"Q-Trading": Learning to Scalp Using Order Book Features

Kevin J. Wu

Columbia University, Department of Computer Science

05/11/2018

Introduction and Motivation

Objective: Explore the feasibility of using Q-learning to train a high-frequency trading (HFT) agent that can capitalize on short-term price fluctuations.

Why use reinforcement learning (RL)?

- Easily-quantifiable reward structure; trader's goal is to maximize profits.
- Agent's actions may affect not only on its own future rewards but also the future state of the market.
- Potential to learn strategies that adapt to changing market regimes.

Background: Related Work

Previous examples of applying RL methods to the financial markets:

- **Non-HFT:** Deep Q-learning applied to intraday and longer-term investment horizons (Y. Deng and Dai, 2017).
- **HFT:** Q-learning for optimal trade execution (Nevmyvaka et al., 2006). Q-learning for price prediction on equities data, (Kearns et al., 2010) and (Kearns and Nevmyvaka, 2013).

The main innovations of this project and its approach are:

1. Learning directly from order book features, as opposed to learning from time series of past prices and trades.
2. Applications of reinforcement learning to new markets (cryptocurrencies).

Background: Market Microstructure

A few basic definitions:

- *Market order:* A buy or sell order to be immediately executed at the best available (market) price.
- *Limit order:* A buy (sell) order to be executed at or below (above) a specified price.
- *Limit order book:* A list of unexecuted limit orders, aggregated by price.
- *Bid/Ask (Bid/Offer):* The highest price a buyer is willing to pay to purchase an asset (bid); or conversely, the lowest price a seller is willing to take to sell an asset (ask).
- *Market-taker/market-maker:* Two broad categories of traders; generally, market-makers provide liquidity by posting limit orders to buy/sell, while market-takers remove liquidity by submitting market orders that immediately execute against resting bids/asks.

Background: Market Microstructure

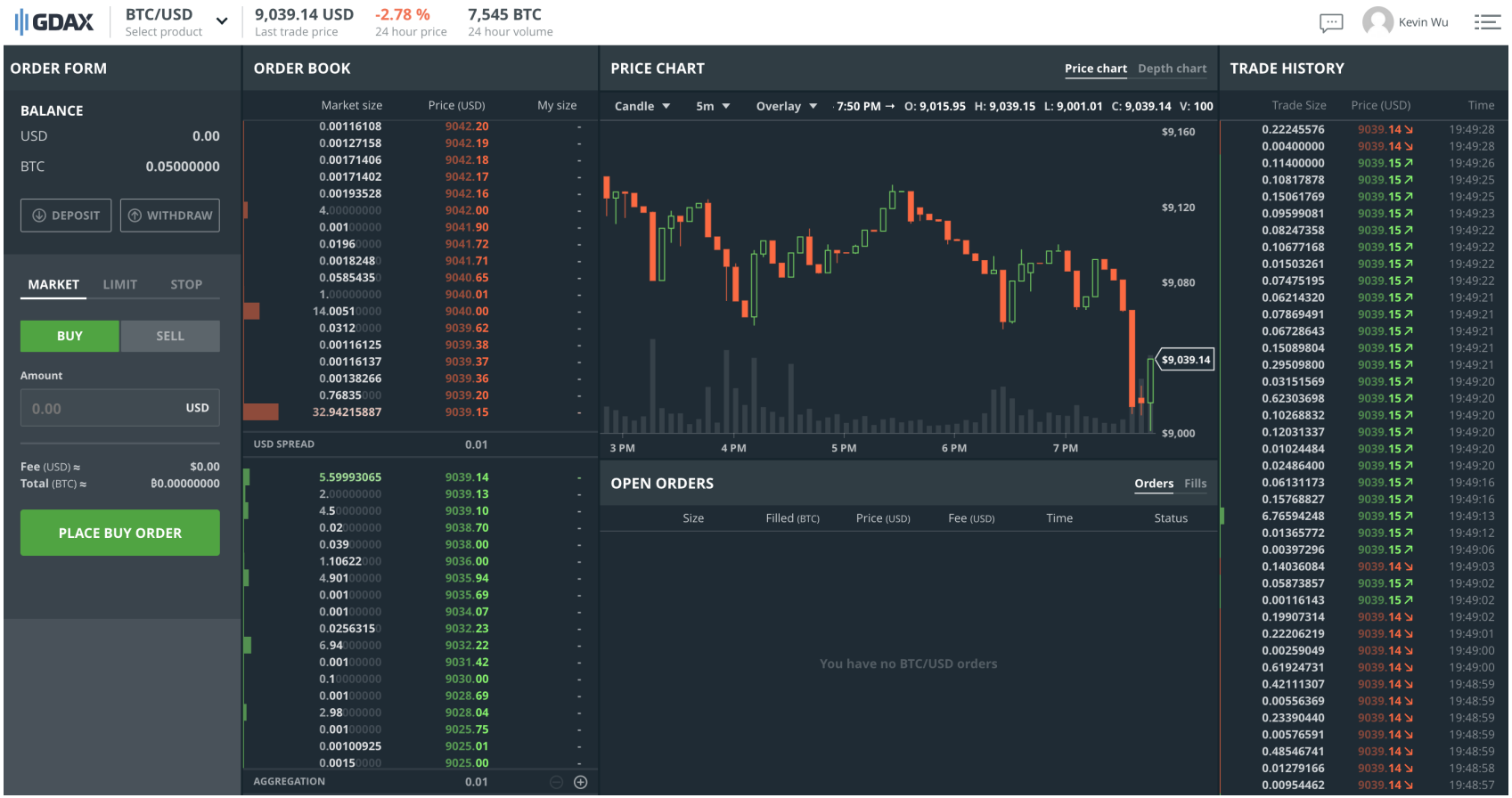

Figure 1: Snapshot of the limit order book for BTC-USD (Source: GDAX)

Methods: Q-Learning

We can represent the HFT setting as a Markov Decision Process (MDP), represented by the tuple (S, A, P, R, H, γ), with state space S, action space A, transition probabilities P, expected rewards R, time horizon H, and discount factor γ.

In this project, I use Q-learning (Watkins and Dayan, 1992) to learn the value of every state-action pair and acts greedily or ϵ-greedily with respect to these values. The optimal value function for every state-action pair, $Q^*(s, a)$, is defined recursively as:

$$Q^*(s, a) = R(s, a) + \gamma \sum_{s' \in S} P(s, a, s') \max_{a'} Q^*(s', a')$$

Reinforcement Learning Setting for HFT

State representation: Includes both the set of resting buy and sell orders represented by the limit order book and the agent's current inventory.

- *Order book state:* At time t, an order book K levels deep consists of bid prices $b_t^{(1)}, ..., b_t^{(K)}$, ask prices $a_t^{(1)}, ..., a_t^{(K)}$, bid volumes $u_t^{(1)}, ..., u_t^{(K)}$, and ask volumes $v_t^{(1)}, ..., v_t^{(K)}$. We denote the midpoint price as p_t, where $p_t = (b_t^{(1)} + a_t^{(1)})/2$.
  - Order book state is summarized by $\mathbf{x_t}$, the volume-weighted distance of each bid/ask price from the mid price, where $\mathbf{x_t} = (\mathbf{u_t^{(1)}}(\mathbf{b_t^{(1)}} - \mathbf{p_t}), ..., \mathbf{v_t^{(1)}}(\mathbf{a_t^{(1)}} - \mathbf{p_t}), ...)$, plus a bias term.
- *Agent state:* Total position/inventory at time t, $s_t \in \mathbb{R}$.

Action space: Discrete action space representing size and direction of order, i.e. $A = \{-1, 0, 1\}$ (sell, hold, and buy).

Reward: Profits and losses (PnL) from trading. This includes both realized profits, computed on a first-in-first-out (FIFO) basis, and unrealized profits at the end of H time steps.

Q-Function Approximation

To formulate a value function approximator, I first start with the idea that the expected future price change $\mathbb{E}[p_{t+1} - p_t]$ is directly related to buying/selling pressure in the order book (*book pressure*), encoded in the order book state vector, $\mathbf{x_t}$. The initial hypothesis is that this relationship linear, so that $\mathbb{E}[p_{t+1} - p_t] = \theta^{\mathbf{T}}\mathbf{x_t}$, plus a bias term.

$$Q(\mathbf{x_t}, \mathbf{s_t}, \mathbf{a}) = \tanh[(\mathbf{a} + \mathbf{s_t}) \cdot \theta^{\mathbf{T}}\mathbf{x_t}] - \lambda|\mathbf{s_t} + \mathbf{a}| \tag{1}$$

Besides encoding the idea of *book pressure*, the above value function has the following desirable properties:

- Linearity in $\mathbf{x_t}$ (order book features) given an action a makes financial sense.
- Squashing tanh non-linearity restricts Q-function to only predicting a general downward or upward directional movement, reducing the possibility of overfitting to abnormal market movements.
- Regularization term expresses the agent's preference for a small position s and short holding periods.

Empirical Evaluation: BTC-USD

Dataset: Level-2 order book data for US dollar-denominated Bitcoin prices (BTC-USD) on GDAX.

- 1.7 million timestamped order book snapshots spanning a period of 20 trading days from 04/11/2017 to 04/30/2017.

Training and hyperparameters:

- Episode length: 300 (5 minutes of trading time).
- # episodes: 1000
- Exploration: ϵ-greedy policy with $\epsilon = 1$ for the first 50 episodes and annealed to 0.001.
- Batch size: 15
- Discount factor (γ): 0.8, step size (α): 0.0005
- Action spaces: $A_s = \{-1, 0, +1\}$, and $A_l = \{-2, -1, 0, +1, +2\}$.

Market environment:

- Fill rate: Orders are filled at the best available bid/ask in proportion to the top-of-book quantity.
- "Naive" simulation: future order book snapshots are played back as is.

Results: Performance

$K = 10$ and an action space consisting of only 3 actions resulted in the best performance (2.03) for multiple values of regularization, but the standard error of the final average reward statistic is relatively large (1.40).

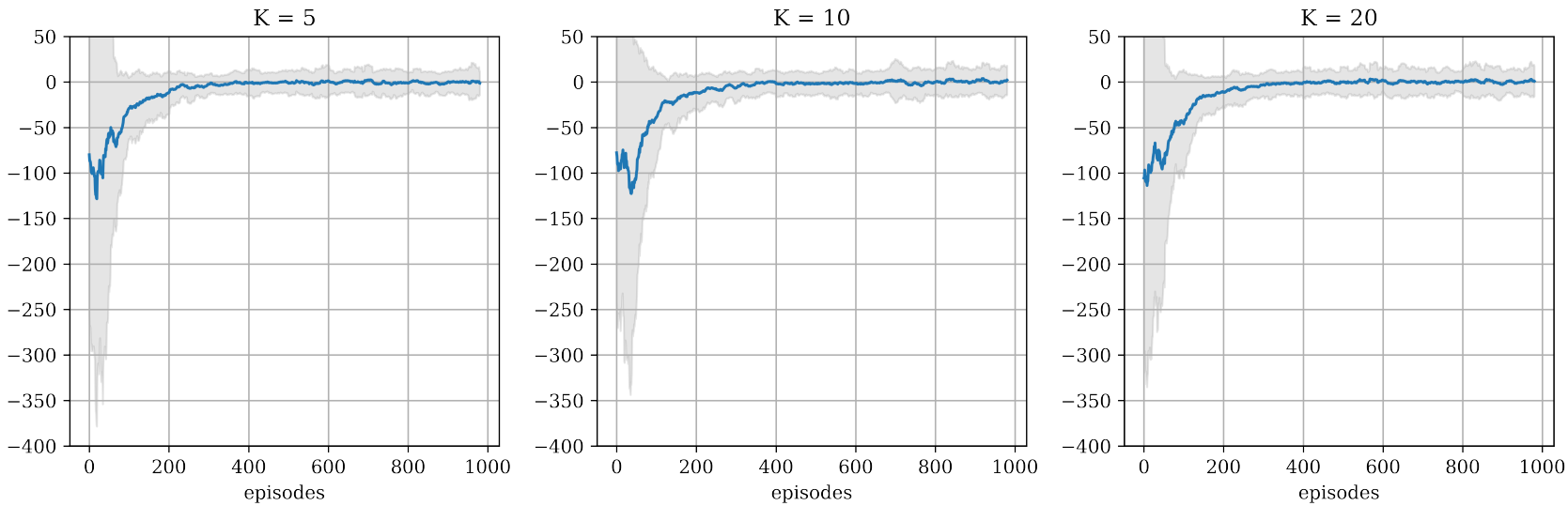

Figure 2: Final reward (as measured by a 20-episode moving average) obtained by Q-learning agents under different settings of K, averaged over 10 training runs. $\lambda = 0.05$, $\gamma = 0.8$. The gray bars represent one standard deviation from the average reward obtained at each episode.

Results: Performance

Q-learning performance relative to a baseline strategy (described below) was varied.

Baseline: Given an action space A, the baseline agent takes a random action in A at the beginning of the episode, and hold the position until the end of the episode.

		$\lambda = 0.1$			$\lambda = 0.05$		
		μ	σ	$s.e.\ (\mu)$	μ	σ	$s.e.\ (\mu)$
A_s	*baseline*	*-0.01*	*16.18*	–	*-0.01*	*16.18*	–
	K = 5	-0.25	17.01	1.38	-1.32	13.13	0.92
	K = 10	0.43	10.58	0.85	2.03	15.88	1.40
	K = 20	0.39	10.59	0.62	0.83	16.45	0.86
A_l	*baseline*	*0.04*	*28.18*	–	*0.04*	*28.18*	–
	K = 5	-1.29	14.93	1.27	-0.66	13.90	1.11
	K = 10	-0.71	14.33	1.03	-0.65	14.85	1.36
	K = 20	-1.08	14.35	1.29	0.12	11.51	0.77

Figure 3: Mean (μ) and standard error (σ) of final reward across 10 different agents ($N = 10$). σ is the average standard deviation of rewards accumulated in the last 20 episodes of each training run.

Results: Interpretation

For the "bid states", the effect of price/volume in the order book on the potential *upward* movement in price decreases deeper into the order book.

Similarly, for the "ask states," the effect of price/volume in the order book on the expected *downward* movement in price decreases deeper into the order book.

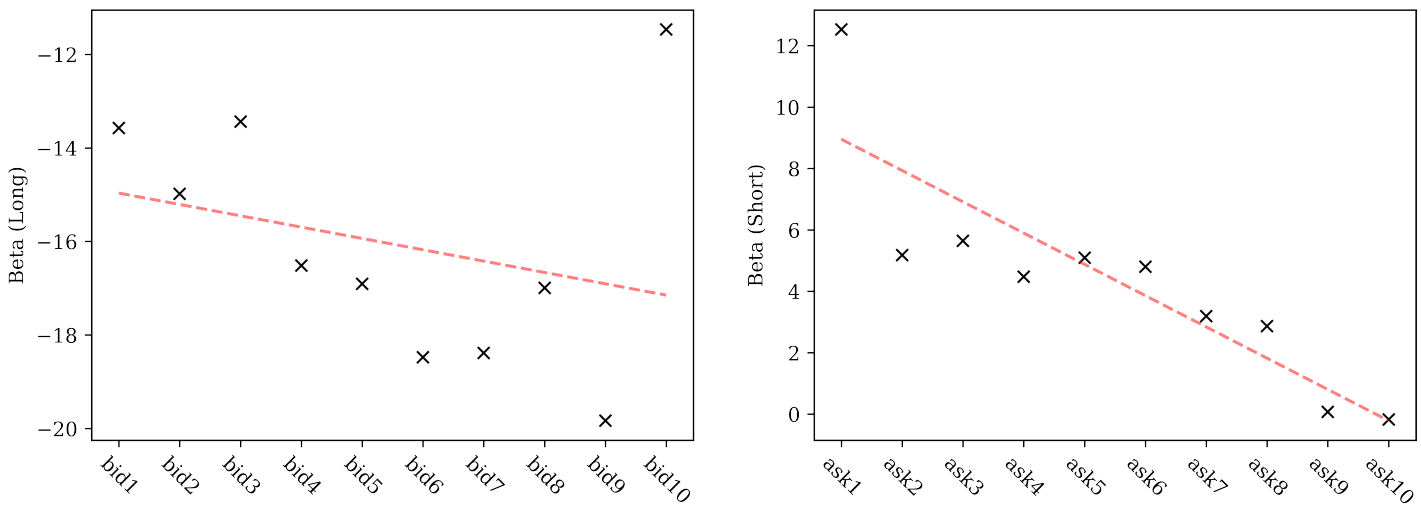

Figure 4: Coefficients on bid variables for positive-valued ("long") positions (left); coefficients on ask variables for negative-valued ("short") positions (right).

Results: Interpretation

Visualizing the Q-values, actions, and positions of the agent after training confirms that our function approximator is inducing the correct "scalping' behavior.

Placing a buy or sell order (shown in the middle graph) causes a discontinuity in the Q-values of the three actions (top graph). The overall position (bottom graph) is kept small at all times.

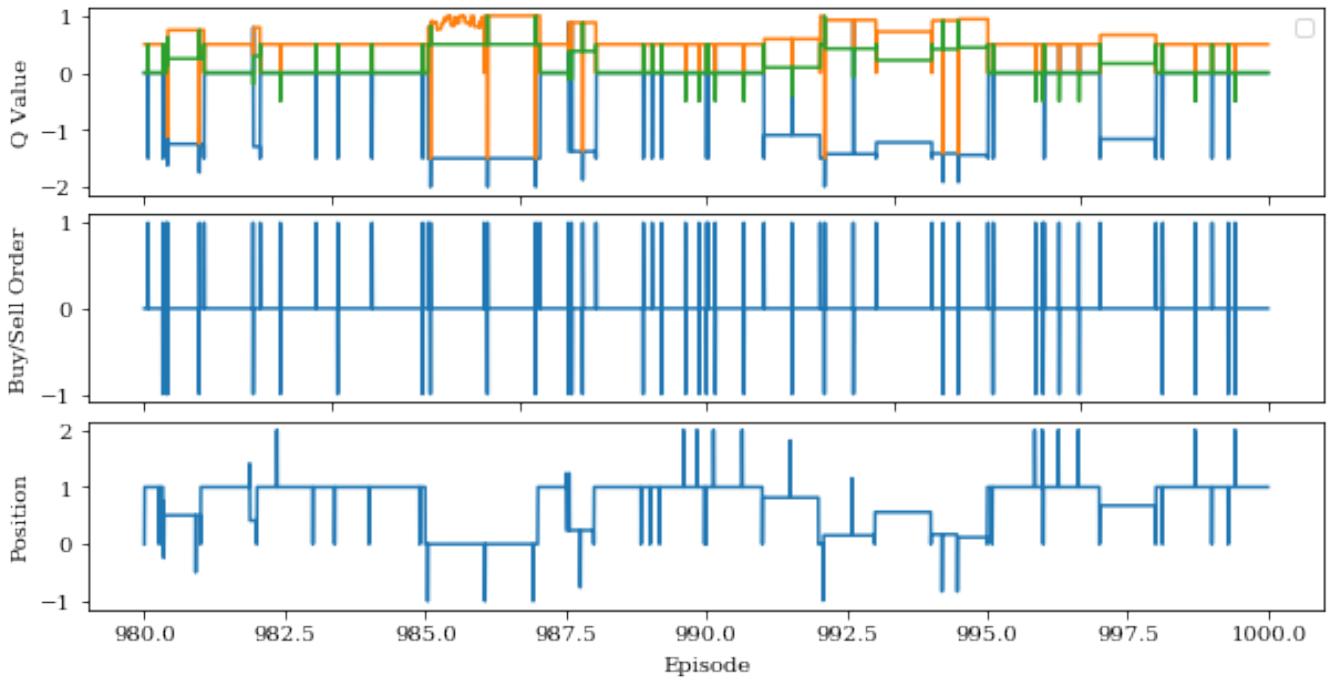

Figure 5: Q-values (top), actions (middle), and positions (bottom) of the agent over the last 20 episodes of a training run.

Future Work

Summary: Q-learning may be used to generate modest returns from an aggressive scalping strategy with short holding periods, while providing interpretable and intuitive results.

With the given framework for modeling the HFT setting, directions for future work are as follows (listed in order of priority):

- Alternative order book representations; i.e. aggregating orders by price intervals, or representing states as diffs between consecutive order book snapshots.
- More complex function approximators for the Q-function (i.e. tree-based methods, SVMs, neural networks), sacrificing some degree of interpretability for a more flexible model.
- A more sophisticated and realistic market simulation environment, based on either a set of heuristics or models taken from empirical studies of HFT behavior.
- More granular state updates. In an actual live market setting, state updates (or "ticks") occur upon the execution of any trade or any update to an existing order, at the frequency of milliseconds and even microseconds.

References

Michael Kearns and Yuriy Nevmyvaka. Machine learning for market microstructure and high frequency trading. 2013.

Michael Kearns, Alex Kulesza, and Yuriy Nevmyvaka. Empirical limitations on high frequency trading profitability. 5, 09 2010.

Yuriy Nevmyvaka, Yi Feng, and Michael Kearns. Reinforcement learning for optimized trade execution. *Proceedings of the Twenty-Third International Conference (ICML 2006)*, pages 673–680, 01 2006.

C. Watkins and P. Dayan. Q-learning. *Machine Learning*, 8(3): 279–292, 1992.

Y. Kong Z. Ren Y. Deng, F. Bao and Q. Dai. Deep direct reinforcement learning for financial signal representation and trading. *IEEE Transactions on Neural Networks and Learning Systems*, 28(3):653–664, 2017.