Theme Article: AI and FinTech

An Asymptotic Statistical Learning Algorithm for Prediction of Key Trading Events

Jianfei Yin
Shenzhen University

Ruili Wang
Massey University

Shunda Ju
Shenzhen University

Yizhe Bai
Shenzhen University

Joshua Zhexue Huang
Shenzhen University

***Abstract*—In financial trading, trading at key time points (i.e., the best times for buying or selling in specific contexts) is one of the most effective trading methods. Since key points are random and sparse, their randomness needs to be investigated to predict key points reliably. Based on measure theory, we propose an event mapping model to formally define this randomness and express it in a low dimensional and normalized space $\mathcal{Q}$ through mapping functions. Interestingly, when mapping functions are monotonic, the probability of a key event can be approximated by those of nested random events in $\mathcal{Q}$. Based on this finding, we designed a long short-term memory based neural network to learn the mapping functions and derived an asymptotic statistical learning (ASL) algorithm. ASL automatically analyzes the convergence of random events in space $\mathcal{Q}$ and makes point estimates on key events in trading. Compared with 12 existing algorithms on six real datasets from NYSE, S&P 500, NASDAQ, cryptocurrency markets, etc., the algorithm ASL reliably predicts sparse key events from many random events, which provides a method to deal with the imbalance classification problem in predicting key events in trading. ASL significantly outperforms other algorithms under different market situations in both bull and bear markets.**

Digital Object Identifier 10.1109/MIS.2020.2977590
Date of publication 2 March 2020; date of current version 12 May 2020.

Designing effective algorithms for financial trading to make profits has long been a broad and in-depth research topic.[1–12] From the perspective of stochastic optimization,[13] the volatility and noise in financial sequences pose an essential challenge for optimal trading: *how to achieve high returns and low risks in situations that are unknown at the time of decision-making[10] but will be known in the future*?

The first solution is the prediction of trading actions.[6,9,14,15] Trading actions are operations such as buying, selling, and holding. One popular method of prediction of trading actions is applying deep reinforcement learning[16] to map financial sequences directly to trading actions.[6,14,15] There are two problems in this popular method. First, considering that optimal buying and selling actions are random and sparse events, predicting trading actions at every time point will output a lot of holding actions[4,6] and still get a high prediction accuracy. Second, considering the statistical correlation of financial subsequences, different subsequences have different prediction effects on action predictions. One should split a financial sequence into multiple subsequences beforehand, instead of feeding the whole sequence into neural networks for training and predicting.

The second solution is price prediction.[7,8,17] It is assumed that one can make optimal trading if he/she can make accurate price prediction. However, accurate price prediction is not enough for optimal trading, because trading loss can be large if one does not know price trends[17] and makes error buying or selling at wrong times. Even one can predict price trends,[17] she can still loss largely if she cannot accurately predict the random time points at which trend reversals will happen.[3,4]

The third solution is turning point prediction.[3,4,18–20] Turning points are such time points that mark reversals of trends.[3,4] There are some ambiguities about turning points. Because turning points are defined by algorithms[3,4,19] such as using piecewise linearization approximation and support vector machine (SVM)[21] based classification, different turning points can be obtained in two runs of the same algorithm on the same context subsequences. Turning points found by using the methods[3,4,19] can contain a lot of suboptimal points, which are not true turning points. Because true turning points are random and sparse in financial sequences, turning point prediction can be thought as an instance of imbalanced classification problem.[22]

Based on the above analysis, first we propose a concept of key point from the view of optimal trading. Since optimal trading is context dependent, in this article, *key points are defined as the best time points to buy or sell in specific context subsequences*. Key points are very close to turning points conceptually, but definition of key points captures three important concerns, i.e., randomness, sparseness, and context-dependences, since key points are optimal random events in the optimal trading. Thus, second we study the randomness of key points and make prediction of key points. The sparseness of key points is handled in our prediction algorithm. The context-dependence of key points is handled in the formal definition of key points in the "Context Subsequences for Key Points" section.

We summarize our main contributions as follows:

- The concept of key points is formally defined in specific context subsequences, namely round bottom (RB) and round top (RT) subsequences for buying and selling, respectively.
- Based on measure theory,[23] an event mapping model is proposed to formally define the randomness of key points and express it as *key events* in a space $\mathcal{Q}$ through mapping functions.
- When mapping functions are monotonic, the probability of a key event can be approximated by those of nested random events in $\mathcal{Q}$. Based on this finding, we designed a long short-term memory (LSTM)[24] based neural network to learn the mapping functions and derived an asymptotic statistical learning (ASL) algorithm to predict sparse key points.

As such, our solution provides a method to deal with the imbalance classification problem of predicting key points. Compared with 12 different algorithms on six real datasets from NYSE, S&P 500, NASDAQ, and cryptocurrency markets, algorithm ASL significantly outperforms the 12 algorithms under different market situations such as in bull and bear markets.

The rest of this article is organized as follows. The section "Related Work" reviews some related work on financial trading. The section "Context Subsequences for Key Points" defines context subsequences for key points. The section "From key Points to Key Events" gives the event mapping model. The section "ASL algorithm" presents our algorithm ASL. The section "Experiments and Analysis" presents the convergence and performance comparison experiments. The section "Conclusions and Future Work" concludes this article with some future studies.

RELATED WORK

The prediction of timing of trading focuses on how to effectively predict the optimal time of buying and selling in order to achieve high returns and low risks.[3,4,18–20] A typical example of trading timing is the turning point prediction, which marks the trend reversal points in financial sequences.[3,4,18] Luo *et al.*[3,4] investigated turning point prediction by using piecewise linearization and support vector machine (SVM)[21] based classification, but the turning points obtained were noisy. By using a Long Short-Term Memory[24] based architecture, JuHyok *et al.*[18] used upward/downward reversal point feature sets to predict reversal points, which were almost the same as turning points by definition. Leung and Li[20] studied optimal timing strategies for pair trading and derived the optimal price intervals for market entry and exit by applying a probabilistic methodology. Moews *et al.*[19] proposed the use of lagged correlation among different time series to predict trend changes on a larger time scale than timing of trading.

The prediction of trading actions of buying, selling, and holding provides an end-to-end solution for financial trading.[6,9,14,15] The deep reinforcement learning based trading[6,14,15] can predict trading actions directly from financial sequences, since the reward computed from future returns is a kind of weakly supervised signals.[25] Future returns can be estimated by using price prediction.[8,26] One challenge of reinforcement learning is that the reward functions designed should match the problem. Deng *et al.*[14] proposed a deep recursive network combined with a direct reinforcement learning to obtain trading actions from financial sequences. By using transfer learning to reduce the need for large amounts of training data, Jeong *et al.*[6] designed a deep Q-learning network that can predict trading actions and shares to trade. Portfolio selection algorithms[1,2,10] can also be considered as a kind of action prediction, since they predict the wealth allocation for assets at the beginning of each trading period.

Almost any prediction task is a problem of imbalanced classification.[22] The general solutions are oversampling, undersampling,[27] data augmentation,[28] etc. In this article, multiple random sample points are used to estimate key events, which can be considered as a method related to oversampling and data augmentation.

CONTEXT SUBSEQUENCES FOR KEY POINTS

Suppose a financial sequence $\mathcal{X}$ can be represented as $\mathcal{X} = \{\boldsymbol{x}_i : \boldsymbol{x}_i = (o_i, c_i, h_i, l_i, a_i)\}_{i=1}^n$, where o_i, c_i, h_i, l_i, and a_i are the open, close, highest and lowest prices, and trading volume at period i, respectively. This section defines two types of subsequences: RB and RT subsequences, to define key buy points and key sell points, respectively.

We first define an auxiliary sequence $\mathcal{S} = \{s_i : s_i \in \mathbb{R}\}_{i=1}^n$, such that

$$s_i = \beta \mathrm{RSI}_k(c_{i-k+1}^i) + (1-\beta)\mathrm{RSI}_k(a_{i-k+1}^i), \beta \in [0,1] \tag{1}$$

where RSI_k denotes the relative strength index indicator;[26] $k \in \mathbb{N}$ is the backtracking parameter, and is usually set to $10, 14, 30$.

Through combining $\mathcal{S}$ and $\mathcal{C} = \{c_i\}_i^n$, we define the ith RB subsequence $\mathcal{B}_i$ as

$$\mathcal{B}_i := \mathcal{X}_{D_i}^{U_i} \tag{2}$$

where D_i, L_i, and U_i are three time points, as shown in Figure 1. The first point D_i is defined by

$$\begin{aligned} D_i &= \underset{t\in[t_1,t_2]}{\arg\min}\{e^{\lambda(t_2-t)}c_t\} \\ t_1 &= \min\{t : s_t > \alpha, t \in [L_{i-1}, U_{i-1}]\} \\ t_2 &= \min\{t : s_t < \alpha - \epsilon_D, t \in [U_{i-1}, U_{i-1} + T_B]\} \end{aligned} \tag{3}$$

where $\lambda \in [0,1]$ and $\epsilon_D > 0$ are small constants; $\alpha \in [60,100]$ is a threshold; T_B is a window constant. Initially, we set $L_0 = 0$ and $U_0 = n$.

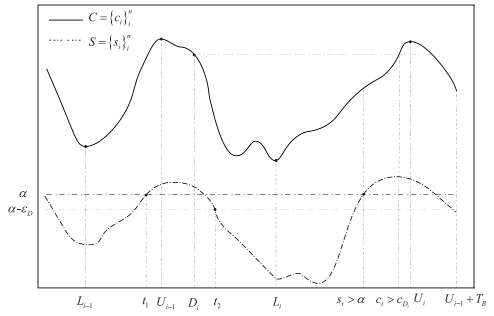

Figure 1. Time points D_i, L_i, and U_i used to define a RB subsequence $\mathcal{B}_i$.

Base on the definition of D_i, the points U_i and L_i are

$$\begin{aligned} U_i &= \underset{t\in[U_{i-1},U_{i-1}+T_B]}{\arg\max}\{c_t : s_t > \alpha, c_t > c_{D_i}\} \\ L_i &= \underset{t\in[D_i,U_i]}{\arg\min}\{c_t\}. \end{aligned} \tag{4}$$

For any RB subsequence $\mathcal{B}_i$, we know that the key buy point is at L_i according to (4). Once two adjacent key buy points L_{i-1} and L_i are obtained, we can define the ith RT subsequence $\mathcal{T}_i$ as

$$\mathcal{T}_i := \mathcal{X}_{L_{i-1}}^{L_i} \tag{5}$$

and the key sell point is at

$$T_i = \underset{t\in[L_{i-1},L_i]}{\arg\max}\{c_t\}. \tag{6}$$

Some examples of RB/RT subsequences are shown in Figure 2.

FROM KEY POINTS TO KEY EVENTS

Considering the randomness of the occurrences of key points in RB and RT subsequences, this section establishes an event mapping model based on measure theory.[23] The model formally defines this randomness and represents it in space $\mathcal{Q}$ through the mapping functions. Under the formalism, we also present a reference implementation of the mapping functions based on neural network learning to serve key point prediction.

Event Mapping Model

The event mapping model consists of the source space $\mathcal{M}$, the image space $\mathcal{Q}$, and the mapping functions f and g between them.

Given any financial sequence $\mathcal{X} \in \mathbb{R}^{n\times 5}$, we define the source space $\mathcal{M} = (\Omega \subset \mathbb{R}^5, \mathcal{A}, \mu)$, where $\mathcal{X} \subset \Omega$; $\mathcal{A}$ is a set of measurable events; $\mu : \mathcal{A} \to \mathbb{R}$ is a measure function for events. It is assumed that subsequences $\mathcal{X}_i^j, i \le j$ are μ-measurable events, i.e., $\mathcal{X}_i^j \in \mathcal{A}$.

From above definition, we know that any key point $L_i \in \mathcal{B}_i$ ($T_i \in \mathcal{T}_i$) can be represented as a buy (sell) event $\mathcal{X}_{L_{i-1}}^{L_i}$ ($\mathcal{X}_{T_{i-1}}^{T_i}$) in space $\mathcal{M}$.

Considering the data noise and the dimension of $\mathcal{M}$, we define the mapping function $f : \mathcal{A} \to \mathcal{D}$ to map events of $\mathcal{M}$ to those of the image space $\mathcal{Q} = (\mathcal{Y} \subset \mathbb{R}^3, \mathcal{D}, \mathcal{P})$, where the function f is required to have the monotonic property.

Definition 1. *$f(\mathcal{X}_i^j) \subseteq f(\mathcal{X}_{i'}^{j'})$, if $\mathcal{X}_i^j \subseteq \mathcal{X}_{i'}^{j'}$.*

For each buy event $\mathcal{X}_{L_{i-1}}^{L_i}$, we call the image event $f(\mathcal{X}_{L_{i-1}}^{L_i})$ the key buy event and have following main theorem.

Theorem 1. *Given any RB subsequence $\mathcal{B}_i$, we construct a set of nested events $V_k = \mathcal{X}_{L_{i-1}}^k$ to*

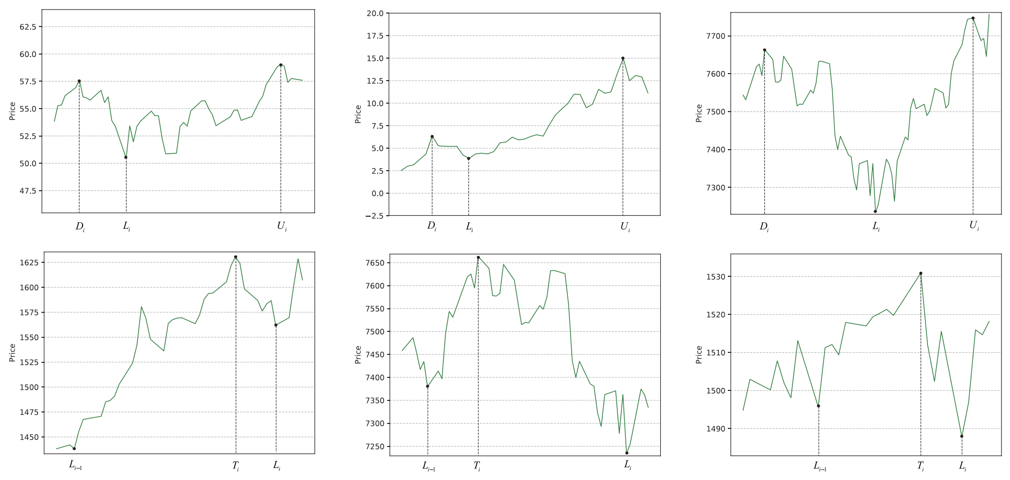

Figure 2. Close prices of three RB subsequences (top three) and three RT subsequences (bottom three).

approach the buy event $V_* = \mathcal{X}_{L_{i-1}}^{L_i}$. *Let* $k = L_{i-1}, L_{i-1}+1, \ldots, L_i,\ k' = U_i, U_i - 1, \ldots, L_i$, *we have*

$$\begin{aligned} & V_k \uparrow V_*,\ V_{k'} \downarrow V_* \\ & \quad \Rightarrow \mu(V_k) \uparrow \mu(V_*),\ \mu(V_{k'}) \downarrow \mu(V_*) \\ & \quad \Rightarrow \mathcal{P}(f(V_k)) \uparrow \mathcal{P}(f(V_*)),\ \mathcal{P}(f(V_{k'})) \downarrow \mathcal{P}(f(V_*)). \end{aligned}$$

Here, the binary relationship $X_k \uparrow X_*$ represents $X_k \le X_{k+1}$ and $X_k = X_*$ when $k = L_i$; $X_{k'} \downarrow X_*$ represents $X_{k'} \ge X_{k'-1}$ and $X_{k'} = X_*$ when $k' = L_i$.

Proof. The first inference follows from applying the monotonic convergence axiom of measure spaces.[23] Since the image measure $\mathcal{P}$ is a derived measure, let $\mathcal{P}(f(V)) = \inf_{W: f(V) \subseteq f(W)} \mu(W)$. If $V_k \subseteq V_{k'}$, then by Definition 1, $f(V_k) \subseteq f(V_{k'})$. Thus,

$$\inf_{W_k: f(V_k) \subseteq f(W_k)} f(W_k) \subseteq f(V_{k'}) \subseteq \inf_{W_{k'}: f(V_{k'}) \subseteq f(W_{k'})} f(W_{k'}).$$

By the monotonic property of μ, we have $\mathcal{P}(f(V_k)) \le \mathcal{P}(f(V_{k'}))$ and the second inference follows. ■ □

Similarly, we can define another mapping $g: \mathcal{A} \to \mathcal{D}$, which follows the same definition as Definition 1 to map sell events in $\mathcal{M}$. We call the image events $g(\mathcal{X}_{T_{i-1}}^{T_i})$ the key sell events.

Neural Network for Event Mapping

This section gives a reference implementation of the mapping functions f and g. The implementation is based on the LSTM[24] based network to learn our optimal objectives, so as to learn the convergence pattern of key points from many samples of RB/RT subsequences.

We first derive an optimal objective for $f: \mathbb{R}^{M \times 5} \to \mathbb{R}^3$. For any event $\mathcal{X}_{L_{i-1}}^j$ of space $\mathcal{M}$,

$$f(\mathcal{X}_{L_{i-1}}^j; \boldsymbol{\theta}) = \hat{\boldsymbol{y}}_j^i = (\hat{\rho}_j^i, \hat{\eta}_j^i, \hat{\gamma}_j^i)^T,\ j \in [L_{i-1}, L_i] \tag{7}$$

where $\boldsymbol{\theta}$ are parameters to be learned by the neural network defined in (11); $\hat{\boldsymbol{y}}_j^i \in \mathbb{R}^3$ are image events in space $\mathcal{Q}$. $\hat{\boldsymbol{y}}_j^i$ are estimates of the key buy event $\boldsymbol{y}^i = (\rho^i, \eta^i, \gamma^i)$ defined by

$$\rho^i = 1 - \frac{c_{L_i}}{c_{U_{i-1}}}, \eta^i = \frac{\sum_{k=L_{i-1}}^{U_{i-1}} a_k}{\sum_{k=L_{i-1}}^{L_i} a_k}, \gamma^i = \frac{U_{i-1} - L_{i-1}}{L_i - L_{i-1}}. \tag{8}$$

Note that ρ^i, η^i and γ^i are normalized variables related to prices, trading volumes, and time points, respectively. Given the price factor ρ^i of $\boldsymbol{y}^i$, when the close price $c_{U_{i-1}}$ is known at point U_{i-1}, we can predict the best buy price c_{L_i} at the future key point L_i after the time passes U_{i-1}. The components η^i and γ^i of $\boldsymbol{y}^i$ can be analyzed similarly.

By (7) and (8), we have the optimal objective to train $f(\mathcal{X}_{L_{i-1}}^j; \boldsymbol{\theta})$ as

$$\min_{\boldsymbol{\theta}} \frac{1}{N} \sum_{i=1}^{N} \frac{1}{M} \sum_{j=L_{i-1}}^{L_i} w_{i,j} \left\| \hat{\boldsymbol{y}}_j^i - \boldsymbol{y}^i \right\|_2^2 \tag{9}$$

where $\|\|_2^2$ denotes the square of l^2-norm; N is the number of RB subsequences; $M = L_i - L_{i-1} + 1$ is the number of nested subsequences; $w_{i,j} = \tan\left(\frac{j\pi}{2(j+1)} \frac{j - L_{i-1}}{L_i - L_{i-1}}\right)$ are weights to strengthen the need for the monotonicity defined in Definition 1: the greater the j the smaller the loss.

Similarly, given any RT subsequence $\mathcal{T}_i$, let $\hat{\boldsymbol{y}}_j^i = g(\mathcal{X}_{T_{i-1}}^j; \boldsymbol{\vartheta}), j \in [T_{i-1}, T_i]$. The key sell event $\boldsymbol{y}^i$ is

$$\rho^i = 1 - \frac{c_{L_{i-1}}}{c_{T_i}}, \eta^i = \frac{\sum_{k=T_{i-1}}^{L_{i-1}} a_k}{\sum_{k=T_{i-1}}^{T_i} a_k}, \gamma^i = \frac{L_{i-1} - T_{i-1}}{T_i - T_{i-1}}. \tag{10}$$

The optimal objective for g is the same as (9) except that N is the number of RT subsequences, M is $T_i - T_{i-1} + 1$, and $w_{i,j}$ is $\tan\left(\frac{j\pi}{2(j+1)} \frac{j - T_{i-1}}{T_i - T_{i-1}}\right)$.

With the optimization objectives such as (9), we design the following LSTM[24] based network as the realization of $f(\mathcal{X}_{L_{i-1}}^j; \boldsymbol{\theta})$ and $g(\mathcal{X}_{T_{i-1}}^j; \boldsymbol{\vartheta})$. For $t \in [1, M]$:

$$\boldsymbol{h}_t^1, \boldsymbol{c}_t^1 = \text{lstm}_1(\boldsymbol{x}_t, \boldsymbol{h}_{t-1}^1, \boldsymbol{c}_{t-1}^1,) \tag{11a}$$

$$[\boldsymbol{o}_1, ..., \boldsymbol{o}_M] = \text{BatchNorm} \circ \text{Dropout}([\boldsymbol{h}_1^1, ..., \boldsymbol{h}_M^1]) \tag{11b}$$

$$\boldsymbol{h}_t^2, \boldsymbol{c}_t^2 = \text{lstm}_2(\boldsymbol{o}_t, \boldsymbol{h}_{t-1}^2, \boldsymbol{c}_{t-1}^2) \tag{11c}$$

$$\boldsymbol{z}_1 = \text{Dropout} \circ \text{Prelu}(\boldsymbol{h}_M^2) \tag{11d}$$

$$\boldsymbol{z}_2 = \underset{32\times128}{\boldsymbol{W}_1} \underset{128\times1}{\boldsymbol{z}_1} + \underset{32\times1}{\boldsymbol{b}_1} \tag{11e}$$

$$\boldsymbol{z}_3 = \text{Dropout}(\boldsymbol{z}_2) \tag{11f}$$

$$\boldsymbol{y} = \underset{3\times32}{\boldsymbol{W}_2} \underset{32\times1}{\boldsymbol{z}_3} + \underset{3\times1}{\boldsymbol{b}_2} \tag{11g}$$

where $\boldsymbol{x}_t \in \mathbb{R}^5$ is an event in space $\mathcal{Q}$, which is the input of the network; $\boldsymbol{h}_t^1, \boldsymbol{c}_t^1 \in \mathbb{R}^{64}$ and

$h_t^2, c_t^2 \in \mathbb{R}^{128}$ are parameters of the network. The main idea is to encode an $\mathcal{Q}$'s event $[x_1, ..., x_t, ..., x_M]$ into a vector $h_M^2 \in \mathbb{R}^{128}$ by using LSTM[24] cells (11a) and (11c), and then output the vector y.

ASL ALGORITHM

Since the mapping functions f and g can transform a context subsequence into multiple events in space $\mathcal{Q}$, i.e., random sample points, this section presents the algorithm ASL to analyze the convergence of each set of random sample points and make a point estimate for each key event.

Considering the procedures of buying and selling are similar, we put their shared part into ASL and extract their own state variables outside ASL, i.e., the first part of the input of Algorithm 1. For example, we can invoke Algorithm 1 with $tobuy = \text{True}$, $S_1 = L$, $S_2 = U$, and $\kappa = f$ to decide whether the current time point is a key buy point.

Algorithm 1. Algorithm ASL

```
Input:
  1. Financial sequence $\mathcal{X}$, current time index $j$, mappings $\kappa \in \{f, g\}$, types of time points $S_1 \in \{L, T\}$, $S_2 \in \{U, L\}$, close price $c_{S_2}$ at points of type $S_2$, queue $q$ of $\hat{\rho}$, set $\bar{q}$ of $\hat{\rho}$'s means, $tobuy \in \{\text{True}, \text{False}\}$; RSI[26] indicator $rsi$
  2. Mini-sample size $\delta \in [10, 100]$ of $\hat{\rho}$, percentile $p \in [1, 100]$, shift parameter $\Delta \in [-10, 10]$, RSI[26] conditions $rsi_b$, $rsi_s$
Output: $buy\ now$; $sell\ now$
1  begin
2    let $t$ = backtracking $S_1$ in $\mathcal{X}_1^j$
3    if not $t$ then return if $t+1 \geq j$ then $q = [\ ]$ put $\kappa(\mathcal{X}_t^j)[1]$ in $q$; let $l = \text{length}(q)$
4    if $l \bmod \delta = 0$ then
5        put the mean of $q[l-\delta+1, l]$ in $\bar{q}$
6    let $t$ = backtracking $S_2$ in $\mathcal{X}_1^j$
7    if not $t$ then return if $t+1 \geq j$ then
8        $\bar{q} = \bar{q}[\text{length}(\bar{q}) + \Delta, \text{length}(\bar{q})]$
9        $c_{S_2} = \mathcal{X}_t[2]$
10    let $n$ be $p$th percentile of $\bar{q}$
11    let $\underline{\rho}$ be the mean of $n$ smallest elements in $\bar{q}$
12    if $tobuy$ and $rsi < rsi_b$ then
13        compute $\hat{c}_{L_i}$ by using (8) on $\underline{\rho}$ and $c_{S_2}$
14        if $\mathcal{X}_j[2] < \hat{c}_{L_i}$ then outputs $buy\ now$
15    if not $tobuy$ and $rsi > rsi_s$ then
16        compute $\hat{c}_{T_i}$ by using (10) on $\underline{\rho}$ and $c_{S_2}$
17        if $\mathcal{X}_j[2] > \hat{c}_{T_i}$ then outputs $sell\ now$
18    end
```

The second part of the input of Algorithm 1 is the hyper-parameters of ASL. Considering the sparseness of key events, we design a lower bound estimation for the price factor ρ of any key event and the result is stored in the variable $\underline{\rho}$. Step 7 puts the mean of the most recent δ estimates $\hat{\rho} = \kappa(\mathcal{X}_t^j)[1]$ into the set $\bar{q}$. Steps 13 and 14 sample the pth percentile samples of $\bar{q}$ and store their mean in $\underline{\rho}$. We use a shift parameter Δ to filter out some outliers in Step 11, so as to obtain a stable sample of means. We use two RSI[26] conditions rsi_b and rsi_s to prevent buying and selling from premature convergence of the empirical distribution of ρ.

EXPERIMENTS AND ANALYSIS

We present two experiments and their analysis in this section. The first demonstrates that the mapping function realized by the neural network (11) satisfies the convergence property defined in Theorem 1. The second compares the trading performance and behaviors of 13 algorithms on six datasets.

Datasets and Evaluation Measure

The six datasets were randomly selected from six sources: NYSE Composite, AAPL, S&P 500 Index, TSLA, ETH-USD, and NASDAQ Composite.[†] Each dataset was divided into three parts: the training, validation, and testing datasets as shown in Table 1.

We use the three indices to compare performance: the rate of return (RoR), the sharpe ratio (SR), and the maximum drawdown (MDD).

RoR is defined as

$$\text{RoR} := \frac{v_f - v_i}{v_i} \times 100$$

where v_f is the final value and v_i is the initial value of the asset.

SR is defined as

$$\text{SR} := \frac{\mathbb{E}[r_i] - r_f}{\sigma[r_i]} \times \sqrt{252}$$

where r_i is the random variable of the rate of return of the asset; $\mathbb{E}[r_i]$ is the expectation of r_i; $\sigma[r_i]$ is the standard deviation of r_i; r_f is the risk-

[†]All six datasets can be downloaded from https://finance.yahoo.com/

Table 1. Summary of datasets.

Dataset	Training	Validation	Test	Source
D_1	[2005.05.07, 2009.06.25]	[2009.06.26, 2010.07.08]	[2010.07.09,2015.07.09]	NYSE Composite
D_2	[2011.01.08, 2012.08.25]	[2012.08.26, 2013.01.11]	[2013.01.12, 2015.01.09]	AAPL
D_3	[2013.01.11, 2014.07.26]	[2014.07.27, 2014.12.14]	[2014.12.15, 2016.10.21]	S&P 500 Index
D_4	[2011.01.02, 2013.07.03]	[2013.07.04, 2014.02.17]	[2014.02.18, 2017.02.24]	TSLA
D_5	[2015.12.17, 2017.08.03]	[2017.08.04, 2017.12.30]	[2017.12.31, 2019.05.10]	ETH-USD
D_6	[1997.06.17, 1999.08.09]	[1999.08.10, 2000.02.21]	[2000.02.22, 2002.09.27]	NASDAQ Composite

free rate of return; and the default value for r_f here is 0.02.

MDD is defined as

$$\text{MDD} := \max_{\tau\in(0,n)}\left\{\max_{t\in(0,\tau)}\frac{v_t - v_\tau}{v_\tau}\right\}$$

representing the maximum drop ratio of the value v of the asset in the period $[0, n]$.

Parameter Settings

Algorithm ASL is compared with the other 12 algorithms, one of which is the benchmark algorithm UBAH (Uniform Buy And Hold); six are the machine learning style algorithms: NE,[9] PLR-WSVM,[4] AC,[15] QL,[6] AR,[8] and MIE;[7] five are the portfolio selection style algorithms: UP,[2] EG,[2] ONS,[2] A1,[2] and CFR-OGD.[1] The reason for considering portfolio selection algorithms is that the trading of a single risky asset can be seen as a special case of portfolio selection. The transaction fee is set to 2 % for both buying and selling actions taken by all algorithms. The parameter settings of the algorithms are as follows:

- *ASL:* Our ASL algorithm: minisample size $\delta = 50$, percentile $p = 2$, shift parameter $\Delta = -2$, RSI[26] conditions $rsi_b = 40$, and $rsi_s = 60$ under RSI_{24}.
- *UBAH:* The Market strategy, i.e., the uniform BAH (Buy And Hold) approach.
- *NE:* An evolution computing based algorithm[9]: the population size is 512; the overall mutation rate is 0.2; the crossover probability is 0.75;
- *PLR-WSVM:* A turning point prediction algorithm:[4] $\text{RSI}_{24}(t) < 40$ for buying, $\text{RSI}_{24}(t) > 60$ for selling; $\alpha_{\text{PLR}} = 0.05$, $\beta_{\text{PLR}} = 5$, $nw_{op} = 1$, and $nw_{tp} = 1$.
- *AC:* An actor critic based algorithm:[15] num_units $= 128$, keep_prob $= 0.5$, learning rate $\eta = 0.001$, exponential decay rates $\beta_1 = 0.9$, $\beta_2 = 0.999$, and $\epsilon = 10^{-8}$.
- *QL:* A Q-learning based algorithm[6]: learning rate is 0.00001; batch size is 64; γ is 0.85.
- *AR:* An attention based LSTM[24] algorithm:[8] $p = q = m = 64$, $T = 10$, $T_r = 20$; P and N are equal and chosen from $\{4, 6, 8, 10, 15, 20\}$.
- *MIE:* A neural network algorithm using multiple technical indicators:[7] layers in $\{4, 5, 4, 3\}$, epochs $= 200$, blocksize $= 128$, and seed $= 1234$.
- *UP:* Universal Portfolio algorithm:[2] $\delta_0 = 0.004$, $\delta = 0.005$, $m = 100$, and $S = 500$.
- *EG:* Exponential Gradient algorithm:[2] $\eta = 0.05$.
- *ONS:* Online Newton Step:[2] $\eta = 0$, $\beta = 1$, and $\gamma = 1/8$.
- *A1:* AntiCor algorithm. A1 is called BAH_{30} (Anticor) in the OLPS tool.[2]
- *CFR-OGD:* Combination Forecasting Reversion strategy for Online Gradient Descent:[1] $\epsilon = 10$ and $w = 5$.

Convergence Analysis

This experiment demonstrates the effect of Theorem 1. According to Theorem 1, we randomly picked a RB subsequence $\mathcal{B}_i$ from a test dataset and constructed a set of nested subsequences V_k from $\mathcal{B}_i$. After calling $f(V_k)$, we obtained the random events shown in Figure 3.

The horizontal axes of subfigures in the top half of Figure 3 are values of random variables $\hat{\rho}, \hat{\eta}, \hat{\gamma}$, and the vertical axes are relative time $t = j - L_{i-1}, j \in [L_{i-1}, U_i]$. The instances can be divided into three groups. The first group $t \in [0, U_{i-1} - L_{i-1} = 152)$ represents the migration

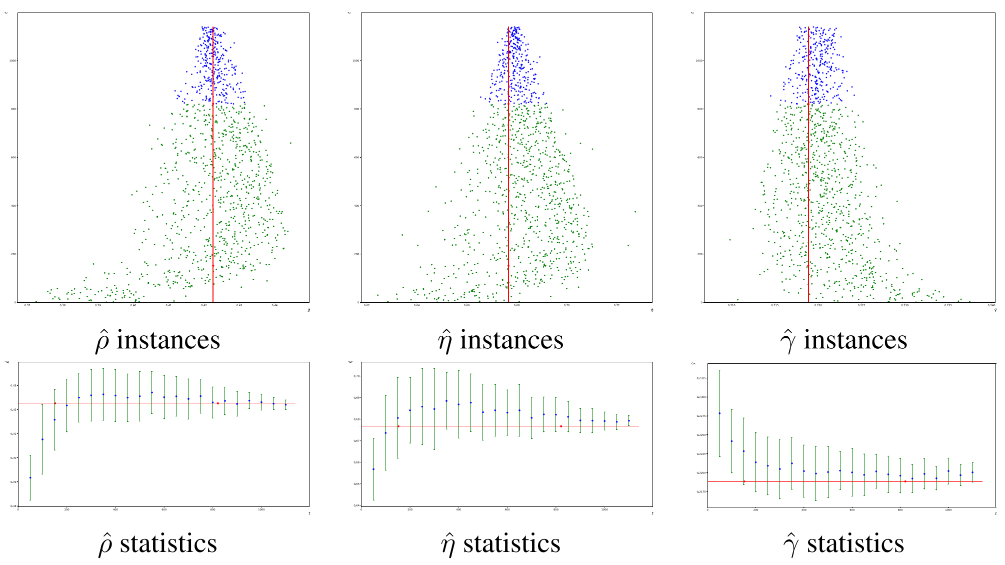

Figure 3. Empirical distributions of $\hat{\rho}$, $\hat{\eta}$, and $\hat{\gamma}$.

of data points. The second group $t \in [152, L_i - L_{i-1} = 821)$ represents the asymptotic convergence before the key point L_i. The third group $t \geq 821$ represents the data points after L_i.

Empirical distributions are given by three subfigures in the lower half of Figure 3, where the axes are switched. Each short vertical line segment on each time t represents the mean and variance of data points in interval $[t-49, t]$. The key values $\rho = 0.4226, \eta = 0.6767, \gamma = 0.2188$ are marked as the horizontal lines. Data points in Figure 3 show a convergence pattern from Theorem 1: the means gradually approach the horizontal lines and the variances show a tendency to decay.

Performance Analysis

The experimental results from the parameter settings in the section "Parameter Settings" are shown in Table 2. The performance ASL is the best on all except one MDD index. Compared with the second-ranking algorithms, ASL improves RoR rates in $[25.87\%, 153.67\%]$, SR rates in $[38.29\%, 1298.80\%]$, and MDD rates in $[26.86\%, 84.99\%]$. Compared with UBAH, ASL improves RoR rates in $[74.65\%, 551.88\%]$, SR rates in $[137.03\%, 448.87\%]$, and MDD rates in $[41.40\%, 259.00\%]$.

The running RoRs of top six algorithms are shown in Figure 4. Because of the sparseness of key points, the algorithm ASL did not take many trading actions, but it did well at the key buy and sell points. For example, the buy points near the 450th day of D_1 and the 290th day of D_3 were captured; the sell points near the 150th day of D_2 and the 520th day of D_4 were also not missed. By taking a small number of robustly actions, the profits of ASL on

Table 2. Summary of performance.

Dataset	indices	UBAH	ASL	NE	PLR-WSVM	AC	QL	AR	MIE	UP	EG	ONS	A1	CFR-OGD
D_1	RoR(%)	56.9976	**101.8371**	4.5707	77.3152	25.0207	12.6835	11.4905	20.8649	27.2996	27.3346	25.3451	27.2723	34.3359
	SR	0.5249	**1.2441**	-0.0899	0.7887	0.3356	0.0888	0.0627	0.2443	0.3986	0.3980	0.4000	0.3982	0.4044
	MDD	0.2422	**0.0826**	0.2044	0.2247	0.1336	0.3333	0.1048	0.1252	0.1262	0.1262	0.1106	0.1257	0.2205
D_2	RoR(%)	62.4111	**109.0004**	58.1839	69.5018	43.9502	35.5827	-20.4219	-17.0323	26.6795	26.6711	25.1987	26.6175	10.8004
	SR	1.0298	**2.4654**	1.4068	1.4494	1.3238	1.2079	-1.0297	-0.7935	0.8246	0.8246	0.7909	0.8236	0.2468
	MDD	0.2358	**0.0744**	0.1065	0.2106	0.1066	0.1027	0.2833	0.2063	0.1243	0.1243	0.1292	0.1244	0.1787
D_3	RoR(%)	7.6250	**30.6905**	24.3827	21.1870	2.1305	12.2027	-4.1882	-10.3159	4.2579	4.2458	4.9578	4.2688	15.0898
	SR	0.2071	**1.1368**	0.5035	0.8220	-0.0354	0.7616	-0.9938	-0.8046	0.1024	0.1016	0.1518	0.1032	0.5724
	MDD	0.1416	0.1001	0.2635	0.1054	0.1050	**0.0520**	0.0615	0.1899	0.0716	0.0709	0.0672	0.0708	0.1227
D_4	RoR(%)	25.6701	**167.3395**	16.5053	78.1235	1.1356	18.3313	16.7743	-22.0000	19.6851	19.2645	28.8577	19.5987	-28.2897
	SR	0.3405	**1.5962**	0.2502	0.6869	-0.0347	0.2760	0.2570	-0.1916	0.3368	0.3309	0.4730	0.3357	-0.2658
	MDD	0.4977	**0.1386**	0.4395	0.3407	0.2897	0.4854	0.3732	0.5849	0.2752	0.2755	0.2565	0.2750	0.4233
D_5	RoR(%)	-76.7247	**56.5970**	-75.7754	22.3116	-54.3811	-26.6218	-59.1449	8.0000	-39.0610	-38.5312	-44.9750	-38.6516	-17.6341
	SR	-0.4537	**0.7043**	-1.2240	0.4413	-0.3600	0.0836	-0.6329	0.3233	-0.4257	-0.4335	-0.1800	-0.4186	0.1237
	MDD	0.9394	**0.2922**	0.8376	0.4783	0.7837	0.7039	0.7852	0.5571	0.7220	0.7145	0.8182	0.7203	0.7780
D_6	RoR(%)	-72.6352	**65.2178**	-63.1170	-4.6729	-30.6770	-20.6053	-32.0000	-67.3513	-44.2830	-43.9820	-56.2433	-44.3953	-57.0048
	SR	-0.9797	**0.7302**	-0.9792	0.0522	-0.6303	-0.4642	-0.4049	-0.8668	-1.0580	-1.0672	-0.9369	-1.0615	-0.8592
	MDD	0.7658	**0.2441**	0.6802	0.4616	0.4051	0.3361	0.3366	0.7356	0.4848	0.4822	0.5971	0.4861	0.6333

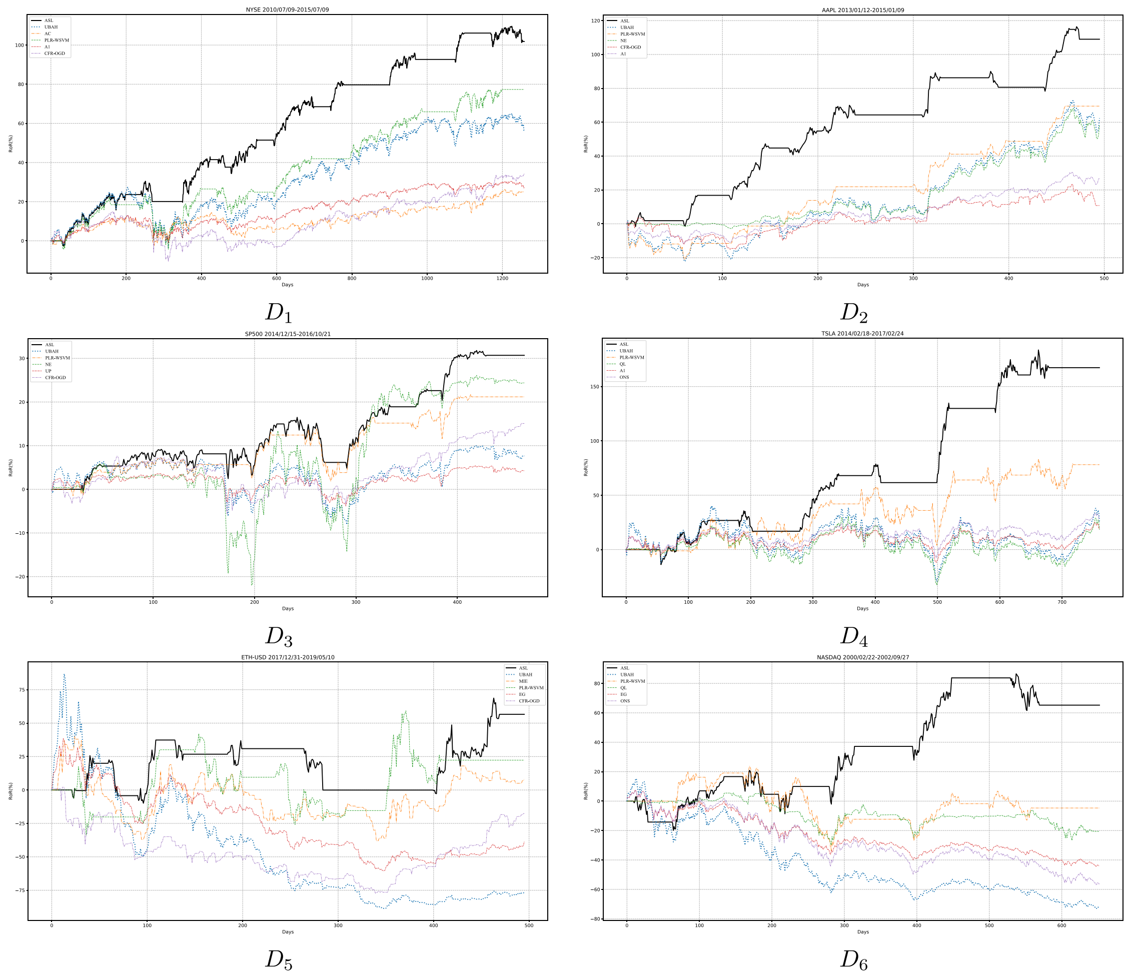

Figure 4. Dynamic RoRs of six algorithms on the six datasets.

the all datasets except D_5 show steady growth trends.

We also find that key events predicted by ASL show some robustness in dealing with the randomness of key points. For example, between day 240 and 350 of D_2, ASL did not choose the first lowest price point near the 260th day, but the second one near the 310th day. We also see this robustly behavior from day 250 to 320 of D_3 and from day 200 to 350 of D_4. The reasons for those may be that ASL estimates the lower bound of the price factor ρ and the time factor γ is also used as one of the training targets in (8).

CONCLUSION AND FUTURE WORK

In order to predict random and sparse key buy and sell points reliably, it is necessary to establish a mathematical model to predict their random occurrences, and then derive an efficient algorithm. Based on the measure theory, we propose an event mapping model and algorithm ASL to solve this problem. Compared with other 12 existing algorithms in six datasets, the results show that our ASL algorithm outperforms the other algorithms in accurate predicting random and sparse key events in trading.

The main idea of the ASL algorithm is to analyze the convergence of random sample points generated by mapping functions instead of directly estimating the probability of key events. In the future, we will further study the asymptotic statistical properties of random sample points, including the upper and lower bound probability estimation of key events, and design of new neural networks to achieve these.

ACKNOWLEDGMENT

We would like to thank the editors and reviewers for their valuable comments and constructive suggestions, and the support of the National Natural Science Foundation of China No. 61972261.

REFERENCES

1. D. Huang *et al.*, "Combination forecasting reversion strategy for online portfolio selection," *ACM Trans. Intell. Syst. Technol.*, vol. 9, no. 5, pp. 1–22, 2018.
2. B. Li, D. Sahoo, and S. C. Hoi, "Olps: A toolbox for online portfolio selection," *J. Mach. Learn. Res.*, vol. 17, no. 35, pp. 1–5, 2016.
3. L. Luo, S. You, Y. Xu, and H. Peng, "Improving the integration of piecewise linear representation and weighted support vector machine for stock trading signal prediction," *Appl. Soft Comput.*, vol. 56, pp. 199–216, 2017.
4. H. Tang, P. Dong, and Y. Shi, "A new approach of integrating piecewise linear representation and weighted support vector machine for forecasting stock turning points," *Appl. Soft Comput.*, vol. 78, pp. 685–696, 2019.
5. Y. C. Tsai *et al.*, "Assessing the profitability of timely opening range breakout on index futures markets," *IEEE Access*, vol. 7, pp. 32061–32071, 2019.
6. G. Jeong and H. Y. Kim, "Improving financial trading decisions using deep q-learning: Predicting the number of shares, action strategies, and transfer learning," *Expert Syst. Appl.*, vol. 117, pp. 125–138, 2019.
7. O. B. Sezer, A. M. Ozbayoglu, and E. Dogdu, "An artificial neural network-based stock trading system using technical analysis and big data framework," in *Proc. SouthEast Conf.*, 2017, pp. 223–226.
8. H. Li, Y. Shen, and Y. Zhu, "Stock price prediction using attention-based multi-input LSTM," in *Proc. 10th Asian Conf. Mach. Learn.*, pp. 454–469, 2018.
9. J. Nadkarni and R. F. Neves, "Combining neuroevolution and principal component analysis to trade in the financial markets," *Expert Syst. Appl.*, vol. 103, pp. 184–195, 2018.
10. F. Paiva *et al.*, "Decision-making for financial trading: A fusion approach of machine learning and portfolio selection," *Expert Syst. Appl.*, vol. 115, pp. 635–655, 2019.
11. K. Zhang *et al.*, "Stock market prediction based on generative adversarial network," *Procedia Comput. Sci.*, vol. 147, pp. 400–406, 2019.
12. N. Akbarzadeh, C. Tekin, and M. van der Schaar, "Online learning in limit order book trade execution," *IEEE Trans. Signal Process.*, vol. 66, no. 17, pp. 4626–4641, Sep. 2018.
13. W. Powell, "A unified framework for stochastic optimization," *Eur. J. Oper. Res.*, vol. 275, no. 3, pp. 795–821, 2019.
14. Y. Deng, F. Bao, Y. Kong, Z. Ren, and Q. Dai, "Deep direct reinforcement learning for financial signal representation and trading," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 28, no. 3, pp. 653–664, Mar. 2016.
15. J. Li, R. Rao, and J. Shi, "Learning to trade with deep actor critic methods," in *Proc. 11th Int. Symp. Comput. Intell. Des.*, 2018, vol. 02, pp. 66–71.
16. H. Van Hasselt, A. Guez, and D. Silver, "Deep reinforcement learning with double q-learning," in *Proc. 13th AAAI Conf. Artif. Intell.*, 2016, pp. 2094–2100.
17. A. Picasso *et al.*, "Technical analysis and sentiment embeddings for market trend prediction," *Expert Syst. Appl.*, vol. 135, pp. 60–70, 2019.
18. U. JuHyok *et al.*, "A new LSTM based reversal point prediction method using upward/downward reversal point feature sets," *Chaos, Solitons Fractals*, vol. 132, 2020, Art. no. 109559.
19. B. Moews, J. Herrmann, and G. Ibikunle, "Lagged correlation-based deep learning for directional trend change prediction in financial time series," *Expert Syst. Appl.*, vol. 120, pp. 197–206, 2019.
20. T. Leung and X. Li, "Optimal mean reversion trading with transaction costs and stop-loss exit," *Int. J. Theor. Appl. Finance*, vol. 18, no. 03, 2015, Art. no. 1550020.
21. J. Suykens and J. Vandewalle, "Least squares support vector machine classifiers," *Neural Process. Lett.*, vol. 9, no. 3, pp. 293–300, 1999.
22. N. Japkowicz and S. Stephen, "The class imbalance problem: A systematic study," *Intell. Data Anal.*, vol. 6, no. 5, pp. 429–449, 2002.
23. D. Pollard, *A User's Guide to Measure Theoretic Probability*. Cambridge, U.K.: Cambridge Univ. Press, 2002.
24. S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, 1997.
25. Z. Zhou, "A brief introduction to weakly supervised learning," *Nat. Sci. Rev.*, vol. 5, no. 1, pp. 44–53, 2018.
26. A. Rodríguez-González *et al.*, "Cast: Using neural networks to improve trading systems based on technical analysis by means of the RSI financial indicator," *Expert Syst. Appl.*, vol. 38, no. 9, pp. 11489–11500, 2011.

27. H. He and E. Garcia, “Learning from imbalanced data,” *IEEE Trans. Knowl. Data Eng.*, vol. 21, no. 9, pp. 1263–1284, Sep. 2009.
28. T. Dao *et al.*, “A kernel theory of modern data augmentation,” *Proc. Mach. Learn. Res.*, vol. 97, 2019, Art. no. 1528.

Jianfei Yin is currently an Associate Professor with the College of Computer Science and Software Engineering, Shenzhen University, Shenzhen, China. His current research interests include statistical and optimization algorithms, and machine learning. He received the Ph.D. degree in Computer Science from the South China University of Technology, Guangzhou, China, in 2005. Contact him at yjf@szu.edu.cn.

Ruili Wang is currently the Professor of Aritifical Intelligence and the Chair of Research with the School of Natural and Computational Sciences, Massey University, Auckland, New Zealand, where he is the Director of the Centre of Language and Speech Processing. His current research interests include data processing, speech and natural language processing, image and video processing, and intelligent systems. He received the Ph.D. degree in computer science from Dublin City University, Dublin, Ireland. He is the corresponding author of this article. Contact him at Ruili.wang@massey.ac.nz.

Shunda Ju is currently working toward the Master’s degree with the College of Computer Science and Software Engineering, Shenzhen University, Shenzhen, China. His current research interests include financial analysis, statistical and optimization algorithms, and machine learning. Contact him at jushunda2017@email.szu.edu.cn.

Yizhe Bai is currently working toward the Master’s degree with the College of Computer Science and Software Engineering, Shenzhen University, Shenzhen, China. His current research interests include financial trading and risk analysis, statistical and optimization algorithms, and machine learning. Contact him at baiyizhe2017@email.szu.edu.cn.

Joshua Zhexue Huang is currently a Distinguished Professor with the College of Computer Science and Software Engineering, Shenzhen University. He is also the Director of the Big Data Institute and the Deputy Director of the National Engineering Laboratory for Big Data System Computing Technology. He has published more than 200 research papers in conferences and journals. His current research interests include big data technology and applications. He received the Ph.D. degree from the Royal Institute of Technology, Stockholm, Sweden, in 1993. Contact him at zx.huang@szu.edu.cn.