Sparse Coding-Inspired Optimal Trading System for HFT Industry

Yue Deng, Youyong Kong, Feng Bao, and Qionghai Dai, *Senior Member, IEEE*

***Abstract*—The financial industry has witnessed an exceptionally fast progress of incorporating information processing techniques in designing knowledge-based automated systems for high-frequency trading (HFT). This paper proposes a sparse coding-inspired optimal trading (SCOT) system for real-time high-frequency financial signal representation and trading. Mathematically, SCOT simultaneously learns the dictionary, sparse features, and the trading strategy in a joint optimization, yielding optimal feature representations for the specific trading objective. The learning process is modeled as a bilevel optimization and solved by the online gradient descend method with fast convergence. In this dynamic context, the system is tested on the real financial market to trade the index futures in the Shanghai exchange center.**

***Index Terms*—Financial industry, financial signal processing, high-frequency trading (HFT), reinforcement learning (RL), sparse coding (SC).**

I. Introduction

IN THE era of big data, tremendous revolutions are taking place in the conventional financial industry, which have encouraged the highly productive developments in high-frequency trading (HFT). Different from typical trading behaviors, HFT participators adopt computer algorithms to accumulate the market information from exchange center at millisecond resolutions and then make trading decisions in a flash manner. Without too many concerns about the fundamental news of a traded stock, HFT strategy only interacts with the financial signal and believes that everything about the market is represented on the price fluctuations. In the contemporary world, big data in financial industry provide significant opportunities, but meanwhile, imposes great challenges to researchers in both the academic and industrial communities [1].

In the HFT scenario, acquiring a large volume of historic trading records is no longer a difficult problem. The big data opportunity thereby opens a door for designing intelligent information processing system in financial engineering [2], [3]. The financial signal is a sequence of time-course data that are incrementally released from the exchange center. To cope with such dynamic, direct reinforcement learning (DRL) system was introduced to forex and stock trading [4]–[6], which makes policies and adjusts state (trading positions) according to the current market condition [Fig. 1(a)]. Such a system exploits the merits of self-taught learning to incrementally update the policy-making part (i.e., a set of parameter) for rewards maximization.

Typical DRL directly utilizes price sequence sampled every day or week to summarize the market conditions. Such an implementation is reasonable in the low-frequency setting because long-term price sequences contain little noise and always imply certain trend of the market [7]. In HFT, the market information is updated almost every second and potentially contains high amount of noises and unpredictable random effects. Therefore, directly exploiting the price sequence to summarize the market condition is not plausible. To overcome this drawback, we introduce a more reasonable way to summarize the noisy high-frequency market conditions as some high-level interpreted features [8] with sparse coding (SC).

SC is an ideal feature representation approach for HFT due to its significant advantages in noise reduction and task-driven dictionary learning. Typical SC encodes the raw data as sparse representations with an over-complete dictionary [9], [10]. It shows great promises in dealing with noises [11], [12] and has been applied to many signal recovery problems including image/video completion [13], [14], sensor network [15], three-dimensional (3-D) reconstruction [16], and transient light decomposition [17]. Moreover, one inspiring work [18] indicates that SC could also be trained in a task-driven manner [19]–[21] to incorporate feature learning concepts into dictionary.

As a primary technical contribution, this paper proposes a sparse coding-inspired optimal trading (SCOT) system to learn the feature representation and DRL trading in a joint optimization. In a nutshell, SCOT learns a set of bases as well as the trading parameters simultaneously within the supervision of specific trading goals, e.g., maximizing the profits [Fig. 1(b)]. Such a closed-loop training strategy is elucidated from the aspect of task-driven learning [18], [19] and encourages much reasonable dictionary for the trading objective. To the best of our knowledge, it is the first time to extend the power of SC into the new field of financial signal processing and trading.

While the concept of the system is elegant, it does not lead to an equivalently easily solved objective. In fact, SCOT relies on a recurrent bilevel optimization [22], which includes another optimization problem in the constraint. We tried to solve this complicated and recurrent optimization by online gradient methods to iteratively update the parameters with the

Manuscript received October 16, 2014; revised January 03, 2015; accepted February 07, 2015. Date of publication February 16, 2015; date of current version March 27, 2015. Paper no. TII-14-1141.

Y. Deng is with the School of Electronic Science and Engineering, Nanjing University, 210093 Nanjing, China (e-mail: yuedeng.thu@gmail.com).

Y. Kong is with the School of Computer Science and Engineering, Southeast University, 210000 Nanjing, China.

F. Bao and Q. Dai are with the Automation Department, Tsinghua University, 100084 Beijing, China.

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TII.2015.2404299

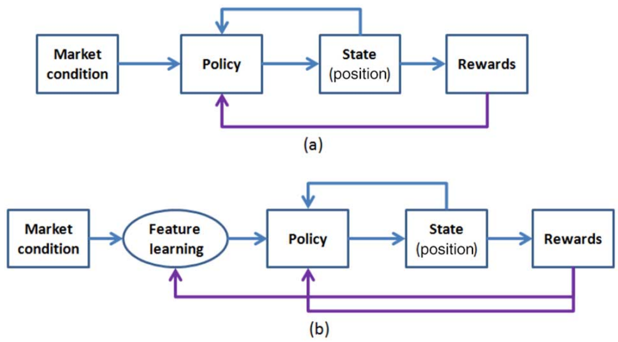

Fig. 1. Schematic comparison on conceptual differences between (a) DRL [4] and (b) SCOT. The blue arrows indicate the information flow and the purple arrows indicate the learning processes (parameter updating). In SCOT, both the trading policy and feature representations (in ellipse) are simultaneously learned to maximize the trading rewards, while DRL does not exhibit feature learning mechanism.

latest price information. The effectiveness of the SCOT is evaluated on the high-frequency data of China Stock Index Future (IF) (also known as Shanghai IF). With different trading conditions, SCOT consistently outperforms other trading methods and is especially appropriate for the high-frequency data with large intraday movements.

II. Preliminary

A. Basic Concepts in HFT

Some basic HFT concepts are briefly introduced here for the readers who are not familiar with trading. The decision making problem in algorithmic trading is generally concluded as selecting the appropriate time to open and close the positions on the financial market. When a long (short, respectively) position is established, the trader can thus make profits if the consequent price goes up (down, respectively) and vice versa. For decision-making, traders or algorithms mainly interact with the price and order book information released from the exchange. Price information mainly summarizes the current transaction price and the corresponding volume (quantity of shares). The order book lists the number of shares being bid or offered at each price point, or market depth. Order book information is crucial for identifying the market participants behind the buy and sell orders.

The trading charge is composed of two parts: 1) the transaction costs; and 2) the slippage costs. Transaction costs are the money charged by the exchange center when traders buy and sell stocks. Slippage costs are not important in the conventional sense, but are quite critical in the high-frequency environment. It is mainly caused by the bid-ask spread and the costs by some unpredicted reasons, e.g., communication delay. The trader is always required to give a higher (lower, respectively) price to buy (sell, respectively) the stock than the market price (p_t) due to the slippage.

B. Reinforcement Learning

Designing trading models is challenging due to the lack of supervision information [23]. Despite conventional supervised learning strategies fail, reinforcement learning (RL) well addresses the problem by introducing the concept of self-enhancement [24]. RL assumes that when more and more knowledge is acquired in an unknown environment, the algorithm can iteratively train an agent to take corresponding actions so as to maximize some notion of cumulative reward. According to a recent survey [25], RL algorithms are mainly categorized into three representative types: 1) critic-only; 2) actor-only; and 3) actor-critic methods. We will, respectively, analyze these three different kinds of RL and discuss whether they are suitable for HFT.

Critic-only methods are the most typical algorithms in the RL family. In these methods, the policy of an agent is mainly determined by some discrete states which can be derived to minimize some value function across the whole environment. Such value function-based methods, e.g., TD-learning and Q-learning [26], work well on some scheduler problems where immediate feedback on performance is not readily available [4]. Mathematically, the definition of value function always involves a term recoding future discounted returns. However, rather than evaluating the returns during the whole trading period, HFT just concerns about immediate return. The nature of HFT is to make profits from very short-term price fluctuations and does not care about long-term rewards. Meanwhile, trading decisions are made in an online manner and not any kind of future market information is allowed in the model. This is because once the agent has sensed the environment/market information after time t, there is no opportunity to come back to t and establish a corresponding position. As indicated in [4], value function-based methods are good choices for some offline scheduler problems [26], but are not suitable to the dynamic trading problem for online decision making.

Actor-only method directly learns a spectrum of continuous actions which that are derived from a parameterized family of policies. Such RL methods behave like modern machine learning approaches to optimize an objective function with latent parameters. As indicated in [25], the only requirement in actor-only method is a differentiable objective function, and it does not concern anything about future discounting returns. Different from typical critic-only method that always involves dynamic programming for learning, the optimization in actor-only system is flexibly implemented via gradient descent. Besides, rather than describing responses to diverse market conditions with some discrete states (in Q-learning), actor-only method learns the policy in the continuous space that exhibits much generalized scalability in addressing the complex environment. In conclusion, the actor-only RL exhibits the advantages: 1) flexible objective function (flexibility in both design and optimization); and 2) continuous descriptions of market condition. Therefore, it is plausible for HFT.

Actor-critic methods combine the two aforementioned RL frameworks altogether in which actor computes a parameterized policy and critic evaluates such action with the purpose to maximize the ultimate rewards. For long-term trading, the training efficiency is not an important issue. However, for HFT, decision and parameter learning are updated in almost real-time. Thereby, the actor-critic method is not preferred here due to its complicated two-layer configurations that potentially requires heavy computational burden.

In conclusion, a good RL system working for HFT should have three desired properties: 1) immediate estimates of incremental performance (rewards); 2) robust to diverse market conditions; and 3) efficiently optimized. It is apparent that actor-only RL paradigm addresses these three goals making it an ideal framework for HFT. In fact, it has already been applied to conventional long-term algorithmic trading [4], [5], [27]. In these previous works, actor-only RL is termed as DRL. To make our expression consistent with previous works, we will call actor-only RL as DRL in the following parts. The advantages of DRL over other RL approaches are also analyzed in the work [4].

C. Direct Reinforcement Trading System

In trading, the sequence of prices $p_1, p_2, \ldots, p_t, \ldots$ is released from exchange center and p_t is the price information at time t. Consequently, f_t is defined as the feature vector extracted at time t. For instance, in [4], the price changes in a window are directly used as the feature vector. When defining $z_t = p_t - p_{t-1}$ as the price change, the feature vector is formulated as $f_t = [z_{t-wid+1}, \ldots, z_t] \in \mathbb{R}^{\mathrm{wid}}$, where wid is the length of a window. Further, $\delta(t) \in \{\mathrm{long}, \mathrm{neutral}, \mathrm{short}\} = \{1, 0, -1\}$ is the trading decision made at time point t whose value is related to the current market condition (summarized into feature f_t) and the holding positions δ_{t-1}

$$\delta_t(f_t) = g[\underbrace{\mathbf{w}f_t + u\delta_{t-1} + b}_{Q}] \tag{1}$$

where $\mathbf{w}$ and u are the regressors for the feature and holding position and b is the bias term. $g(\cdot)$ gets the sign of the variable Q. The hard sign function is not differentiable and, therefore, the soft sigmod function is used in $g(\cdot)$ [4]. Till now, it is obvious that (1) is essentially an one-layer recurrent neural network with parameter set $\Theta = \{\mathbf{w}, u, b\}$.

With the symbols defined above, the profit R_t made by the trading model at time t is obtained, i.e.,

$$R_t = \delta_{t-1} z_t - C(\delta_t, \delta_{t-1}). \tag{2}$$

In RHS of (2), the first term is the profit/loss made from the market fluctuations, and the second term is the trading cost for changing positions at time t. In our model, if $\delta_t = \delta_{t-1}$, nothing needs to be changed. However, when δ_t and δ_{t-1} indicates opposite positions, the positions should be changed accordingly and the trading costs apply. The value of RL is capturing the delayed effect of an action through a Markovian decision process (MDP). It is not immediately clear whether buying a particular stock at time t was a good decision. Only after time t, it becomes clear that the decision was good (bad) if the price went up (down). Therefore, essentially, the model parameters are only refined by all the market information till $t - 1$ and the learned parameters are exploited to trade on time t.

By considering the practical HFT scenario, the trading positions are not expected to reverse frequently to avoid the large amount of trading costs. Therefore, in this paper, the quadratic term is exploited to penalize the positions changes with the transaction costs tc, i.e., $C(\delta_t, \delta_{t-1}) = tc(\delta_t - \delta_{t-1})^2$. As a result, the optimal trading model is defined with the learning parameter set $\Theta = \{\mathbf{w}, u, b\}$

$$\max_{\Theta} U_T\{R_1, \ldots, R_T | \Theta\} \tag{3}$$

where $U_T\{\cdot\}$ defines the reward function in the period of $1, \ldots, T$. The most straight-forward reward is the average profit made in the T period.

III. SCOT

A. Motivations

Although DRL introduced in the last section is a good trading paradigm, another important issue, i.e., feature representation, is not explicitly considered in the DRL framework [4]. In machine learning, data representation and decision making are equivalently important to the final performance [28]. In the context of HFT, tick data[1] contain unpredictable uncertainties and hence appeal for intelligent data summarization approaches. It is natural to ask is it possible to incorporate feature learning into DRL and, if so, which feature representation method is ideal for HFT.

This work incorporates the prevalent SC into the DRL for robust data representation and feature learning. SC exhibits two irreplaceable properties: 1) robustness; and 2) interpretability, making it a natural choice for HFT. For robustness property, SC is effective in removing conventional Gaussian noises and also robust to large outliers [16]. As the tick sequence may suffer significant jump and reverse in very short period, HFT tick series are hence nonstationary and contain great disturbances. SC is a desired approach in removing diverse types of noise structures from the HFT data. From the view of interpretability, the sparse activation phenomena have been observed in the neuron [29], making the statistic model well explained by the information processing mechanism in the brain [30]. Practically, SC gains outstanding performances in a number of classification-related tasks. Although incorporated in an RL framework, trading is essentially a classification problem and SC is powerful on this kind of tasks.

One intuitive way to combine SC with DRL may follow a sequential implementation. In such a manner, unsupervised SC is exploited to encode the raw feature as some high-level representations. Then, these sparse representations are fed into the DRL to train the trading model $(\mathrm{SC} + \mathrm{DRL})$. While sequential implementation is straightforward, it does not shed light on the task-driven feature learning aspect. In machine learning, better feature learning ways are always performed in a closed-loop to extract task-dependent features. Therefore, in this paper, an SCOT framework will be introduced to learn the feature representation and trading rules in a joint framework. It encourages the task-driven dictionary set to encode the data into much interpreted high-level representations for trading. SCOT does not rely on any kind of manually provided labels and could iteratively enhance its own performances on the sequential time-course data.

[1]The high-frequency data are also named as ticks, i.e., one tick per update.

B. SCOT

Before giving the formulations of SCOT, the notation and symbols are summarized first. Let's suppose f_t be the raw feature extracted from the price time sequence at time point t. α_t is the high-level representation of f_t after SC with an overcomplete dictionary $\mathbf{D}$. The trading direction δ_t is determined by the joint interactions of the parameter set $\Theta = \{\mathbf{w}, u, b\}$ and the high-level representation α_t as in (1), making the reward $R_t(\alpha_t)$. Accordingly, the SCOT model is defined as

$$\begin{aligned} &\min_{(\mathbf{D},\Theta)} -U_T(R_1(\alpha_1), \ldots, R_T(\alpha_T)|\Theta) \\ &\text{s.t.} \quad \alpha_t = \arg\min_{\alpha} |||f_t - \mathbf{D}\alpha||_2^2 + \lambda||\alpha||_1, \quad t = 1, \ldots, T \\ &\qquad\qquad ||\mathbf{d}_j|||_2 < 1 \quad \forall j. \end{aligned} \tag{4}$$

In the above formulation, U_T is the trading objective and its simplest form, i.e., the average profit in the T period is adopted in this work

$$U_T = \frac{1}{T}\sum_{t=1}^{T} R_t(\alpha_t). \tag{5}$$

Equation (4) is a bilevel optimization [22], [31] because the constraint itself contains another optimization problem. Its learning procedures will be given in Section III-C.

The mechanism about how (4) addresses the HFT problem is apparent. In (4), two set of parameters should be learned simultaneously: 1) the dictionary $\mathbf{D}$ (related to feature representation); and 2) trading parameter Θ (related to trading decision). The dictionary should not only satisfy the signal reconstruction objective in the constraint and meanwhile its coded feature α_t should also be informative for trading decision. Learning such a set of dictionary is always termed as task-driven dictionary learning [18]. From the objective function U_T, SCOT does not require any kind of human supervision, i.e., requiring no label information. Moreover, it is worthwhile to emphasize here that the calculation of $R_t(\alpha_t)$ only requires all the market information before time t. Not any kind of future information of the market is involved, which is desired in the online trading scenario. It is noted that SC itself induces a two-variables optimization: 1) one for the dictionary; and 2) the other for sparse codes α. In the optimization of SCOT, one should consequently alternate among three variables, the two in SC (dictionary $\mathbf{D}$ and α) and the Θ induced by the RL.

C. Optimization

The objective in (4) is highly nonlinear and nonconvex. A good way to cope with it may follow the alternative optimization strategy by iteratively solving α, Θ, and $\mathbf{D}$ while keeping others fixed. First, when the dictionary $\mathbf{D}$ is fixed, the sparse representations α are correspondingly obtained by the first constraint in (4). This part of optimization is the same as the procedures in [30]. When α and $\mathbf{D}$ are fixed, the rules for Θ updating are similar to the implementations in [4]. It is simply understood as seeking for the gradient of the objective with respect to Θ. For simplicity, $R_t(\alpha_t)$ is shortened as R_t and the gradient for Θ is obtained by

$$\frac{\partial U_T}{\partial \Theta} = \sum_{t=1}^{T} \frac{\partial U_T}{\partial R_t}\left[\frac{\partial R_t}{\partial \delta_t}\frac{\partial \delta_t}{\partial \Theta} + \frac{\partial R_t}{\partial \delta_{t-1}}\frac{\partial \delta_{t-1}}{\partial \Theta}\right]. \tag{6}$$

From (6), it is worth noting that the calculation of δ_t in (1) relies on a recurrent neural network implying that the exact calculation of $\frac{\partial \delta_t}{\partial \Theta}$ depends on the entire sequence before t. To simplify it, we follow the implementations in [4] by exploiting back propagation through time (BPTT) [32], i.e.,

$$\frac{\partial \delta_t}{\partial \Theta} = \nabla_\Theta \delta_t + \frac{\partial \delta_t}{\partial \delta_{t-1}}\frac{\partial \delta_{t-1}}{\partial \Theta}. \tag{7}$$

In the above formulation, $\nabla_\Theta \delta_t$ is directly obtained by applying the gradient rule on (1) with respect to Θ. In practice, gradient calculations are implemented sequentially with the evolutions of t. When calculating $\frac{\partial \delta_t}{\partial \Theta}$, the gradient of $\frac{\partial \delta_{t-1}}{\partial \Theta}$ has already been obtained in the last time point. The learning process for Θ generally falls into the category of *policy gradient method* for actor-only RL system [25].

When α and Θ are fixed, the gradient for $\mathbf{D}$ is much more difficult since it is involved in a low-level optimization implicitly embedded in the constraint. Even worse, this low-level optimization itself contains an undifferentiate ℓ_1 norm. Inspired by the policy gradient method for actor-only RL, we derive its gradient as follows:

$$\frac{\partial U_T}{\partial \mathbf{D}} = \sum_{t=1}^{T} \frac{\partial U_T}{\partial R_t}\left[\frac{\partial R_t}{\partial \delta_t}\frac{\partial \delta_t}{\partial \mathbf{D}} + \frac{\partial R_t}{\partial \delta_{t-1}}\frac{\partial \delta_{t-1}}{\partial \mathbf{D}}\right]. \tag{8}$$

Consequently, we calculate

$$\frac{\partial \delta_t}{\partial \mathbf{D}} = \nabla_{\mathbf{D}} \delta_t + \frac{\partial \delta_t}{\partial \delta_{t-1}}\frac{\partial \delta_{t-1}}{\partial \mathbf{D}}. \tag{9}$$

Similar to (7), $\frac{\partial \delta_{t-1}}{\partial \mathbf{D}}$ has already been obtained at the last time point $t-1$. The difficulty remains in the derivatives of the first term in RHS of (9). Due to the ℓ_1 norm, the derivatives of δ_t with respect to the dictionary $\mathbf{D}$ is not explicitly obtained. Fortunately, in a previous work [18], this implicit differentiation has already been analytically defined for general task-driven SC problem. The theoretical justifications are not trivial and interested readers are referred to the appendix part of [18] for details. In this paper, we just directly provide this gradient based on the existing theoretical result, i.e.,

$$\nabla_{\mathbf{D}} \delta_t = -\mathbf{D}\beta_t\alpha_t + (f_t - \mathbf{D}\alpha_t)\beta_t^T \tag{10}$$

in which $\beta_t \in \mathbb{R}^l$ is composed of two parts: 1) $\beta_{\Lambda^c} = 0$; and 2) $\beta_\Lambda = (\mathbf{D}^T\mathbf{D})^{-1}\nabla_{\alpha_\Lambda}\delta_t$. Λ is active set defining the indices of the nonzero coefficients in α. Till now, all the required gradient information for SCOT learning have been stated.

Due to the recurrent nature of the trading model in (1), the whole SCOT is trained in an online manner as in Algorithm 1. With the bilevel optimization strategy, the convergence is only expected to be a local minimum, which turned out to be sufficient and robust in practical trading. In line 8, after updating the dictionary along the gradient direction, the gradient projection procedure is performed to make the learned dictionary satisfy the second constraint in (4). Due to the recurrent nature of trading function, the SCOT could only be trained sequentiality. To avoid bias in training, the training results are further refined by repeating the online process for multiple times. That is why the outer iterations (controlled by c) are added. In each outer iteration, the learning rate is decreased by $1/c$ in line 3. The algorithm is regarded as converged when $\frac{\|\mathbf{D}_{c+1}-\mathbf{D}_c\|_2}{\|\mathbf{D}_c\|_2} < 1e^{-3}$.

Algorithm 1. Online Gradient Descent Algorithm for Optimal Trading Inspired Sparse Coding Learning

Input: Raw price ticks $p_1 \dots p_T$ received in an online manner; $\mathbf{D} \in \mathbb{R}^{k \times l}$ (initial dictionary), N (the number of outer iterations); ρ, c_0 (learning rate).

1 **repeat**
2 **for** $c = 1$ to C **do**
3 Update learning rate $\rho_c = \min(\rho, \rho\frac{c_0}{c})$ for this outer iteration;
4 **for** $t = 1$ to T **do**
5 Generate Raw feature $\mathbf{f}_t$ vector from price ticks;
6 Sparse Coding: get α_t from the first constraint in (4);
7 Trading Strategy Parameter Updating: $\Theta_t = \Theta_{t-1} - \rho_c \frac{\partial U_t}{\partial \Theta}$;
8 Dictionary Updating by a projected gradient procedure:

$$\mathbf{D}_t = \Pi_{\mathcal{D}} \left\{ \mathbf{D} - \rho_c \frac{\partial U_t}{\partial \Theta} \right\},$$

where $\Pi_{\mathcal{D}}$ is the orthogonal projections on the learned dictionary $\mathbf{D}$;
9 **end**
10 **end**
11 **until** *convergence*;

Output: $\mathbf{D}$(dictionary), Θ (trading parameter).

IV. Trading System

In this section, some practical issues about the SCOT system will be discussed covering both the aspects of feature extraction and system configuration.

A. Raw Price Features

Raw feature f_t is the first-step summarization of the current market condition. In Section II-C, we have indicated that existing work used the price changes in the last wid period to construct the raw feature, i.e., $f_t = [z_{t-wid+1}, \dots, z_t]$, where $z_t = p_t - p_{t-1}$. This kind of raw feature works well for long-term trading but may not be that ideal for HFT. It is because that in high-frequency environment, the price change between two ticks may be quite small and cannot fully represent the emotion of the market. Moreover, each updated tick does not only

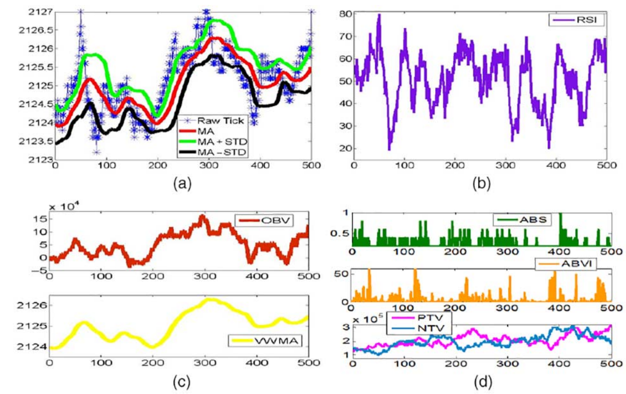

Fig. 2. High-frequency data [stars in (a)] and the visualization of some extracted features from the perspectives of (a) trend, (b) volatility, (c) volume, and (d) order book.

contain the price information, but also takes the volume information and the order book information. Just exploiting the price information to generate the feature may not be sufficient in quantifying the market dynamics. We prefer to extract features on the high-frequency data from multiple aspects.

To better illustrate the historic data, we extract the feature from five aspects: 1) trend indicator; 2) oscillator indicator; 3) price change; 4) volume pattern; and 5) order book feature. Moving average (MA) and standard deviation (STD) of prices are calculated as the trend indicator to measure the momentum of the movement. Relative strength indicator (RSI) is derived to represent the oscillator indicator, which summarizes the volatility of the price fluctuation. The price change feature is calculated by normalizing the high-frequency price returns in a short period as in [3]. For the volume feature, on balance volume (OBV) and volume weighted moving average (VWMA) are adopted as suggested by some existing works. The above technical indicators are all benchmark things in the algorithmic trading field [33]. Interested readers are referred to the previous works [6], [34] for details.

In the high-frequency data, the order book information recording the current bid and ask prices are also accumulated. In a much common sense, the order book is informative for the short-term price movement because it potentially reveals the gambling between the buyer and the seller. Following the idea in [35], a group of features are extracted including the ask-bid spread (ABS), ask-bid volume imbalance (ABVI), and signed transaction volume (SGV). SGV is the difference between positive transaction volume (PTV) and negative transaction volume (NTV). The visualizations of the extracted features from different perspectives are provided in Fig. 2.

After accumulating the features for one tick, a raw feature vector of more than 80 dimensions are obtained. To speed up the SCOT learning, a feature with small dimensions are highly desired. Therefore, the raw feature is projected into a lower dimensional subspace of $\mathbb{R}^{20}$ using principal component analysis (PCA). Then, the 20-dimensional PCA projections are utilized as f_t for SC. To note, f_t is only the raw feature serving as the input of the SCOT system and the trading parts do not explicitly interacts with f_t to make trading decisions.

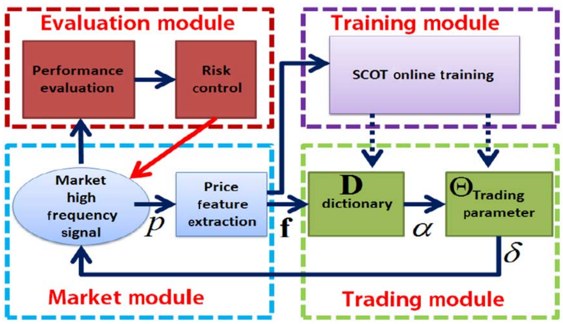

Fig. 3. Overview of the SCOT trading system with the signal flows between different modules.

B. SCOT Trading System

An overview of SCOT trading system is shown in Fig. 3 which is composed of four parts: 1) market; 2) training; 3) trading; and 4) evaluation modules. The arrows indicate the signal flows among different modules. In a nutshell, the market module is the direct interface with the market which updates high-frequency information for the system. More importantly, market module also plays an important role in operating trading strategies. It receives the trading signal from the trading module and then places bid/ask orders to the market.

From the market module, the market information is summarized as a sequence of raw features f extracted by the methods in Section IV-A. Then, the raw feature sequences are simultaneously sent to the training module and the trading module. In the SCOT training module, the dictionary $\mathbf{D}$ and trading parameter Θ are iteratively updated in an online manner according to Algorithm 1. Then, the newly updated dictionary and parameters reflecting the latest market conditions are conveyed into the trading modules. In trading model, the real-time price features are coded as some sparse representations for decision-making. To note, the training module and trading module only temporarily communicate with each other to update the parameter. In this paper, the parameters in the trading module are updated for about every 30 min (3000 ticks). Therefore, the dashed arrow is used between the training and trading modules.

In addition to the aforementioned three modules, the performance/risk evaluation module should also be added. In this module, the risk and profit are continuously evaluated to avoid large losses. The stop loss function is implemented immediately to shut down the opening position if the market goes against the expected direction and makes a large loss in the portfolio. The stop loss mechanism is indicated by the red arrow in Fig. 3.

V. Experimental Verification

A. Intraday Evaluations

In the experiment, the SCOT was tested on the high-frequency data of Shanghai IF which is the most liquid financial derivative in China. The information of the IF is updated twice per second leading to an accumulation of more than 32 000 ticks every day. Each tick contains the information of market price, trading volume, bid/ask prices, and their depths. The data are captured by our own trading system in real-time and the historic data have been maintained in the database. All the profits and losses (P&L) by trading IF are calculated in points. In practice, one point represents the value of 300 CNY (approximate 50 USD) and the transaction cost is about 27 CNY (about 0.1 point). The SCOT system is only implemented for intraday trading that any reaming positions will be mandatorily shut down at close time to avoid overnight risks.

In the tests, SCOT will be compared with other related trading systems. The competitors selected in this work are DRL [4] model and SC due to their natural connections to SCOT. For the fairness of comparisons, the multiple features used in SCOT are also applied to DRL and SC. The differences of the three systems are generally summarized here. DRL directly uses the raw feature f_t as the input to learn a trading rule. In typical SC, f_t is further coded as the sparse representation α_t by the prevalent efficient SC [30]. Then, α_t is used to train a DRL trading strategy (SC+DRL). In SCOT, training is implemented in a closed-loop manner to encourage trading strategy and dictionary mutually supervise each other.

One major difficulty in HFT is caused by the unpredictable heterogeneity of the intraday price patterns. The most representative intraday patterns are tested in the experiment containing up trend, down trend, and swing pattern. The meaning of up/down trend is clear from its definition and swing pattern implies the market showing no obvious mood in going to one direction. In finance, the swing pattern is always described by a mean-reversion process. The testing days are all selected from the market data in September 2014 and the trading results produced by different systems on these respective days are reported in Fig. 4. In the P&L results of Fig. 4, the curve is flat in periods when no trading position is held by the system.

As shown in the results, the systems perform differently under different market conditions. When there is an obvious trend/direction, all the three systems produce reliable profits, but the failure case happens on the trading day with swing pattern. This is conceivable because the trading rule incorporated in all the three systems is inspired by the DRL strategy which is a trend following method as indicated in [6] and [34]. Finally, we will analyze the successful and failure cases, respectively.

Within the days exhibiting significant price movements, SCOT makes the highest profits among the three systems no matter the direction is up or down. Taking the results in Fig. 4(a) as instances, it is apparent that SCOT keeps more neutral periods than other systems while having a fast response to the major price lift (around 15 000 ticks). With the swing intraday pattern, none of the system makes positive profits, while SCOT suffers the minimal losses among the three systems. When market is suffering the mean-reversion process, too many fake signals are made by the trading models. However, even if in the swing day, there may also exist small intraday trend (around 18 000 ticks). SCOT discovers this trend in an early time and protects the profit when market suffering small volatility.

B. Consecutive Evaluations

The performances of different trading systems are further investigated on the consecutive trading days. All the trading days from September 30, 2013 to September 30, 2014 are

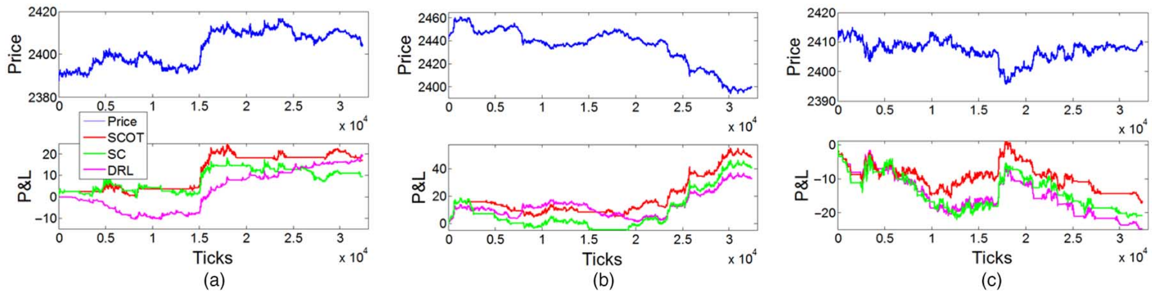

Fig. 4. Comparisons of different trading systems on three representative intraday patterns. In each figure, red, green, and purple colors correspond to P&L of SCOT, SC, and DRL, respectively. (a) Up trend (0916). (b) Down trend. (c) Swing pattern (0917).

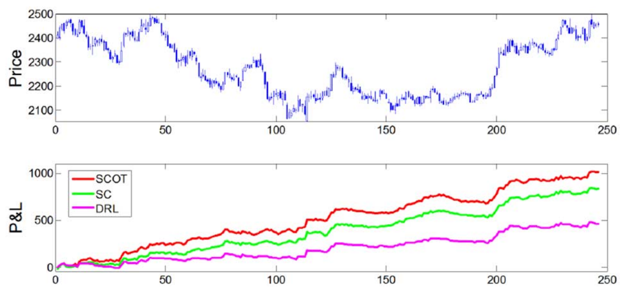

Fig. 5. Portfolio value changes over the 246 consecutive trading days from September 30, 2013 to September 30, 2014. The upper panel provides the candle bars of the index future and the lower panel shows the P&L of different trading systems.

considered in this part of experiments. The three trading systems start analyzing the ticks when the market open and will close any remaining positions at the close time. As discussed in the previous section, systems may suffer the profit losses during the days with swing patterns. To protect the portfolio values, a stopping loss strategy is applied to all the trading systems. If any system has made the losses for more than 15 points on some particular day, the system will be mandatorily shut down. The P&L curves of different systems for this consecutive test are provided in Fig. 5.

From the figure, SCOT produced the highest amount of profits on the consecutive evaluations, which agrees with the previous results on intraday test. By analyzing the figure in details, the systems always make profits during the days with significant price movement and suffer loss when the daily price range is narrow. Compared with others, SCOT is much robust to the small fluctuations and keeps its profit on the bad days. Fortunately, it can actively gain the profits when the market is advantageous to it.

The quantitatively evaluations of different trading systems (TSs) on the consecutive trading are summarized in Table I, where the total profits, Sharpe ratio (SR), and profitable days (PDs) are, respectively, reported. SR is a benchmark indicator used to assay the performance of a trading method by adjusting the returns with their risks, i.e., $\mathrm{SR} = \frac{\mathrm{mean}(R_t)}{\mathrm{std}(R_t)}$, where R_t is the return of the tth trading day. In addition to the DRL-related systems, two other famous trading systems utilized by other brokerage firms are also tested here: 1) MA crossover (MC) [7]; and 2) range break (RB). In general, MC sends a trading signal when the short-period MA cross the long-period MA. RB

TABLE I
QUANTITATIVE EVALUATIONS OF THE TRADING SYSTEMS

TS	SCOT	SC	DRL	MC	RB	BH
Profits	1 008	831	459	213	377	57
SR	0.29	0.25	0.17	0.07	0.12	0.01
PD	53%	49%	44%	37%	40%	–

calculates the upper and lower bounds of each trading day using the last day's information. A corresponding long/short position is opened only if current price breaks the range [36]. The buying and holding (BH) performance is also reported in the Table I. As BH continuously holds the position throughout the whole trading period, the profitable rate (PD) for it is not calculated in table. Although it is happy to see the points earned in Table I, we still remind the readers that all these results are obtained in an ideal laboratory setting with back testing. In the real trading conditions, many more complicated factors, e.g., communication delay and portfolio management, should be considered. Therefore, the P&L in both Table I and Fig. 5 are only possibly regarded as ideal references.

In the table, SCOT has the best portfolio performance. SCOT produces the highest amount of rewards during this one-year period test. From the PD comparison, it is interesting to note that the trading systems cannot make money everyday. In totally, 246 trading days, the rate of PDs is below 50% for others and is only slightly above fifty percents for SCOT. Such a low profitable rate is reasonable due to the nature of the trend following systems. In the stock market, the trend patterns only occupy a small portion of the whole market fluctuations. However, the major price movement is determined by these limited trending days, which is obvious from the candle cartoon in the upper panel of Fig. 5. The trend following systems are not established on the purpose for the highest winning rate. Instead, they just seek for the profits from the major trend and avoid losing money when the market's mood goes against their mechanism. From both the total profits and PD, SCOT is the best among the three in tracking trend and protecting profits during bad periods.

C. Offline Initializations Strategies

In optimization, a well-known fact is that online training needs less knowledge of the system but more efforts in convergence, whereas offline training saves time but needs more

TABLE II
OFFLINE INITIALIZATION STRATEGIES

Dictionary	SC	LS	Random	Voting
Profits	1 008	932	665	871
SR	0.29	0.27	0.21	0.33
PD	53%	52%	45%	55%

knowledge. For the HFT problem, during the trading process, at the time point t, the system cannot access any future information after t. Therefore, it is hard to directly apply offline training strategy to train SCOT due to the lack of full knowledge. To facilitate the online training process, we adopt a training strategy called offline initialization plus online training. In practice, an initial dictionary $\mathbf{D}_{\text{ini}}$ is trained in an offline manner from the previous trading records. Then, in the current trading day, $\mathbf{D}_{\text{ini}}$ is used to initialize the whole SCOT for better performances.

Here, we discuss three different offline initialization strategies for SCOT: 1) SC [30]; 2) least-square (LS); and 3) random initialization in Table II. SC (LS, respectively) learns the dictionary by imposing the ℓ_1 (ℓ_2, respectively) norm to penalize the feature vector. Different initialization strategies are applied to the same data in Fig. 5. From the results, it is interesting to note that SC generally outperforms LS and random methods. It is reasonable because SCOT is not convex and different initializations lead to different local minimum. Further, a majority voting strategy is also exploited to combine the results of different initialization strategies. The trading signal is only believed to be true if it is voted by at least two SCOT systems (initialized by different methods). This voting strategy decreases the total profits because it sets more strict rules to send the trading signal. However, both the SR and PDs are increased. Therefore, the voting strategy is recommended for the conservative traders, while SC initialization is recommended for aggressive traders.

VI. CONCLUSION

This paper presents SCOT, an SC inspired trading system for financial industry. SCOT takes both the aspects of feature learning and optimal trading into consideration whose learning process is subject to a bilevel optimization. The success of SCOT is ascribed to the following facts. First, it well addresses the important issue of noise reduction in the financial data with the prevalent SC method. Second, the learning process is embedded into a task-driven paradigm for optimal feature learning in an online manner, making it favorable in capturing the dynamic market conditions. The dictionary learned from SCOT is very robust in coping with the intraday price heterogeneity. From the experimental verifications, SCOT makes reliable profits on a sequence of consecutive trading days by balancing the trading risks and opportunities.

REFERENCES

[1] T. Hendershott, C. M. Jones, and A. J. Menkveld, "Does algorithmic trading improve liquidity?" *J. Finance*, vol. 66, no. 1, pp. 1–33, 2011.

[2] V. Behbood, J. Lu, and G. Zhang, "Fuzzy refinement domain adaptation for long term prediction in banking ecosystem," *IEEE Trans. Ind. Informat.*, vol. 10, no. 2, pp. 1637–1646, May 2014.

[3] Y. Deng, Q. Dai, R. Liu, Z. Zhang, and S. Hu, "Low-rank structure learning via nonconvex heuristic recovery," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 24, no. 3, pp. 383–396, Jan. 2013.

[4] J. Moody and M. Saffell, "Learning to trade via direct reinforcement," *IEEE Trans. Neural Netw.*, vol. 12, no. 4, pp. 875–889, Jul. 2001.

[5] J. Moody, L. Wu, Y. Liao, and M. Saffell, "Performance functions and reinforcement learning for trading systems and portfolios," *J. Forecast.*, vol. 17, no. 56, pp. 441–470, 1998.

[6] M. A. Dempster and V. Leemans, "An automated fx trading system using adaptive reinforcement learning," *Expert Syst. Appl.*, vol. 30, no. 3, pp. 543–552, 2006.

[7] M. J. Pring, Technical Analysis Explained: The Successful Investor's Guide to Spotting Investment Trends and Turning Points. New York, NY, USA: McGraw-Hill, 1991, vol. 4.

[8] G. Susto, A. Schirru, S. Pampuri, S. McLoone, and A. Beghi, "Machine learning for predictive maintenance: A multiple classifiers approach," *IEEE Trans. Ind. Informat.*, Aug. 2014.

[9] W. E. Vinje and J. L. Gallant, "Sparse coding and decorrelation in primary visual cortex during natural vision," *Science*, vol. 287, no. 5456, pp. 1273–1276, 2000.

[10] B. A. Olshausen and D. J. Field, "Sparse coding with an overcomplete basis set: A strategy employed by v1?" *Vis. Res.*, vol. 37, no. 23, pp. 3311–3325, 1997.

[11] T. Bai and Y. Li, "Robust visual tracking using flexible structured sparse representation," *IEEE Trans. Ind. Informat.*, vol. 10, no. 1, pp. 538–547, Feb. 2014.

[12] M. Elad, M. Figueiredo, and Y. Ma, "On the role of sparse and redundant representations in image processing," *Proc. IEEE*, vol. 98, no. 6, pp. 972–982, Jun. 2010.

[13] Y. Deng, Q. Dai, and Z. Zhang, "Graph laplace for occluded face completion and recognition," *IEEE Trans. Image Process.*, vol. 20, no. 8, pp. 2329–2338, Aug. 2011.

[14] J. Suo, Y. Deng, L. Bian, and Q. Dai, "Joint non-Gaussian denoising and superresolving of raw high frame rate videos," *IEEE Trans. Image Process.*, vol. 23, no. 3, pp. 1154–1168, Mar. 2014.

[15] S. Li, L. D. Xu, and X. Wang, "A continuous biomedical signal acquisition system based on compressed sensing in body sensor networks," *IEEE Trans. Ind. Informat.*, vol. 9, no. 3, pp. 1764–1771, Aug. 2013.

[16] Y. Deng, Y. Liu, Q. Dai, Z. Zhang, and Y. Wang, "Noisy depth maps fusion for multiview stereo via matrix completion," *IEEE J. Sel. Topics Signal Process.*, vol. 6, no. 5, pp. 566–582, Sep. 2012.

[17] X. Hu *et al.*, "Robust and accurate transient light transport decomposition via convolutional sparse coding," *Opt. Lett.*, vol. 39, no. 11, pp. 3177–3180, 2014.

[18] J. Mairal, F. Bach, and J. Ponce, "Task-driven dictionary learning," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 34, no. 4, pp. 791–804, Aug. 2012.

[19] J. Mairal, F. Bach, J. Ponce, G. Sapiro, and A. Zisserman, "Supervised dictionary learning," in *Proc. Adv. Neural Inf. Process. Syst. Conf.*, 2008, pp. 1033–1040.

[20] J. Yang, K. Yu, Y. Gong, and T. Huang, "Linear spatial pyramid matching using sparse coding for image classification," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR'09)*, 2009, pp. 1794–1801.

[21] G. S. Sivaram, S. K. Nemala, M. Elhilali, T. D. Tran, and H. Hermansky, "Sparse coding for speech recognition," in *Proc. IEEE Int. Conf. Acoust. Speech Signal Process. (ICASSP)*, 2010, pp. 4346–4349.

[22] J. F. Bard, *Practical Bilevel Optimization: Algorithms and Applications*. New York, NY, USA: Springer, 1998, vol. 30.

[23] L. Ponta, E. Scalas, M. Raberto, and S. Cincotti, "Statistical analysis and agent-based microstructure modeling of high-frequency financial trading," *IEEE J. Sel. Topics Signal Process.*, vol. 6, no. 4, pp. 381–387, Aug. 2012.

[24] R. S. Sutton and A. G. Barto, *Introduction to Reinforcement Learning*. Cambridge, MA, USA: MIT Press, 1998.

[25] I. Grondman, L. Busoniu, G. A. Lopes, and R. Babuska, "A survey of actor-critic reinforcement learning: Standard and natural policy gradients," *IEEE Trans. Syst. Man Cybern. C, Appl. Rev.*, vol. 42, no. 6, pp. 1291–1307, Nov. 2012.

[26] G. Tesauro, "Td-gammon, a self-teaching backgammon program, achieves master-level play," *Neural Comput.*, vol. 6, no. 2, pp. 215–219, 1994.

[27] E. Pednault, N. Abe, and B. Zadrozny, "Sequential cost-sensitive decision making with reinforcement learning," in *Proc. 8th ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining*, 2002, pp. 259–268.

[28] Y. Deng, Y. Li, Y. Qian, X. Ji, and Q. Dai, "Visual words assignment via information-theoretic manifold embedding," *IEEE Trans. Cybern.*, vol. 44, no. 10, pp. 1924–1937, Oct. 2014.

[29] B. A. Olshausen *et al.*, “Emergence of simple-cell receptive field properties by learning a sparse code for natural images,” *Nature*, vol. 381, no. 6583, pp. 607–609, 1996.
[30] H. Lee, A. Battle, R. Raina, and A. Ng, “Efficient sparse coding algorithms,” *Adv. Neural Inf. Process. Syst.*, vol. 19, p. 801, 2007.
[31] J. Yang, Z. Wang, Z. Lin, S. Cohen, and T. Huang, “Coupled dictionary training for image super-resolution,” *IEEE Trans. Image Process.*, vol. 21, no. 8, pp. 3467–3478, Aug. 2012.
[32] P. J. Werbos, “Backpropagation through time: what it does and how to do it,” *Proc. IEEE*, vol. 78, no. 10, pp. 1550–1560, Oct. 1990.
[33] M. P. Taylor and H. Allen, “The use of technical analysis in the foreign exchange market,” *J. Int. Money Finance*, vol. 11, no. 3, pp. 304–314, 1992.
[34] M. Dempster, T. W. Payne, Y. Romahi, and G. Thompson, “Computational learning techniques for intraday fx trading using popular technical indicators,” *IEEE Trans. Neural Netw.*, vol. 12, no. 4, pp. 744–754, Jul. 2001.
[35] M. Kearns and Y. Nevmyvaka, High frequency trading-New realities for traders, markets and regulators, in *Machine Learning for Market Microstructure and High Frequency Trading*. Risk Books, 2013.
[36] H. Bessembinder and K. Chan, “The profitability of technical trading rules in the Asian stock markets,” *Pacific-Basin Finance J.*, vol. 3, no. 2, pp. 257–284, 1995.

Yue Deng received the B.E. degree (Hons.) in automatic control from Southeast University, Nanjing, China, in 2008, and the Ph.D. degree (Hons.) in control science and engineering from the Department of Automation, Tsinghua University, Beijing, China, in 2013.

He was a Visiting Scholar with the School of Computer Science, Carnegie Mellon University, Pittsburgh, PA, USA, from 2010 to 2011. Currently, he is an Associate Professor with the School of Electronic Science and Engineering, Nanjing University, Nanjing. His research interests include machine learning and signal processing.

Dr. Deng was a recipient of the Springer Best Thesis Award in 2014 and the Microsoft Fellowship in 2010.

Youyong Kong received the B.S. and M.S. degrees in computer science and engineering from Southeast University, Nanjing, China, in 2008 and 2011, respectively, and the Ph.D. degree in imaging and diagnostic radiology from the Chinese University of Hong Kong, Shatin, Hong Kong, in 2014.

Currently, he is an Assistant Professor with the College of Computer Science and Engineering, Southeast University. His research interests include machine learning, and medical image processing and analysis.

Feng Bao received the B.E. degree in electronics and information engineering from Xidian University, Xi’an, China, in 2014. Currently, he is pursuing the M.S. degree in automation at Tsinghua University, Beijing, China.

His research interests include machine learning and signal processing.

Qionghai Dai (SM’05) received the B.S. degree in mathematics from Shanxi Normal University, Xi’an, China, in 1987, and the M.E. and Ph.D. degrees in computer science and automation from Northeastern University, Shenyang, China, in 1994 and 1996, respectively.

Since 1997, he has been with the Faculty at Tsinghua University, Beijing, China. He is now a Cheung Kong Professor with Tsinghua University, and is the Director with the Broadband Networks and Digital Media Laboratory. His research interests include signal processing and computer vision.