equipped with our newly developed attention modules, RAT is structurally innovated to capture both sequential patterns and asset correlations for portfolio selection. Based on the extracted sequential features, RAT is able to make profitable portfolio decisions regarding each asset via a newly devised leverage operation. Extensive experiments on real-world crypto-currency and stock datasets verify the state-of-the-art performance of RAT.[1]

1 Introduction

Portfolio selection (PS) is an important research problem in computational finance [Li and Hoi, 2014; Li and Ng, 2000]. PS aims to maximize the long-term returns of wealth by dynamically allocating the wealth among a set of assets, *e.g.,* stocks and crypto-currencies. Despite making great financial sense, it is very difficult for investors to handle such a laborious task, since even domain experts have to spend a large amount of effort/time in investigating each asset and managing portfolios. Recently, machine learning based methods have been proposed to address this task and have shown empirical improvements at PS [Agarwal *et al.*, 2006; Das *et al.*, 2014]. However, due to the complex nature of PS, existing methods may not be able to achieve promising performance in practice. Specifically, it is very hard to represent the non-stationary price series of assets, as they often contain significant noises and oscillations [Zhang *et al.*, 2020].

The majority of existing PS methods [Cover and others, 1991; Shen and others, 2015] heavily rely on handcrafted features, such as moving average [Li and Hoi, 2012], stochastic technical indicators [Neely *et al.*, 2014], etc. These features, however, have shown limited representation ability over practical price series [Zhang *et al.*, 2020]. Recently, deep neural networks (DNNs) have become popular in sequential modeling and shown stronger sequential representation ability [LeCun *et al.*, 2015]. However, directly applying existing DNNs for portfolios cannot extract price sequential features well due to **two practical challenges** listed below.

First, price sequences of assets often follow complex financial laws that are hard to capture using existing DNNs. For instance, asset prices contain complex short-term trends [Atsalakis and others, 2009], which potentially shed light on local patterns of price series. Here, the local pattern indicates the sequential pattern of a local price subsequence. Moreover, asset prices generally satisfy the long-term mean reversion principle [Poterba and Summers, 1988]. That is, the price of an asset will finally reflect its true value. In this sense, both long-term and local sequential patterns of price series are important for PS. To handle this, one may employ recurrent neural networks (RNNs) (such as LSTM [Hochreiter and others, 1997] and GRU [Cho *et al.*, 2014]) to model price series. However, empirical studies [Li *et al.*, 2019b; Zhang *et al.*, 2020] have found that these methods are struggling to capture (very) long-term dependencies. Thus, how to exploit DNNs to effectively capture the sequential patterns for portfolio assets remains an open challenge.

Second, financial assets in portfolios contain complex correlations that may vary rapidly over time [Stefanova and Elkamhi, 2011]. Such asset correlations are important in mining financial data [Frye, 2008; Lopez, 2004], as they are helpful for the estimation of investment risk, thereby guiding more effective portfolio management. For instance, during economic downturns, investors may reduce investment risk by increasing portfolio diversification [Tasca *et al.*, 2017] based on asset correlations; while during economic upturns, they may increase portfolio returns by exploiting the synergy of upward/related assets. However, existing DNNs for series modeling generally extract features for each temporal series, without particular considerations in modeling the correlations among assets. Hence, how to capture asset correlations in price sequences is another under-explored problem in PS.

*Equal contributions. Work done as interns at Tencent AI Lab.
†Corresponding author.
[1]The source code is available: https://github.com/Ivsxk/RAT.

In recent years, the attention mechanism [Vaswani *et al.*, 2017] has gained great attention in compelling sequence modeling. More recently, Transformer [Vaswani *et al.*, 2017], which relies on the attention mechanism, has shown great potential in grasping repeating sequential patterns with very long-term dependencies. However, Transformer cannot be directly applied to PS due to two reasons. First, in Transformer, the similarities between queries and keys in self-attention layers are computed based on point-wise values, which makes it incapable of capturing local context information and easily confused by local noisy points [Li *et al.*, 2019b]. Moreover, as Transformer is primarily designed for word-level sequence translation, it is unable to capture the assets correlations that are useful in making portfolio decisions.

In this paper, we propose to exploit and improve Transformer for long-term sequence modeling in portfolios, so that it can well capture local sequential patterns of price series and the correlations among assets. Specifically, we propose a Relation-aware Transformer (RAT), which serves as a policy network for PS. RAT is structurally innovated to capture sequential patterns and asset correlations via newly proposed attention-based modules, and makes profitable portfolio decisions through a newly devised decision-making module. By exploiting reinforcement learning to train the policy network, RAT yields significant improvements in PS performance.

Our main contributions are summarized as follows:

- We propose a novel RAT method for PS. According to our best knowledge, this is the first attention-based method to simultaneously model complex sequential patterns and varying asset correlations for PS.
- By exploring financial leverage, we resolve a decision-making limitation in existing deep reinforcement learning methods for PS. Based on a new decision-making module, RAT makes more profitable portfolio decisions under the deep reinforcement learning framework.
- Extensive experiments on real-world datasets verify the significant superiority of our method, compared with state-of-the-art methods in PS, including both online learning and reinforcement learning based methods.

2 Problem Formulation

Without loss of generality, we consider a portfolio selection (PS) task with m assets during a total number of n trading periods. Assume each asset has d kinds of prices. Here, we consider 4 kinds of prices, namely *opening, highest, lowest* and *close* prices (which means $d = 4$). As the prices often change over time, we use $\mathbf{p}_{t,i}{\in}\mathbb{R}_+^d$ to denote the price of asset i at period t (where $t{=}1, .., n$) and let $\mathbf{P}_t{\in}\mathbb{R}^{m\times d}$ be the prices for all assets at this period. Moreover, let $\boldsymbol{\mathcal{P}}_t{=}\{\mathbf{P}_{t-k}, .., \mathbf{P}_{t-1}\}{\in}\mathbb{R}^{k\times m\times d}$ be the price series of previous k-moment prices regarding period t.

Portfolio Selection as a Markov Decision Process. The price of assets is determined by many factors (such as the market), so it is impossible to annotate the data in advance. As a result, the traditional supervised learning paradigm is not suitable for modeling the PS process. In fact, given the decision nature of PS, it is more natural and convenient to model it as a Markov Decision Process (MDP). An MDP can be defined as a tuple $(\mathcal{S}, \mathcal{A}, \mathcal{T}, \mathcal{R})$, where $\mathcal{S}$ denotes a finite state space, $\mathcal{A}$ denotes a finite set of actions, $\mathcal{T}(s'|s, a)$ is a state transition function that defines the next state s' given the current state s and action a, and $\mathcal{R}(s, a)$ is a reward function. Moreover, a policy $\pi(a|s)$ determines an action a given the current state s. In the context of PS, the MDP model will be slightly different from the standard ones. Specifically, the action is specified by a portfolio vector $\mathbf{a}_t{=}[a_{t,1}, a_{t,2}, ...a_{t,m}]^\top{\in}\mathbb{R}^m$, where $a_{t,i}$ indicates the wealth proportion regarding asset i and $\sum_{i=1}^m a_{t,i}{=}1$. To construct a PS policy, at period t, an agent observes a state of price series $\mathbf{s}_t{=}\boldsymbol{\mathcal{P}}_t{\in}\mathcal{S}$, and takes an action $\mathbf{a}_t{=}\pi(\boldsymbol{\mathcal{P}}_t){\in}\mathcal{A}$. Afterwards, the agent receives a reward $r_t{\in}\mathcal{R}(s, a)$, while the next state $\mathbf{s}_{t+1}$ is reached based on $\mathbf{s}_{t+1}{\sim}\mathcal{T}(\mathbf{s}_t, \mathbf{a}_t)$. In practice, if $\mathbf{s}_{t+1}$ is mainly determined by the market, then $\mathbf{s}_{t+1}{\sim}\mathcal{T}(\mathbf{s}_t)$.

In this paper, we aim to devise a policy network (served as π) to maximize the accumulated reward (*e.g.,* the overall wealth of portfolios) via reinforcement learning (RL). However, it is non-trivial to devise such a policy network, as non-stationary price series and complex asset correlations make sequential modeling very difficult. Existing RL-based methods for PS directly use existing DNN models for PS, which, however, cannot extract the complex information well. To solve this, we propose a Relation-Aware Transformer.

3 Relation-Aware Transformer

In portfolio selection (PS), both price series patterns and asset correlations are significant for correct decision-making of portfolios and it is important to capture both types of information in the learning process. To this end, we propose a Relation-Aware Transformer (RAT), which will serve as a policy network for PS. The goal of such a policy network is to extract informative features for asset prices, and make portfolio decisions based on these features. To this end, we extend the standard Transformer structure by making it be relation-aware and handle this task with reinforcement learning (RL).

3.1 General Architecture

RAT follows an encoder-decoder structure [Vaswani *et al.*, 2017]. As shown in Figure 1, RAT consists of an encoder and a decoder, where the encoder is for sequential feature extraction, while the decoder is for decision making.

Encoder: In PS, both sequential patterns and asset correlations are necessary for correct decision-making of portfolios. Therefore, we propose two task-specific attention modules for the encoder. To be specific, a ***sequential attention layer*** is devised to capture sequential patterns for asset prices (See Section 3.2), and a ***relation attention layer*** is devised for capturing asset correlations (See Section 3.3).

Decoder: The decoder has network modules similar to the encoder, apart from a new ***decision-making layer*** (See Section 3.4). Such a layer aims to select portfolios by comprehensively considering the extracted features, local price context, and the decision of last period. Here, a new leverage operation is designed to enhance the decision making. Other components of the encoder-decoder structure, such as the positional encoding, feed forward layer and layer normalization, are identical to Transformer [Vaswani *et al.*, 2017].

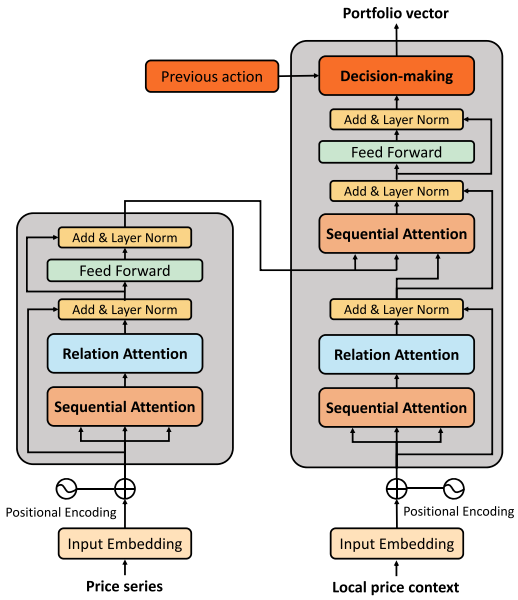

Figure 1: The scheme of Relation-aware Transformer. The left is the encoder, whose input is a price series $\mathcal{P}_t$ at period t. The right is the decoder, whose input is local price context $\mathcal{P}_t^l=\{\mathbf{P}_{t-l},..,\mathbf{P}_{t-1}\}$.

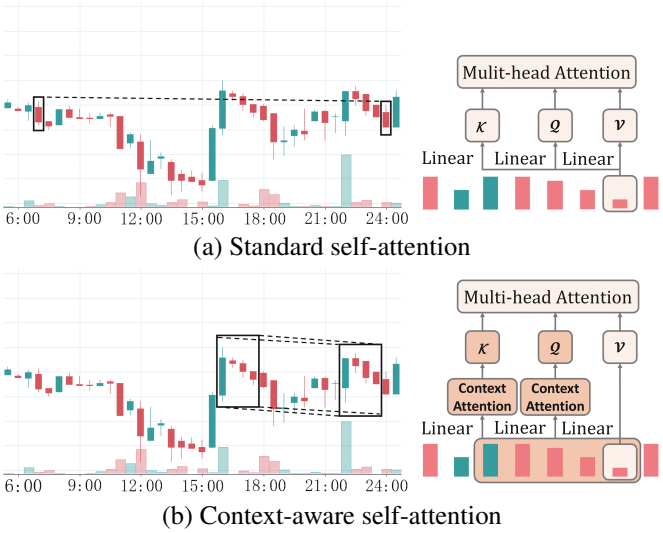

Figure 2: The comparison between different self-attention mechanisms. (a) standard self-attention in Transformer can be confused by noisy points due to point-wise matching. Instead, (b) context-aware self-attention uses context attention to transform local price context into queries/keys, thus being more robust to local price noise.

3.2 Sequential Attention Layer

Financial price series generally follows the mean reversion principle in the long term. Besides, a price series is often affected by surrounding events, such as interest rate cutting, leading to oscillations in the short term. Hence, it is non-trivial to devise DNNs for capturing such complex sequential patterns. Although self-attention works well in modeling long-term dependencies in Transformer, it falls short of exploiting local context since its query-key matching is computed based on point-wise values (as shown in Figure 2 (a)). As a result, standard self-attention may be confused by local noisy points, causing underlying optimization issues. To solve this, we devise a sequential attention layer, which enhances multi-head attention via new devised context attention. First, we introduce the multi-head attention scheme.

Multi-head Attention. With $\mathcal{P}_t\in\mathbb{R}^{k\times m\times d}$ as the input, a multi-head attention processes the price series $\mathcal{P}_{t,i}\in\mathbb{R}^{k\times d}$ for each asset $i\in\{1,2,..,m\}$. Specifically, it first transforms $\mathcal{P}_{t,i}$ into H distinct query matrices $\mathcal{Q}_{t,i}^h=\mathcal{P}_{t,i}\mathbf{W}_h^Q$, key matrices $\mathcal{K}_{t,i}^h=\mathcal{P}_{t,i}\mathbf{W}_h^K$ and value matrices $\mathcal{V}_{t,i}^h=\mathcal{P}_{t,i}\mathbf{W}_h^V$, respectively. Here, $h\in\{1,2,..,H\}$ denotes the head index, while $\mathbf{W}_h^Q,\mathbf{W}_h^K,\mathbf{W}_h^V\in\mathbb{R}^{d\times d_f}$ denote parameters of linear projections and d_f indicates the dimension of the projected feature space. After linear projections, the scaled dot-product attention is adopted to compute the output values as follows:

$$\mathcal{O}_{t,i}^h=\text{softmax}\left(\frac{\mathcal{Q}_{t,i}^h\mathcal{K}_{t,i}^{h\top}}{\sqrt{d_f}}\right)\mathcal{V}_{t,i}^h, \tag{1}$$

where $\sqrt{d_f}$ is a scale term [Vaswani *et al.*, 2017]. We then concatenate the outputs of all heads, *i.e.,* $\mathcal{O}_{t,i}\in\mathbb{R}^{k\times Hd_f}$. The final output is to concatenate all assets, *i.e.,* $\mathcal{O}_t\in\mathbb{R}^{k\times m\times Hd_f}$.

Note that the dot-product attention in Eq. (1) cannot well exploit the context information. To address this, as shown in Figure 2 (b), instead of using dot-point projections for query-key matching, we propose a context attention scheme to transform the local context into queries and keys.

Context Attention. To enhance the queries and keys with locality, we explore short-term dependencies between the current price and local price context. Specifically, at period t, we first introduce how to obtain the query matrix $\mathcal{Q}_{t,i}^h$ regarding head h in Eq. (1) with linear projection matrix $\hat{\mathbf{W}}_h^Q\in\mathbb{R}^{d\times d_f}$.

Without loss of generality, at any moment $\tau\in\{1,2,...,k\}$ in the price series $\mathcal{P}_t\in\mathbb{R}^{k\times m\times d}$, we input local price context $\mathcal{P}_\tau^l=\{\mathbf{P}_{\tau-l},..,\mathbf{P}_{\tau-1}\}\in\mathbb{R}^{l\times m\times d}$ (instead of only the current price) into context attention. Here, l is the length of context, and we use padding when $\tau\leq l$. For asset i, the context attention first transforms local context $\mathcal{P}_{\tau,i}^l\in\mathbb{R}^{l\times d}$ into the *context-aware key matrix* and *value matrix* $\hat{\mathcal{K}}_{\tau,i}^h=\hat{\mathcal{V}}_{\tau,i}^h=\mathcal{P}_{\tau,i}^l\hat{\mathbf{W}}_h^Q\in\mathbb{R}^{l\times d_f}$, while transforming the current price $\mathbf{P}_{\tau-1,i}$ into *context-agnostic query matrix* $\hat{\mathcal{Q}}_{\tau,i}^h=\mathbf{P}_{\tau-1,i}\hat{\mathbf{W}}_h^Q\in\mathbb{R}^{1\times d_f}$. After projection, the query matrix can be obtained by exploiting the dependencies between context-aware key and context-agnostic query through scaled attention as follows:

$$\mathcal{Q}_{\tau,i}^h=\text{softmax}\left(\frac{\hat{\mathcal{Q}}_{\tau,i}^h\hat{\mathcal{K}}_{\tau,i}^{h\top}}{\sqrt{d_f}}\right)\hat{\mathcal{V}}_{\tau,i}^h. \tag{2}$$

We then concatenate the outputs of all moments to obtain the queries $\mathcal{Q}_{t,i}^h\in\mathbb{R}^{k\times d_f}$. The keys $\mathcal{K}_{t,i}^h$ can be computed in the same manner with different parameters $\hat{\mathbf{W}}_h^K$. In this way, the queries and keys can be more aware of the local context and thus more robust to local price noise. When $l=1$, the sequential attention degrades to standard multi-head attention.

3.3 Relation Attention Layer

Assets in a portfolio inherently contain complex and varying correlations. Such correlations are important for PS since they reveal macro market trends and help to address investment risk by adjusting portfolio diversification. To capture the correlations, we devise a relation attention layer to enhance features after the sequential attention layer. With $\boldsymbol{\mathcal{O}}_t^h \in \mathbb{R}^{k\times m\times d_f}$ as the output of the h-th head in the previous sequential attention layer, the relation attention uses scaled self-attention to model asset correlations and enhance features $\boldsymbol{\mathcal{O}}_t^{h,j} \in \mathbb{R}^{m\times d_f}$ for each time point $j \in \{1, .., k\}$ as below:

$$\boldsymbol{\mathcal{Z}}_t^{h,j} = \text{softmax}\left(\frac{\boldsymbol{\mathcal{O}}_t^{h,j}\boldsymbol{\mathcal{O}}_t^{h,j^\top}}{\sqrt{d_f}}\right)\boldsymbol{\mathcal{O}}_t^{h,j}. \tag{3}$$

The output of the h-th attention head is the concatenation of all time points by $\boldsymbol{\mathcal{Z}}_t^h = \text{Concat}\{\boldsymbol{\mathcal{Z}}_t^{h,1}, .., \boldsymbol{\mathcal{Z}}_t^{h,k}\} \in \mathbb{R}^{k\times m\times d_f}$, while the output of the relation attention layer is to concatenate all heads by $\boldsymbol{\mathcal{Z}}_t = \text{Concat}\{\boldsymbol{\mathcal{Z}}_t^1, .., \boldsymbol{\mathcal{Z}}_t^H\} \in \mathbb{R}^{k\times m\times Hd_f}$.

3.4 Decision-making Layer

It is non-trivial to design a decision-making module for RAT due to the complexity of PS. Existing RL based methods for PS [Jiang *et al.*, 2017; Liang and others, 2018] decide the portfolio through a fully connected layer with softmax. However, using a softmax layer inevitably enforces the proportion of assets to be positive, *i.e.,* $a_{t,i} \in (0, 1)$ for asset i, where $\sum_{i=1}^m a_{t,i} = 1$. This may make the agent suffer a huge loss of wealth when the asset prices decrease in the future. The critical problem is that even if the agent can predict the decrease, it cannot avoid the loss due to $a_{t,i} \in (0, 1)$.

To enhance the decision-making ability of RAT, motivated by leverage [Mandelker and others, 1984], we introduce *short sale* [Shen and others, 2015; Shen and Wang, 2016] into the decision making. The *short sale* means that the policy network can first borrow some assets (whose prices are predicted to decrease by the network) for sale, and then reinvest the liquidated money into other assets (whose prices are predicted to increase). In this way, RAT is able to make more accurate decisions regarding each asset.

To this end, we devise a new leverage operation with three independent softmax fully-connected decision-making layer. Specifically, one head outputs an **initial portfolio vector** $\hat{\mathbf{a}}_t$, and one head outputs a **short sale vector** $\hat{\mathbf{a}}_t^s$, while the last one outputs a **reinvestment vector** $\hat{\mathbf{a}}_t^r$. Based on the three heads, we improve the decision by considering both short sale and reinvestment. That is, the final portfolio vector is decided by $\mathbf{a}_t = \hat{\mathbf{a}}_t - \hat{\mathbf{a}}_t^s + \hat{\mathbf{a}}_t^r$. In this way, the proportion regarding asset i becomes $a_{t,i} \in (-1, 2)$, where $\sum_{i=1}^m a_{t,i} = 1$. The negative sign of the weight indicates that investors will make money if asset price drops, while suffering a loss if the price rises.

In addition, since PS is also influenced by transaction costs, we adopt a recursive mechanism [Moody and Saffell, 2001] to avoid heavy transaction fees. That is, the decision-making requires taking into consideration the action from last period, which discourages very large changes between portfolios. To this end, we concatenate the portfolio vector from last period a_{t-1} into feature maps so that RAT can make profitable portfolio decisions while constraining aggressive trading.

3.5 Learning with Reinforcement Learning

Since the asset prices are determined by many factors such as the financial market, it is impossible for us to annotate the data in advance and train RAT with supervised learning. To address this, we resort to reinforcement learning (RL). Specifically, we adopt a classical direct policy gradient algorithm [Moody and Saffell, 2001] to train the policy network (RAT) by maximizing a reward function. Here, we first denote the price change of all assets by a price relative vector $\mathbf{y}_t := \frac{\mathbf{p}_t^c}{\mathbf{p}_{t-1}^c} \in \mathbb{R}^m$ regarding period t, where $\mathbf{p}_t^c$ indicates the close price. Based on the price relative vector, we devise the reward relying on the log-return of portfolios as follows:

$$R(\mathbf{s}_0, \mathbf{a}_0, .., \mathbf{s}_n, \mathbf{a}_n) = \frac{1}{n}\sum_{t=0}^{n} \ln(\mathbf{a}_t^\top \mathbf{y}_t(1 - c_t)), \tag{4}$$

where c_t denotes the transaction cost, computed using the method presented in [Jiang *et al.*, 2017]. The motivation lies in that the direct policy gradient algorithm guarantees at least a theoretical log-optimal strategy with the log-return as the reward. This can be proved by combining the previous theoretical results [Györfi and Vajda, 2008; Sutton *et al.*, 2000]. One can also use more advanced RL methods, *e.g.,* DDPG [Lillicrap and others, 2016] and PPO [Schulman and others, 2017]. However, we found that they usually fail to converge in PS in our preliminary studies. Since RL is not our main focus, we leave the study of more task-specific RL to future work.

4 Related work

Portfolio selection (PS) has attracted extensive research focus from the AI community [Shen and others, 2015; Shen and Wang, 2016]. Following the Kelly principle [Kelly, 1956], many types of PS methods have been proposed, including *online learning* and *reinforcement learning based* methods.

Online learning based methods aim to maximize the expected log-return in sequential decision-making [Zhao *et al.*, 2018; Zhang *et al.*, 2018]. Pioneering studies include UCRP [Cover and others, 1991], Anticor [Borodin *et al.*, 2004], and ONS [Agarwal *et al.*, 2006]. Recently, several methods have exploited the mean reversion property to select the portfolio, e.g., OLMAR [Li and Hoi, 2012] RMR [Huang *et al.*, 2013] and SSR [Shen and Wang, 2017]. However, these methods failed to take into account the learning of sequential features and only used handcrafted features, such as moving average and stochastic technical indicators. This can lead to unsatisfactory PS performance due to limited feature representations [Deng and others, 2016].

Reinforcement learning (RL) based methods aim to optimize specific utility functions and learn comprehensive policies via RL algorithms [Moody and Saffell, 2001; Neuneier, 1998]. Very recently, several RL-based studies have used deep learning [Niu *et al.*, 2020; Zhang *et al.*, 2019] to extract features for assets [Guo *et al.*, 2018; Jiang *et al.*, 2017; Kang *et al.*, 2018] and have achieved promising performance for PS. To be specific, the state-of-the-art ones are EIIE [Jiang *et al.*, 2017] and the adversarial deep RL method [Liang and others, 2018]. These methods resort to deep learning techniques [Cao *et al.*, 2019], *e.g.,* CNNs or RNNs, as the sequential feature extractor for PS.

By comparison, our proposed RAT explores attention mechanism to capture in-depth the task-specific information, *i.e.,* correlations among assets and long-term/local sequential patterns of assets. As a result, RAT learns more representative sequential features for PS.

Transformer is a powerful attention model primarily used in the field of NLP [Vaswani *et al.*, 2017]. Note that transformer employs a encoder-decoder structure to capture the sequential dependencies between the source and the target sequence. It uses multiple attention heads (with different parameters of linear projections) to capture diverse aspects of sequential patterns. The outputs of all heads are concatenated and then fed into a fully connected feed-forward layer. In this paper, we exploit it for modeling long-term series dependencies and devise a Relation-aware Transformer for PS.

5 Experimental Results

To verify our proposed method, we evaluate RAT in terms of three main aspects: (1) the profitability on real-world datasets, (2) the feature representation ability for portfolios, and (3) the benefits of the financial leverage.

5.1 Experimental Settings

We first describe the experimental settings.

Datasets. We examine RAT on real-world crypto-currency and stock datasets. The statistics of all datasets are summarized in Table 1. All crypto-currency datasets are accessed with Poloniex[2], where data selection is based on the method in [Jiang *et al.*, 2017]. To be specific, we select the assets according to the crypto-currencies with top month trading volumes in Poloniex. We also evaluate our methods on the S&P500 stock dataset obtained from Kaggle[3]. Since decision-making of PS relies on the relative price, we normalize the price series of each asset by element-wise dividing the prices regarding the last period in the price series.

Datasets	#Asset	Data Range	
		Training	Test
Crypto-A	12	2016-01 to 2017-11	2017-11 to 2018-01
Crypto-B	37	2017-11 to 2019-09	2019-09 to 2019-11
S&P500	506	2013-02 to 2017-08	2017-08 to 2018-02

Table 1: Statistics of datasets. The length of a price series spans 30 periods, where each period has a length of 30 minutes.

Baselines. We compare RAT with several advanced methods, including (1) online learning based methods: UCRP [Cover and others, 1991], Anticor [Borodin *et al.*, 2004], SSR [Shen and Wang, 2017] OLMAR [Li and Hoi, 2012] and RMR [Huang *et al.*, 2013]; (2) reinforcement learning (RL) based methods: ADDPG [Liang and others, 2018], EIIE [Jiang *et al.*, 2017] and MTL [Li *et al.*, 2019a]. All these RL methods use either LSTM or CNNs as the feature extractor. To evaluate the decision-making module, we also present a degenerate variant (**RAT-B**) that only uses a **basic** softmax in the decision-making layer without the leverage operation.

[2]Poloniex's official API: https://poloniex.com/support/api/.

[3]https://www.kaggle.com/camnugent/sandp500.

Metrics. Following [Shen and Wang, 2017], we use three metrics to evaluate performance. The first is the accumulated portfolio value: $\mathbf{APV}{=}S_n{=}S_0\prod_{t=1}^{n}\mathbf{a}_t^\top\mathbf{y}_t(1-c_t)$, where $S_0{=}1$ is the initial wealth. Such a metric evaluates the profitability when considering the transaction cost. A major drawback of APV is that it neglects the risk factor. To take risk into account, the second metric is the Sharp Ratio: $\mathbf{SR}{=}\frac{\text{Average}(r_t-1)}{\text{Standard Deviation}(r_t-1)}$. Although SR considers the volatility of portfolio values, it treats upward and downward movements equally, while downward movements are usually more important. To highlight the influence of downward deviations, the third metric is the Calmar Ratio: $\mathbf{CR}{=}\frac{S_n}{\text{MDD}}$, where MDD denotes the biggest loss from a peak to a trough and is calculated via $\text{MDD}{=}\max_{t:\tau>t}\frac{S_t-S_\tau}{S_t}$.

Implementation Details. RAT is implemented via pytorch. The number of attention heads is set to $H{=}2$, and the dimension of the feature space is set to $d_f{=}12$. In the training process, we adopt Adam optimizer on a single NVIDIA Tesla P40 GPU. The training step is 80000 for crypto-currency data and 20000 for stock data, where the batch size is 128. We set learning rate to 10^{-4} and weight decay of l_2 regularizer to 10^{-7}. The transaction cost rate is 0.25%. The temporal length of the local context is set to $l{=}5$, while the length of the price series is $k{=}30$. We will show the parameter sensitivity analysis of RAT in a future long paper. In addition, the portfolio vector a_0 is initialized by the average assignment. For all RL based methods, results are averaged over 5 runs with random initialization seeds.

5.2 Evaluation on Portfolio Selection

We report the results of all methods in Table 2. Overall, RAT outperforms all other baselines, which demonstrates strong profitability of our method in PS. In comparison, online learning based methods fail to perform well, since these methods do not consider the learning of sequential features for assets, which may lead to unsatisfactory decisions for portfolios. As for deep reinforcement learning methods, ADDPG fails in PS because its Q network is hard to train, leading to a poor portfolio policy. Although EIIE and MTL perform better than ADDPG, They are still worse than RAT-B, since their structure of policy network (*e.g.,* LSTM or CNNs) suffers from limited sequential modeling ability for portfolio prices. Finally, RAT outperforms RAT-B by a large margin in terms of APV, which demonstrates the contribution of the decision-making layer and the leverage operation in PS.

5.3 Ablation Studies

To evaluate the proposed attention components, *i.e.,* context attention (CA) and relation attention (RA), we compare RAT-B with three degenerate variants, *i.e.,* Transformer (without CA and RA), RAT-B-CA (without CA) and RAT-B-RA (without RA). Here, we use RAT-B to eliminate the influence of leverage. According to the results reported in Table 3, both CA and RA contribute to the performance of RAT. This verifies the importance of capturing local sequential patterns and asset correlations. More specifically, CA contributes slightly more than RA, while combining both modules yields a significant improvement in performance.

Algos	Crypto-A			Crypto-B			S&P500		
	APV	SR(%)	CR	APV	SR(%)	CR	APV	SR(%)	CR
UCRP	2.37	4.05	6.32	1.09	1.48	6.73	1.20	11.69	2.34
Anticor	2.49	3.73	6.05	13.38	11.51	68.95	1.24	13.96	4.02
SSR	1.99	2.21	3.16	4.80	2.66	16.48	1.01	3.56	14.59
OLMAR	6.69	4.79	16.44	1117.28	15.07	6506.39	3.64	3.67	36.80
RMR	6.95	4.98	18.25	386.78	14.06	1403.91	2.81	3.14	22.55
ADDPG	5.67 ± 0.68	4.40	11.71	1.40 ± 0.32	1.89	5.88	1.16 ± 0.14	6.86	12.20
EIIE	16.04 ± 1.72	6.87	53.55	903.38 ± 221.03	15.81	5947.74	82.18 ± 4.20	10.30	1080.08
MTL	19.69 ± 2.72	6.90	79.21	961.24 ± 118.03	16.04	6361.52	88.52 ± 15.27	69.59	1095.47
RAT-B	25.06 ± 2.88	6.97	83.20	2058.67 ± 120.10	16.09	13085.71	112.47 ± 6.74	168.22	151827.01
RAT	156.53 ± 14.25	7.13	304.26	180007.24 ± 69765.49	16.42	737505.11	843945.62 ± 199529.36	217.68	9377166.66

Table 2: Performance comparisons on different datasets.

Algos	Crypto-A			Crypto-B			S&P500		
	APV	SR(%)	CR	APV	SR(%)	CR	APV	SR(%)	CR
Transformer	17.20 ± 1.95	6.70	58.35	1404.12 ± 114.15	15.94	9283.34	101.49 ± 24.24	156.65	16782.29
RAT-B-CA	18.22 ± 2.76	6.85	59.91	1529.64 ± 220.92	15.99	9667.82	106.19 ± 17.67	157.25	46252.79
RAT-B-RA	18.52 ± 2.88	6.89	67.22	1573.41 ± 302.01	16.01	9903.57	110.14 ± 4.66	159.21	69379.18
RAT-B	25.06 ± 2.88	6.97	83.20	2058.67 ± 120.10	16.09	13085.71	112.47 ± 6.74	168.22	151827.01

Table 3: Ablation studies, where RAT-B-CA means RAT-B without context attention, and RAT-B-RA means RAT-B without relation attention.

Algos	APV	SR(%)	CR
CNN	14.99 ± 1.18	6.95	54.72
LSTM	16.04 ± 1.72	6.87	53.55
CNN-LSTM	13.13 ± 1.61	6.44	41.02
RAT-B	25.06 ± 2.88	6.97	83.20

Table 4: Evaluation on feature representation abilities on Crypto-A.

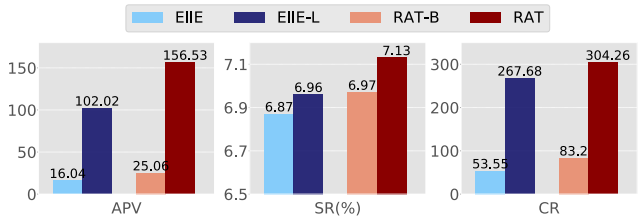

Figure 3: Evaluation of the leverage operation on Crypto-A.

5.4 Evaluation on Feature Representation

To evaluate the representation ability of the proposed method, we compare RAT-B (without the leverage operation) with three variants of RAT equipped with different feature extractors, *i.e.,* CNN, LSTM, and CNN-LSTM (series connection). All variants in these experiments have the same optimization and decision-making mechanisms as RAT yet different network architectures. Specifically, the network architectures of CNN and LSTM use the architectures presented in EIIE [Jiang *et al.*, 2017], while CNN-LSTM follows the network architecture in RWCLDNN [Sainath and others, 2015]. As shown in Table 4, RAT-B outperforms all variants, which verifies the strong series modeling ability of the attention-based model. Moreover, by combining the results in Table 3, we come to a conclusion that the proposed policy network has superior representation ability for asset price series.

5.5 Evaluation on Leverage

We next evaluate the effectiveness of the leverage operation on RAT and another deep reinforcement learning method, *i.e.,* EIIE, named as EIIE-L (with leverage). As shown in Figure 3, the leverage operation is beneficial to both RAT and EIIE, leading to significant improvement in terms of APV and CR. Moreover, the leverage operation also contributes slightly to the SR value. In conclusion, these results demonstrate the effectiveness of the leverage operation and confirm its importance for profitable decision-making for PS.

6 Conclusion

This paper has presented a novel Relation-aware Transformer (RAT) for learning portfolio policy. RAT is structurally novel to simultaneously capture sequential patterns, grasp asset correlations, and make profitable portfolio decisions. By exploiting reinforcement learning to train the policy network (*i.e.,* RAT), our method yields significant performance improvements for portfolio selection. Extensive experiments, on real-world crypto-currency and stock datasets, demonstrate the superiority of RAT. According to the experimental results, we argue that (1) the attention mechanism can be regarded as a dominant scheme for PS, and (2) the leverage operation is of great benefit to the decision making of reinforcement learning based methods for PS.

Acknowledgments

This work was supported by Key-Area Research and Development Program of Guangdong Province (2018B010107001, 2019B010155002, 2019B010155001), National Natural Science Foundation of China (NSFC) 61836003 (key project), 2017ZT07X183, Tencent AI Lab Rhino-Bird Focused Research Program (No. JR201902), Fundamental Research Funds for the Central Universities D2191240.

References

[Agarwal *et al.*, 2006] Amit Agarwal, Elad Hazan, et al. Algorithms for portfolio management based on the newton method. In *ICML*, 2006.

[Atsalakis and others, 2009] George Atsalakis et al. Forecasting stock market short-term trends using a neuro-fuzzy based methodology. *Expert Systems with Applications*, 2009.

[Borodin *et al.*, 2004] Allan Borodin, Ran El-Yaniv, and Vincent Gogan. Can we learn to beat the best stock. In *NeurIPS*, 2004.

[Cao *et al.*, 2019] Jiezhang Cao, Langyuan Mo, Yifan Zhang, et al. Multi-marginal wasserstein gan. In *Advances in Neural Information Processing Systems*, pages 1774–1784, 2019.

[Cho *et al.*, 2014] Kyunghyun Cho, Bart Van Merriënboer, Dzmitry Bahdanau, and Yoshua Bengio. On the properties of neural machine translation: Encoder-decoder approaches. *arXiv*, 2014.

[Cover and others, 1991] Thomas M Cover et al. Universal portfolios. In *Mathematical Finance*. World Scientific, 1991.

[Das *et al.*, 2014] Puja Das, Nicholas Johnson, et al. Online portfolio selection with group sparsity. In *AAAI*, 2014.

[Deng and others, 2016] Yue Deng et al. Deep direct reinforcement learning for financial signal representation and trading. *IEEE Transactions on Neural Networks and Learning Systems*, 2016.

[Frye, 2008] Jon Frye. Correlation and asset correlation in the structural portfolio model. *The Journal of Credit Risk*, 4, 2008.

[Guo *et al.*, 2018] Yifeng Guo, Xingyu Fu, et al. Robust log-optimal strategy with reinforcement learning. *arXiv*, 2018.

[Györfi and Vajda, 2008] László Györfi and István Vajda. Growth optimal investment with transaction costs. In *COLT*, 2008.

[Hochreiter and others, 1997] Sepp Hochreiter et al. Long short-term memory. *Neural Computation*, 1997.

[Huang *et al.*, 2013] Dingjiang Huang, Junlong Zhou, Bin Li, Steven HOI, and Shuigeng Zhou. Robust median reversion strategy for on-line portfolio selection. In *IJCAI*, 2013.

[Jiang *et al.*, 2017] Zhengyao Jiang, Dixing Xu, and Jinjun Liang. A deep reinforcement learning framework for the financial portfolio management problem. *arXiv*, 2017.

[Kang *et al.*, 2018] Qinma Kang, Huizhuo Zhou, et al. An asynchronous advantage actor-critic reinforcement learning method for stock selection and portfolio management. In *ICBDR*, 2018.

[Kelly, 1956] JL Kelly. A new interpretation of information rate. *Bell System Technical Journal*, 1956.

[LeCun *et al.*, 2015] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *Nature*, 521, 2015.

[Li and Hoi, 2012] Bin Li and Steven CH Hoi. On-line portfolio selection with moving average reversion. *arXiv*, 2012.

[Li and Hoi, 2014] Bin Li and Steven CH Hoi. Online portfolio selection: A survey. *ACM Computing Surveys*, 46, 2014.

[Li and Ng, 2000] Duan Li and Wan-Lung Ng. Optimal dynamic portfolio selection: Multiperiod mean-variance formulation. *Mathematical Finance*, 2000.

[Li *et al.*, 2019a] Jianquan Li, Xiaokang Liu, et al. An empirical evaluation of multi-task learning in deep neural networks for natural language processing. *arXiv*, 2019.

[Li *et al.*, 2019b] Shiyang Li, Xiaoyong Jin, et al. Enhancing the locality and breaking the memory bottleneck of transformer on time series forecasting. In *NeurIPS*, 2019.

[Liang and others, 2018] Zhipeng Liang et al. Adversarial deep reinforcement learning in portfolio management. *arXiv*, 2018.

[Lillicrap and others, 2016] Timothy Lillicrap et al. Continuous control with deep reinforcement learning. *ICLR*, 2016.

[Lopez, 2004] Jose A Lopez. The empirical relationship between average asset correlation, firm probability of default, and asset size. *Journal of Financial Intermediation*, 13, 2004.

[Mandelker and others, 1984] G Mandelker et al. The impact of the degrees of operating and financial leverage on systematic risk of common stock. *Financial and Quantitative Analysis*, 1984.

[Moody and Saffell, 2001] John Moody and Matthew Saffell. Learning to trade via direct reinforcement. *IEEE Transactions on Neural Networks*, 2001.

[Neely *et al.*, 2014] Christopher J Neely, David E Rapach, Jun Tu, and Guofu Zhou. Forecasting the equity risk premium: the role of technical indicators. *Management Science*, 60, 2014.

[Neuneier, 1998] Ralph Neuneier. Enhancing q-learning for optimal asset allocation. In *NeurIPS*, 1998.

[Niu *et al.*, 2020] Shuaicheng Niu, Jiaxiang Wu, Yifan Zhang, Yong Guo, Peilin Zhao, et al. Disturbance-immune weight sharing for neural architecture search. *arXiv*, 2020.

[Poterba and Summers, 1988] James M Poterba and Lawrence H Summers. Mean reversion in stock prices: Evidence and implications. *Journal of Financial Economics*, 22, 1988.

[Sainath and others, 2015] Tara Sainath et al. Learning the speech front-end with raw waveform cldnns. In *INTERSPEECH*, 2015.

[Schulman and others, 2017] John Schulman et al. Proximal policy optimization algorithms. *arXiv*, 2017.

[Shen and others, 2015] Weiwei Shen et al. Portfolio choices with orthogonal bandit learning. In *IJCAI*, 2015.

[Shen and Wang, 2016] Weiwei Shen and Jun Wang. Portfolio blending via thompson sampling. In *IJCAI*, 2016.

[Shen and Wang, 2017] Weiwei Shen and Jun Wang. Portfolio selection via subset resampling. In *AAAI*, 2017.

[Stefanova and Elkamhi, 2011] Denitsa Stefanova and Redouane Elkamhi. Dynamic correlation or tail dependence hedging for portfolio selection. In *AFA Chicago Meetings Paper*, 2011.

[Sutton *et al.*, 2000] Richard S Sutton, David A McAllester, et al. Policy gradient methods for reinforcement learning with function approximation. In *NeurIPS*, 2000.

[Tasca *et al.*, 2017] Paolo Tasca, Stefano Battiston, and Andrea Deghi. Portfolio diversification and systemic risk in interbank networks. *Journal of Economic Dynamics and Control*, 2017.

[Vaswani *et al.*, 2017] Ashish Vaswani, Noam Shazeer, et al. Attention is all you need. In *NeurIPS*, 2017.

[Zhang *et al.*, 2018] Yifan Zhang, Peilin Zhao, Jiezhang Cao, et al. Online adaptive asymmetric active learning for budgeted imbalanced data. In *SIGKDD*, pages 2768–2777, 2018.

[Zhang *et al.*, 2019] Yifan Zhang, Ying Wei, et al. Collaborative unsupervised domain adaptation for medical image diagnosis. In *Medical Imaging meets NeurIPS*, 2019.

[Zhang *et al.*, 2020] Yifan Zhang, Peilin Zhao, et al. Cost-sensitive portfolio selection via deep reinforcement learning. *IEEE Transactions on Knowledge and Data Engineering*, 2020.

[Zhao *et al.*, 2018] Peilin Zhao, Yifan Zhang, et al. Adaptive cost-sensitive online classification. *IEEE Transactions on Knowledge and Data Engineering*, 31(2):214–228, 2018.