Proceedings of the 2007 IEEE Symposium on Approximate
Dynamic Programming and Reinforcement Learning (ADPRL 2007)

Short-term Stock Market Timing Prediction under Reinforcement Learning Schemes

Hailin Li, Cihan H. Dagli, and David Enke
Department of Engineering Management and Systems Engineering
University of Missouri-Rolla
Rolla, MO USA 65409-0370
E-mail: {hl8p5, dagli, enke}@umr.edu

Abstract— **There are fundamental difficulties when only using a supervised learning philosophy to predict financial stock short-term movements. We present a reinforcement-oriented forecasting framework in which the solution is converted from a typical error-based learning approach to a goal-directed match-based learning method. The real market timing ability in forecasting is addressed as well as traditional goodness-of-fit-based criteria. We develop two applicable hybrid prediction systems by adopting actor-only and actor-critic reinforcement learning, respectively, and compare them to both a supervised-only model and a classical random walk benchmark in forecasting three daily-based stock indices series within a 21-year learning and testing period. The performance of actor-critic-based systems was demonstrated to be superior to that of other alternatives, while the proposed actor-only systems also showed efficacy.**

I. Introduction

A series-based stock price is a typical nonstationary stochastic process having no constant mean level over time for it to remain in equilibrium. The idea of using a mathematical model to describe the dynamics of such a process and then forecast future prices from current and past values is well established by substantial research in nonlinear time series analysis [1]. Although many academics and practitioners have tended to regard this application with a high degree of skepticism, there has been reliable evidence [2] that markets may not be fully efficient and the random walk hypothesis could be rejected. Proponents of technical analysis have thus made serious attempts in the past decades to apply various statistical models, and more recently, artificial intelligent (AI) methods to test the predictability of stock markets.

Notably, most publicized methods in the literature employ a supervised learning philosophy in the context of regression, i.e. the problem is usually formalized as inferring a forecast function based upon available training sets, and then evaluating the obtained function by how well it generalizes. These efforts, however, have their inherent limitations due to the underlying assumption that price series often exhibit homogeneous nonstationary. In reality, stock markets experience speculative bubbles and crashes which can not be explained by the patterns generalized from history [3]. Forecasting the real market trend of a stock other than its "expectations" in the future by supervised approaches alone, is fundamentally difficult.

Reinforcement Learning (RL), or Approximate Dynamic Programming (ADP) in a broader RL sense, has so far received only limited attention in computational finance community. Applications to date have concentrated on optimal management of asset and portfolios [4], as well as derivative pricing and trading systems [5], given the fact that they can be directly treated as a class of learning decision and control problems in terms of optimizing relative performance measures over time under constraints. Such research continues earlier efforts [6] in which similar problems are formulated from the standpoint of dynamic programming and stochastic control. Financial time series forecasting, on the other hand, appears hard to be abstracted as a straightforward problem of goal-directed learning from interaction. However, this task involves specific long-term goals of market profitability and measurable short-term performance (reward) of the adopted prediction model. And, it is generally agreed that the path that a stock's prices follow is a certain Markov stochastic process (e.g. Geometric Brownian Motion). Such features reveal the possibility for integrating RL techniques to further explore the dynamics of sequential price series movements without explicit training data. Few studies have been made in this area.

In this paper, we present reinforcement-oriented schemes for forecasting short-term stock price movements by using actor-only and actor-critic RL methods, respectively. A comparison study is then implemented to examine the performance of a variety of strategies for predicting three daily-based stock indices series within a 21-year learning and testing period from 1984 to year 2004. Furthermore, the nonparametric Henriksson-Merton test is used to analyze the short-term market timing abilities of two reinforcement schemes at the 5% level.

II. Fundamental issues for system development

Development begins with using observations at time t from a stock price series Z_t to forecast its value at some future time $t+l$, where $l=1,2,\ldots$. It is a typical example of noisy time series prediction. Here the observations are supposed to be available at discrete intervals of time. This problem can be naturally regarded as to infer a prediction function $\hat{z}_t(l) = f(\mathrm{Z}_t)$, based on a training set D_t generated from training sample $Z_t = \{z_t \quad z_{t-1} \quad z_{t-2} \quad \cdots \quad z_{t-k}\}$. The obtained function is evaluated by how accurately it performs on new data which are assumed to have the same distribution as the training data. Such supervised learning philosophy results in the predominance of statistical models (including closely related artificial neural network systems and kernel-based learning methods) in this application field (see, e.g., [7]-[9] and references therein). At any given time, the function $f(Z_t)$ has fixed structure and depends upon a set of parameters β. The prediction function then becomes $f(Z_t, \beta)$ and its result relies on the estimation of β.

These methods, though, face inherent difficulties. First, control of the complexity of the learned function is the key for a model to achieve good generalization. Modelling based on large training sets tend to follow irrelevant properties embedded in nonstationary data (overfitting), while small training sets might create an overly simple mapping which is not enough to capture the true series dynamics (underfitting). Second, time series prediction requires a model to address the temporal relationship of the inputs. With highly noisy data, the typical approaches that are adopted, such as recurrent neural networks (RNNs), are likely to only take into account short-term dependencies and neglect long-term dependencies. Finally, supervised learning used for forecasting essentially is about inferring an underlying probability distribution solely from a finite set of samples and is well known as a fundamentally ill-posed problem. The obtained model lacks the exploration ability to capture out-of sample dynamics.

In the financial area, the simple yet profound idea revealed by the capital asset pricing model (CAPM) holds: the expectation of reward / price is regulated by inherent market rules and can be estimated. The aforementioned supervised learning forecasting efforts have focused on exploiting the underlying market inertia so that a more accurate prediction for the target's expectation value in the future can be generated. The output of such models can be viewed as the "rational" portion of the actual realized price. On the other hand, what investors really care about is the actual realized stock return (or loosely speaking, realized trend). Although ample research [10] has been done in value investing theory and technical analysis to support the non-random walk theory, the existing explanations for realized price behaviours tend to be the after-the-fact story. Like their statistics counterparts, most AI forecasting models are concentrating on generalizing the price behaviours from huge available history data. Supervised learning is extremely useful to catch and adapt to the market inertia which is repeatable within a certain time window. Much of the present effort has stopped here and regarded all differences between actual prices and the corresponding expectations as unpredictable noise. This is probably not true.

In reality, markets are neither perfectly efficient nor completely inefficient. All markets are efficient to a certain extent, some more so than others [11]. Rather than being an issue of black or white, a more appropriate financial time series predictive model need to consider both sides. Stock markets often act in some “strange” motions that change the short-term price trajectory other than just making it follow the rule of market inertia. Individual investors make all kinds of decision directed by all kinds of investment philosophy at every possible time step. The “irrational” part of them can be synthesized daily as a collective behaviour that actually drives day-to-day price fluctuations, and it behaves in predictable ways to some degree [3]. For instance, directed by the long-term goal of profitability, investors tend to expect rising prices, miss price jumps, and learn from experience [12]. In essence, the market movement in next time step is closely related to its current state. Modern RL design is attempting to solve this class of learning decision and control problems that no supervised learning approaches can handle.

III. Reinforcement-oriented Forecasting Framework

The proposed reinforcement-oriented stock forecasting framework is depicted in Fig. 1. At any given time t, the prediction of future prices $\hat{z}(t+l)$ is determined by outputs from both supervised and reinforcement models as following:

$$\hat{z}(t+l) = \hat{z}_{SL}(t+l) + \hat{z}_{RL}(t+l) \tag{1}$$

where $l = 1, 2, \ldots$.

Specifically, $\hat{z}_{SL}(t+l)$ is obtained from the continuous nonlinear function inferred from a training set D_t:

$$\hat{z}_{SL}(t+l) = f(Z_t, t, \beta_{SL}(t)) \tag{2}$$

They can be viewed as the unobserved underlying price expectations which strictly follow market inertia.

Meanwhile, the reinforcement model receives the current input state $s_t \in \mathbf{S}$, where $\mathbf{S}$ is the set of all possible input states in the stock market environment, and generates $\hat{z}_{RL}(t+l)$ which can be viewed as the "extra" value imposed by the synthesized investors' “abnormal” decision. $\hat{z}_{RL}(t+l)$ are determined by a reinforcement policy π which is a mapping from a s_t to its correspond action.

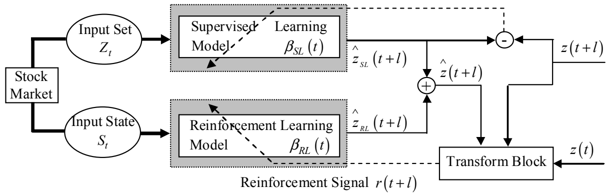

Fig.1. The proposed reinforcement-oriented forecasting framework

π is represented through the structure (β_{RL}) and is dependent or independent of the value function based on adopted RL / ADP techniques.

This architecture is adapted using mixed learning algorithms, i.e. supervised learning and reinforcement learning, respectively. At $t+l$, the supervised learning method is adopted in the first learning phase. The differences between $\hat{z}_{SL}(t+l)$ and the actual available prices $z(t+l)$ are used to generate required derivatives such that values of free parameters in the block β_{SL} can be trained. In the second learning phase, the training in the supervised learning part is frozen and the reinforcement learning method is applied to further trace the portion of the actual stock return that results from "irrational" investment behaviours (i.e. $z(t+l)-\hat{z}_{SL}(t+l)$). A short-term reinforcement signal $r(t+l)$ is needed and established by transforming the error term among $z(t)$, $\hat{z}(t+l)$ and $z(t+l)$. Properly designed $r(t+l)$ will show the quality of $\hat{z}_{RL}(t+l)$ toward the emphasis of prediction. In this learning phase, the parameters β_{RL} will keep evolving online and performance of the RL model should be improved gradually as the learning proceeds. The solid lines in Fig. 1 represent signal flow, while the dashed lines are the paths for parameter tuning.

Supervised learning has the advantages of fast convergence in structure and parameter learning so that it can be employed first to exploit the best interpolation for market inertia. Reinforcement learning techniques are then applied in the significant reduced search space to explore and imitate synthesized sequential "irrational" investment decision series without an explicit training sample. This way, the essential disadvantages of supervised learning are alleviated by the fine-tuning process of reinforcement learning. Furthermore, since the search domain of the reinforcement learning is greatly reduced in advance, learning can be accelerated and premature convergence may also be potentially avoided. In brief, such integration should help to make the forecasting problem less ill-posed.

IV. Supervised Learning Model

A numbers of different supervised learning methods have been applied to generalize the temporal relationship of the financial time series with varying degree of success. Among them, multi-layer perceptrons (MLPs) and recurrent neural networks (RNNs) are two of the most common choices to infer an underlying probability distribution from a small set of training data. In practice, RNNs often perform better than MLPs to address temporal issues since the learning of RNNs is biased towards patterns that occur in temporal order instead of random correlations. Therefore, RNNs are considered as a supervised learning approach in proposed mixed learning algorithms.

The Elman recurrent network [13] is chosen because it is a simplified realization of general RNNs. The architecture is similar to the standard feed-forward form except it also includes one or more context (recurrent) units which store the previous activations of the hidden units and then provides feedback to the hidden units in a fully connected way.

A raw price series is pre-processed and the modelling is based on the first order differences.

$$\hat{\delta}(t+l) = f\left(\boldsymbol{\delta}_t, t, \beta_{SL}(t)\right) \tag{3}$$

$$\hat{z}_{SL}(t+l) = z_t + \hat{\delta}(t+l) \tag{4}$$

where $\boldsymbol{\delta}_t = \{\delta_t,\ \delta_{t-1},\ \cdots,\ \delta_{t-n}\}$, $\delta_t \triangleq z_t - z_{t-1}$.

The training of the Elman network is implemented in batch mode, updating the model using historical data within a selected supervised training window. This is an intuitive method since typically one fixed market inertia would not last for longer amounts of time. After training, a network is used to predict the next l prices. The entire training window will then move forward l time steps (i.e. the length of supervised testing window) and the training process will be repeated. During any given training period, the goal is to minimize the squared error between all the network outputs and corresponding actual first order price differences by adjusting the weights in the network, as defined by (5).

$$\text{minimize } E[\beta_{SL}] = \frac{1}{2}\sum_t \sum_l \left(\delta_{t+l} - \hat{\delta}(t+l)\right)^2 \tag{5}$$

V. Schemes of Reinforcement Learning Model

While the deterministic part of a price can be detected and assessed by supervised learning approaches, the more irregular portion of price evolution is also to a certain degree predictable by means of current RL / ADP tools. The ultimate objective of a reinforcement learning model is to fine-tune predictions so that the goal of short-term market timing can be reached better. Assessed by the sum of the discounted immediate rewards, the system's total expected long-term reward from time t is as following:

$$R(t) = \sum_{k=1}^{\infty} \alpha^{k-1} r(t+k) \tag{6}$$

where α is a discount factor in the infinite continual forecasting problem $(0<\alpha<1)$. Like many financial applications, there is no inherent delay in the financial forecasting task in measuring the system's short-term performance. Such performance is illustrated by a properly designed immediate reinforcement signal r . Note that traditional goodness-of-fit performance criteria in time series analysis are not necessarily suitable in the financial sense. Investors are more concerned about the forecastability in terms of profitability. r should be designed according to this concern.

As following, the reinforcement learning model will be constructed by two ADP techniques, respectively.

A. Actor-only RL Model

A measurable immediate short-term performance of the forecasting system enables the use of an actor-only RL to optimize the parameterized policy structure directly. Direct policy search without learning a value function is appealing in terms of the strengths in problem representation and computation efficiency. The recurrent reinforcement learning (RRL) algorithm in [14] is utilized here to maximize gradually accrued immediate rewards of prediction.

Considering $l = 1$, the actor-only RL model that takes into account the historical price series has the following stochastic decision function:

$$\hat{z}_{RL}(t+1) = F_t\left(\beta_{RL}(t); \hat{z}_{RL}(t), I_t; \varepsilon_t\right) \tag{7}$$

where $I_t = \left\{z_t,\ z_{t-1},\ \cdots;\ \hat{z}_{SL}(t),\ \hat{z}_{SL}(t-1),\ \cdots\right\}$ is the relevant available information set at time t, $\beta_{RL}(t)$ denotes the adjustable model parameters at time t, and ε_t is a random variable. A simple model can take the autoregressive form of:

$$\hat{z}_{RL}(t+1) = u\,\hat{z}_{RL}(t) + v\left(z_t - \hat{z}_{SL}(t)\right) + w\left(z_{t-1} - \hat{z}_{SL}(t-1)\right) + x \tag{8}$$

Model parameters β_{RL} thus become adjustable coefficients vector $\{u,\ v,\ w,\ x\}$. Immediate reward r is proposed in order to reflect the forecasting system's trade-off between traditional in-sample goodness-of-fit and more important profit-earning market timing ability:

$$r(t) = \frac{1}{3}\left[\exp\left(-\left(\frac{\hat{z}_{RL}(t) - \Delta z_t}{\sigma}\right)^2\right) + \frac{2\exp\left(\hat{z}_{RL}(t)\Delta z_t\right)}{\left(\exp\left(\hat{z}_{RL}(t)\Delta z_t\right) + \exp\left(-\hat{z}_{RL}(t)\Delta z_t\right)\right)}\right] \tag{9}$$

where $\Delta z_t = z_t - \hat{z}_{SL}(t)$. This reinforcement signal at time t consider the ability of $\hat{z}_{RL}(t)$ (i.e. output of RL model F_{t-1}) to both minimize magnitude error (the first single Gaussian function term in (9) denoted as $A(t)$) and catch the market trend (the latter term in (9) denoted as $B(t)$). In (9), the reward is weighted twice as much towards market timing. σ controls the sensitivity of system toward in-sample goodness-of-fit.

For a decision function $F\left(\beta_{RL}(t)\right)$, the aim of β_{RL} adaptation is to maximize the accrued performance utility U_t, as defined following:

$$U_t = \sum_{k=1}^{t} \gamma^{k-t} r(k) \tag{10}$$

where $\gamma > 1$ indicates that a short-term reward received k time steps in the past is worth only γ^{-k} times what it would be worth if it was received immediately. The gradient of U_t with respect to β_{RL} after a sequence of t prediction is:

$$\frac{dU_t(\beta_{RL})}{d\beta_{RL}} = \sum_{k=1}^{t} \frac{dU_t}{dr(k)}\left\{\frac{dr(k)}{dF_k}\frac{dF_k}{d\beta_{RL}} + \frac{dr(k)}{dF_{k-1}}\frac{dF_{k-1}}{d\beta_{RL}}\right\} \tag{11}$$

Due to temporal dependencies in decision function F,

$$\frac{dF_k}{d\beta_{RL}} = \frac{\partial F_k}{\partial \beta_{RL}} + \frac{\partial F_k}{\partial F_{k-1}}\frac{dF_{k-1}}{d\beta_{RL}} \tag{12}$$

Closely related to recurrent supervised learning, the adopted RRL algorithm is a simple online stochastic optimization which only considers the term in (9) that depends on the most recent reward. That is,

$$\frac{dU_t(\beta_{RL})}{d\beta_{RL}(t)} \approx \frac{dU_t}{dr(t)}\left\{\frac{dr(t)}{dF_t}\frac{dF_t}{d\beta_{RL}(t)} + \frac{dr(t)}{dF_{t-1}}\frac{dF_{t-1}}{d\beta_{RL}(t-1)}\right\} \tag{13}$$

Learning successively using the most immediate reward tends to be most effective. See also the discussions in [15]. Follow our definition of $r(t)$, a more simple form is obtained:

$$\frac{dU_t(\beta_{RL})}{d\beta_{RL}(t)} \approx \frac{dU_t}{dr(t)}\frac{dr(t)}{dF_{t-1}}\frac{dF_{t-1}}{d\beta_{RL}(t-1)} = \frac{dr(t)}{dF_{t-1}}\frac{dF_{t-1}}{d\beta_{RL}(t-1)} \tag{14}$$

$$\frac{dF_{t-1}}{d\beta_{RL}(t-1)} \approx \frac{\partial F_{t-1}}{\partial \beta_{RL}(t-1)} + \frac{\partial F_{t-1}}{\partial F_{t-2}}\frac{dF_{t-2}}{d\beta_{RL}(t-2)} \tag{15}$$

β_{RL} is then updated online using:

$$\Delta\beta_{RL}(t) = \rho\frac{dU_t\left(\beta_{RL}(t)\right)}{d\beta_{RL}(t)} \tag{16}$$

Equations (8), (9) and (14)-(16) constitute a proposed actor-only RL model and its online adaptation.

B. Actor-Critic RL Model

For gradient-based actor-critic RL methods, the “critic” served as a nonlinear function approximator of the external environment to critique the action generated by the “actor”. The critic network will iteratively adapt its weights to learn a value function which satisfies the modified Bellman equation. The “new” critic is then used to update the policy parameters of the “actor”. Under a more generic problem environment, such methods may have better convergence properties than both actor-only and critic-only methods in terms of convergence speed and constraints.

A group of ADP approaches named as adaptive critic designs (ACDs) fall into this RL category. ACDs [17] consist of three basic designs and their variations, i.e. Heuristic dynamic programming (HDP), Dual heuristic dynamic programming (DHP), Globalized dual heuristic dynamic programming (GDHP), along with their corresponding action dependent (AD) forms, respectively. While in HDP the critic only estimates the Bellman value function, it estimates the gradient of the value function in DHP and for GDHP, critic functions as the summation of its functionality in HDP and DHP.

In this paper, a modified ADHDP proposed in [16] is adopted to construct the actor-critic RL model in our forecasting system. Without sacrificing learning accuracy, this method is likely to produce more consistent and robust online learning under a large scale environment.

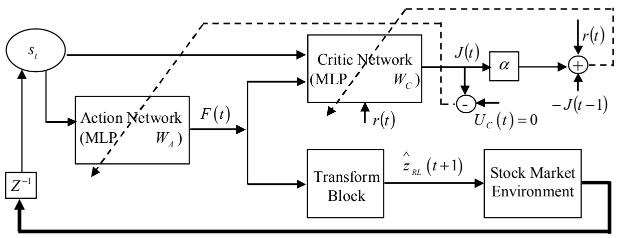

Fig.2. An actor-critic RL model in proposed stock forecasting system

The application is based on the block diagram as depicted in Fig. 2.

Both actor (action network) and critic (critic network) are similar MLPs in which one hidden layer is used for each network.

Given an input state s_t defined as:

$$s_t = \left[0.5A(t),\ \ 0.5B(t),\ \ C(t)\right]^T \tag{17}$$

where $C(t) = [\delta_t,\ \delta_{t-1},\ \ldots,\ \delta_{t-n}] / \|[\delta_t,\ \delta_{t-1},\ \ldots,\ \delta_{t-n}]\|_2$, the action network output $F(t)$ acts as a signal which implicitly demonstrates the influence of synthesized "irrational" investment decision for the actual price at time $t+1$.

$F(t)$ also served as part of the input vector $\left[s_t;\ \ F(t)\right]$ to the critic network.

The output of critic is an approximation (denoted as function J) for function V,

$$V^{\pi}(s) = E\left[\sum_{t=0}^{\infty} \alpha^t \Re_{s_t s_{t+1}}\left(\pi(s_t)\right) \Big| s_0 = s\right] \tag{18}$$

The weights of critic W_C are adapted to approximate the maximum of J to satisfy the modified Bellman equation:

$$J^*(s_t) = \max_{F(t)}\left\{J^*(s_{t+1}) + \Re_{s_t s_{t+1}}\left(F(t)\right) - U_0\right\} \tag{19}$$

where $\Re_{s_t s_{t+1}}\left(F(t)\right)$ (or, $r(t+1)$ if without the model of target MDP) is the next step reward incurred by $F(t)$ and U_0 is a heuristic term used to balance.

To update weights online, an adopted ADHDP utilizes the temporal difference of J to resolve the dilemma, i.e. the prediction error of the critic network is defined as $e_C(t) = \alpha J(t) - J(t-1) + r(t)$ instead of using the typical form $e_C(t) = J(t) - \alpha J(t+1) - r(t)$.

Consequently, the critic network tries to minimize the following objective function:

$$E_C(t) = \frac{1}{2} e_C^{\,2}(t) \tag{20}$$

The expression for its gradient-based weight's update thus becomes:

$$\Delta W_C(t) = -\eta_C(t)\left[\alpha J(t) - J(t-1) + r(t)\right] \frac{\partial J(t)}{\partial W_C(t)} \tag{21}$$

The objective of the action network is to maximize the J in the immediate future, thereby optimizing the overall reward expressed as (6) over the horizon of the problem.

r is defined as (23) to highlight the proper reinforcement direction.

$$r(t+1) = \begin{cases} 0 \text{ (success), if } (z_{t+1} - z_t)\,\&\left(\hat{z}(t+1) - z_t\right) > 0; \\ -1 \text{ (failure), otherwise.} \end{cases} \tag{22}$$

The desired value of the ultimate goal U_C is set to "0" along the timeline.

The action network tries to minimize the following objective function:

$$E_A(t) = \frac{1}{2} e_A^{\,2}(t) \tag{23}$$

$$e_A(t) = J(t) - U_C(t) = J(t) \tag{24}$$

The expression for the corresponding gradient-based weight's update thus becomes:

$$\Delta W_A(t) = -\eta_A(t) \frac{\partial J(t)}{\partial F(t)} \frac{\partial F(t)}{\partial W_A(t)} \tag{25}$$

$\eta_C(t) > 0$ and $\eta_A(t) > 0$ are the corresponding learning rate of two networks at time t.

The mapping from $F(t)$ to $\hat{z}_{RL}(t+1)$ is defined through a simple heuristics as follows:

$$\hat{z}_{RL}(t+1) = \begin{cases} \left|z_t - \hat{z}_{SL}(t)\right|, & \text{if } F(t) > T; \\ 0, & \text{if } \left|F(t)\right| \le T; \\ -\left|z_t - \hat{z}_{SL}(t)\right|, & \text{otherwise.} \end{cases} \tag{26}$$

where T is defined as small positive tolerance value.

C. Learning Procedure of the Proposed System

The sequential learning strategies in Table I summarize the whole hybrid training ideas for forecasting series-based stock price under two different reinforcement schemes.

Each strategy consists of both supervised learning and reinforcement learning cycles. At time t, it is necessary to always start with supervised-modelling first, alternating it with reinforcement-modelling.

For a forecasting system with actor-critic reinforcement learning, the adaptation of its W_C and W_A (Step 5.0) is carried out using incremental optimization. That is, for each iteration, unless the internal error thresholds have been met, (21) should be repeated at most N_C times to update W_C.

Action network’s incremental training cycle is implemented while keeping W_C fixed. (25) should be repeated at most N_A times to update W_A.

TABLE I: ITERATIVE LEARN-TO-FORECAST STRATEGIES

	Actor-only RL (RRL)	Actor-Critic RL (Modified ADHDP)
1.0	Initialize β_{RL}, i.e. coefficients $\{u \quad v \quad w \quad x\}$ in form (8).	Initialize β_{RL}, i.e. weights of both critic and action network W_C, W_A.
2.0	Initialize W_{Elman}, the length of supervised training window T, and the length of supervised testing window $l=1$. Let $t=T$.	
3.0	Set up a supervised training set from available $\boldsymbol{\delta}_t$ and train Elman network according to (5). Generate prediction $\hat{\delta}(t+1)$ as (3) and corresponding $\hat{z}_{SL}(t+1)$ for the testing window.	
4.0	Compute immediate RL signal $r(t)$ from (9), and $\hat{z}_{RL}(t+1)$ using form (8).	Compute the input state s_t from (9) and (17), the output of action network $F(t)$ based on $W_A(t)$, and $\hat{z}_{RL}(t+1)$ from (26). Calculate immediate RL signal $r(t)$ from (22).
5.0	Update β_{RL} by (14)-(16).	Update W_C from (21), and update W_A from (25).
6.0	Compute final prediction $\hat{z}(t+1)$ from (1). Let $t=t+1$. Continue from 3.0.	

VI. EMPIRICAL RESULTS

The results reported below were obtained by applying the systems described above to predict three daily-based stock series (i.e. closing price series adjusted for dividends and splits), namely S&P 500, NASDAQ Composite, and IBM within a 21-year learning and testing period from Jan.-03-1984 to Jun.-30-2004 (The exception is for NASDAQ which started from Oct.-11-1984).

The moving supervised training window (counted by 50 trading-days) for each index is fixed and followed by a one-day prediction (testing) window. The prediction is available every morning before the market opens.

For supervised learning, the size of adopted Elman network is controlled by both the numbers of input layer nodes and the number of hidden neurons.

During any given supervised training window, various lengths of input layers, ranging from 2 to 7 with an increment of 1, and hidden neuron numbers ranging from 4 to 20, with an incremental of 2, are experimented for each security. The choice of an optimal combination was made using the cross-validation approach, in which the data were further divided into a training sub-window (first 48 trading-days data) and a validation sub-window (final 2 trading-days data). All 54 candidate networks were trained using training sub-window data, and the one that generated the smallest root mean squared error for the validation sub-window was selected to perform the prediction in the following testing window. The best network structure is thus changeable along the timeline. Again, the raw daily series inputs are pre-processed by the first order differences method. Inputs were then normalized to zero mean and unit variance. The learning rate for all networks will be linearly reduced over the training period from an initial value of 0.75.

For the proposed actor-only and actor-critic reinforcement-oriented forecasting systems, the very first 50 series data will be pre-collected to initialize the hybrid learning strategies. Corresponding reinforcement learning model will function from the beginning of first supervised testing window, i.e. synthesized prediction and adaptation of RL model are started from the 51 trading-day. Corresponding free parameters $\beta_{RL}(RRL)$ and $\beta_{RL}(ADHDP)$ are initialized randomly. For the subsequent days, the previously learned values of parameters are used to start the training. The learning rate of the RRL algorithm (i.e. ρ in (16)) has been set to a fixed value of 0.09. Configurations of modified ADHDP require more tweaking work - in our experiments the learning rate $\eta_C(0)$ and $\eta_A(0)$ has always been tested from 0.5 while the start position of discount rate α is set to 0.9. Also, values for N_C and N_A were tried from 150 and 400, respectively.

Quantitative performance measures are used to evaluate the effectiveness of two reinforcement learning schemes in comparison to a supervised learning (i.e. Elman network-only) model, as well as a classic random walk benchmark which is the simplest, yet probably the toughest contender.

As mentioned earlier, the ultimate goal of stock series forecasting is profit earning. Traditional goodness-of-fit performance criteria are not capable of revealing the model's ability in market timing.

Therefore, besides three commonly used measures (i.e. Root Mean Squared Error RMSE , Mean Absolute Error MAE , and Mean Absolute Percentage Error MAPE), two direction accuracy indicators are introduced:

$$\text{DA1} = \frac{1}{T}\sum_{t=2}^{T} I\left[\left(z_t - z_{t-1}\right)\left(\hat{z}(t) - z_{t-1}\right)\right] \tag{27}$$

$$\text{DA2} = \frac{1}{T}\sum_{t=2}^{T} I\left[\left(z_t - z_{t-1}\right)\left(\hat{z}(t) - \hat{z}(t-1)\right)\right] \tag{28}$$

where $I(x)=1$ if $x>0$ and $I(x)=0$ if $x \le 0$. DA2 exhibits the coincidence between actual series trend and the synthesized prediction trend.

Precise comparison results are given in Table II in which above five performance measures are calculated for all three markets under the same testing period, that is, closing price series covering 3,775 trading days from Jul.-13-1989 to Jun.-30-2004.

TABLE II: COMPARISON OF DIFFERENT FORECASTING SYSTEMS IN THREE MARKETS/SECURITIES

		S&P 500	NASDAQ	IBM
RMSE	Random Walk	106.16	1395.12	2.45
	Elman Network	124.93	1800.90	3.15
	Actor-only	172.19	4536.78	9.45
	Actor-Critic	149.57	2016.11	4.18
MAE	Random Walk	6.48	19.34	1.01
	Elman Network	6.69	24.75	1.14
	Actor-only	7.47	71.10	2.19
	Actor-Critic	7.48	27.50	1.20
MAPE	Random Walk	0.74%	1.06%	2.92%
	Elman Network	1.01%	1.45%	3.83%
	Actor-only	0.86%	8.73%	10.49%
	Actor-Critic	0.87%	1.44%	2.80%
DA1	Random Walk	——	——	——
	Elman Network	50.14%	54.20%	53.99%
	Actor-only	59.37%	61.18%	59.37%
	Actor-Critic	68.62%	68.18%	62.31%
DA2	Random Walk	——	——	——
	Elman Network	52.16%	58.81%	47.86%
	Actor-only	55.13%	54.64%	56.24%
	Actor-Critic	63.64%	64.47%	58.47%

For each of actor-only and actor-critic RL models, 10 runs using the same configuration (note β_{RL} is always initialized randomly) were implemented to test the robustness of adopted RL approaches.

The best results toward DA1 and DA2 are already reported in Table II, and Table III provides the resulting descriptive statistics about the timing performances of hybrid forecasting systems using RRL algorithm and modified ADHDP, respectively. For random walk benchmark, the additional noise component at each time step is a zero mean Gaussian variable with a specified variance.

The first observation from Table II is that both random walk and supervised learning-only models have generated superb forecasts for IBM in terms of goodness-of-fit. The results are excellent for the S&P 500 index and still acceptable for the NASDAQ Composite for the same metric.

TABLE III: STATISTICS FOR SHORT-TERM TIMING PERFORMANCES OF REINFORCEMENT FORECASTING SCHEMES IN THREE MARKETS/SECURITIES

		S&P 500		NASDAQ		IBM	
		DA1 (%)	DA2 (%)	DA1 (%)	DA2 (%)	DA1 (%)	DA2 (%)
Actor-only	Ave.	54.31	54.82	58.55	53.19	56.18	54.53
	S.Dev.	1.83	0.72	3.19	1.53	1.12	1.95
	Min.	53.18	52.76	52.21	50.36	55.74	50.44
	Max.	59.37	55.13	61.18	54.64	59.37	56.24
Actor-Critic	Ave.	62.41	60.90	63.40	62.11	60.29	57.42
	S.Dev.	5.26	2.32	4.23	1.22	1.64	1.10
	Min.	54.16	57.19	57.66	61.04	57.09	55.13
	Max.	68.62	63.64	68.18	64.47	62.31	58.47

This point is reflected by the average values of three markets during the testing period (789.9189 for S&P 500, 1418.9586 for NASDAQ, and 49.5636 for IBM), the small relative RMSEs and MAEs, and very small MAPEs. Note that in all cases random walk models slightly outperform the supervised learning-only counterparts.

Secondly, the supervised learning-only systems' forecasts appear to have slight short-term market timing ability as indicated by the values of DA1 and DA2. The best case is for the NASDAQ Composite, i.e., it can predict whether the market is going up or down 58.81% of the time based on DA2 despite the fact that the same system provides the worst prediction in the sense of goodness-of-fit. These results support the claim that in the context of financial time series analysis, the most popular goodness-of-fit-based forecasting criterion does not necessarily translate into good forecastability in terms of earning profit.

Finally, the weak short-term market timing ability of supervised learning-only forecasting systems apparently suggest that much of the volatility in an actual price series can not be caught by the values of "expectations" generated from historical market inertia alone. Extra RL models are integrated into the forecasting systems in order to reveal the "irrational" investment decision series which drives much of actual day-to-day price fluctuation of a stock. Relative performances are quantitatively illustrated in Table II and Table III. According to DA1, in average, the proposed actor-only RL-based systems can successfully predict the market's daily trend by 4.17%, 4.35%, and 2.19% (in best, the numbers will be 9.23%, 6.98%, and 5.38%) higher than the supervised learning counterparts for three markets. The average increases are 2.66%, -5.62%, 6.67% based on DA2, respectively (2.97%, -4.17%, and 8.38% in best). It is evident that the adopted RRL algorithm was able to further adjust the prediction toward the direction of the real market trend to some extent. Meanwhile, we find that the proposed actor-critic RL-based system consistently outperforms the actor-only RL-based prediction scheme. Without sacrificing prediction accuracy with regard to the goodness-of-fit, the system results in substantial improvements in short-term market timing compared with the supervised learning-only model. In detail, average DA1-based performances increase by 12.27%, 9.2%, and 6.3%, respectively (in best, i.e. 18.48%, 13.98%, and 8.32% accordingly). Similar average improvements indicated by DA2 are 8.74%, 3.3%, and 9.56% (11.48%, 5.66%, and 10.61% in best), respectively.

The small standard deviations in Table III clearly demonstrate that the two online ADP approaches that were adopted can be robust, i.e. the performance is insensitive to free parameters such as initial values for weights of action / critic networks or coefficients of decision functions.

In addition, the nonparametric Henriksson-Merton test of market timing [18] is adopted to analyze the statistical significance of the correlation between forecasts of an RL model (the worst RRL and ADHDP models in 10 runs are selected to be representatives here) and the actual values of error if only the Elman network model is used.

TABLE IV: RL MODELS' SHORT-TERM MARKET TIMING STATISTICS FROM THE HENRIKSSON-MERTON TEST

	S&P 500	NASDAQ	IBM
Actor-only	2.9986 (0.0014)	2.3308 (0.0099)	2.8546 (0.0022)
Actor-Critic	5.3833 (0)	3.1485 (0.0008)	4.3933 (0)

The results are presented in Table IV. The values in the parentheses give the p-values of the null hypothesis of independence between two series (i.e. the RL model has no short-term timing ability). At 5% level, we reject the null hypothesis of no market timing under all scenarios and conclude that the short-term market timing abilities of all adopted RL models are significant.

VI. CONCLUSIONS

This paper provides a reinforcement learning-oriented architecture for short-term stock series movements' prediction. Fundamental difficulties exist when the whole "learn-to-forecast" process is based on the supervised learning-only philosophy. In this task it is vital, yet impossible that the training set for a model's learning is well distributed over the entire (input, target) space. For the supervised learning method, the exploration of the space relies heavily on the intrinsic disturbances (noise) embedded in financial series itself, and thus lacks direct control. In contrast, active exploration of the (input, action) space is an integral part of reinforcement learning approaches. The stochastic outputs generated by RL/ADP methods are evaluated by the properly designed feedback signal from the environment in order to guide the search for the best output. Furthermore, RL methods directly control the stochastic nature of the outputs to achieve stable learning behaviour, i.e. the exploratory variations for outputs will be decreased as the learning proceeds and the performance of the model improves. Moreover, the statement in Section II that stock investors' "abnormal" psychology does not seem to take a random walk provides the basis for the afterwards development of forecasting schemes. The proposed actor-critic RL-based systems consistently exhibit significant real short-term market timing ability without losing goodness-of-fit. This fact is not only consistent with the tenets of technical analysis and contradiction of the weak form of efficient market hypothesis, but also implies that there is much more predictability in three studied markets than just for their "expectations". The results seem to support the key insight that the "abnormal" part of investor psychology behaves in predictable ways to some degree and can be imitated at least partially by applying a reinforcement learning philosophy.

REFERENCES

[1] A. S. Weigend and N. A. Gershenfeld, Eds., *Time Series Prediction: Forecasting the Future and Understanding the Past.* New York: Addison-Wesley, 1994.

[2] A. W. Lo and A. C. MacKinlay, "Stock market prices do not follow random walks: Evidence from a simple specification test," *Rev. Financial Studies*, vol. 1, pp. 41-66, 1988.

[3] D. P. Porter and V. L. Smith, "Stock market bubbles in the laboratory," *Applied Mathematical Finance,* vol. 1, pp. 111-127, 1994.

[4] B. Van Roy, "Temporal-difference learning and applications in finance," in *Computational Finance 1999*, Y. S. Abu-Mostafa, B. LeBaron, A.W. Lo, and A. S. Weigend, Eds. Cambridge, MA: MIT Press, 2001, pp. 447–461.

[5] J. N. Tsitsiklis and B. Van Roy, "Optimal stopping of Markov processes: Hilbert space theory, approximation algorithms, and an application to pricing high-dimensional financial derivatives," *IEEE Trans. Automat.Contr.*, vol. 44, pp. 1840–1851, Oct. 1999.

[6] E. J. Elton and M. J. Gruber, "Dynamic programming applications in finance," *J. Finance*, vol. 26, no. 2, 1971.

[7] M. B. Priestley, *Nonlinear and Nonstationary Time Series Analysis*. New York: Academic, 1988.

[8] G. Zhang, B.E. Patuwo, and M. Y. Hu, "Forecasting with artificial neural networks: The state of art," *J. Int. Forecasting*, vol. 14, pp. 35-62, 1998.

[9] N. Christianini and J. S. Taylor, *An Introduction to Support Vector Machines and Other Kernel-Based Learning Methods*. Cambridge: Cambridge University Press, 2000.

[10] A. W. Lo and A. C. MacKinlay, *A Non-Random Walk Down Wall Street*. Princeton University Press, 1999.

[11] J. C. Hull, *Options, Futures and Other Derivatives.* Prentice-Hall, Englewood Cliffs, NJ, 4th edition, 2000.

[12] G. Caginalp, D. Porter, and V. L. Smith, "Overreactions, Momentum, Liquidity, and Price Bubbles in laboratory and field asset markets," *J. Psychology and Financial Markets*, vol. 1, No.1, pp. 24-28, 2000.

[13] J. L. Elman, "Finding structure in time," *Cogn. Sci.*, vol. 14, pp. 179–211, 1988.

[14] J. Moody and M. Saffell, "Learning to trade via direct reinforcement," *IEEE Trans. Neural Networks*, vol. 12, pp. 875-889, July 2001.

[15] P. Maes and R. Brooks, "Learning to coordinate behaviors," in *Proc. 8th Nat. Conf. Artificial Intell.*, July 29–Aug. 3, 1990, pp. 796–802.

[16] J. Si and Yu-Tsung Wang, "Online learning control by association and reinforcement," *IEEE Trans. Neural Networks*, Vol. 12, pp. 264-276, March 2001.

[17] D. V. Prokhorov and D. C. Wunsch II, "Adaptive critic designs," *IEEE Trans. Neural Networks*, vol. 8, pp. 997–1007, Sept. 1997.

[18] R. D. Henriksson and R. C. Merton, "On market timing and investment performance. II. statistical procedures for evaluating forecasting skills," *Journal of Business*, vol. 54, pp. 513-533, 1981.