Информационные процессы, Том 19, № 2, 2019, стр. 122–131.

МАТЕМАТИЧЕСКИЕ МОДЕЛИ, ВЫЧИСЛИТЕЛЬНЫЕ МЕТОДЫ

Использование обучения с подкреплением в задаче алгоритмической торговли

Е.С. Пономарев[⋆], И.В. Оселедец[⋆,†], А.С.Чихоцкий[⋆]

[⋆] *Сколковский институт науки и технологий, Москва, Россия*
[†] *Институт вычислительной математики, Российская академия наук, Москва, Россия*

Поступила в редколлегию 10.06.2019

Аннотация—Развитие методов обучения с подкреплением открыло широкие границы по их использованию во многих областях, в том числе в алгоритмической торговле. В данной работе торговля на бирже интерпретируется в терминах среды с марковским свойством, состоящей из состояний, действий и наград.

В работе предложена и экспериментально протестирована система управления фиксированным обьемом финансового инструмента на базе метода асинхронного исполнителя-критика (asynchronous advantage actor-critic) с использованием ряда архитектур нейронных сетей. Исследовано применение рекуррентных слоев в данном подходе. Эксперименты производились на реальных обезличенных данных.

Лучшая архитектура продемонстрировала торговую стратегию для фьючерса индекса РТС (MOEX:RTSI) с прибыльностью 66% годовых с учетом комиссии. Исходный код проекта доступен по ссылке:
https://github.com/evgps/a3c_trading

КЛЮЧЕВЫЕ СЛОВА: обучение с подкреплением, нейронные сети, рекуррентные нейронные сети, алгоритмическая торговля.

1. ВВЕДЕНИЕ

Алгоритмическая торговля, рассматриваемая в данной работе, заключается в конструировании системы управления, способной покупать или продавать фиксированный объем финансового инструмента на бирже. Задача алгоритма - максимизировать стоимость суммарного портфеля или, иными словами, выгоду. В качестве финансового инструмента в проекте рассматривался фьючерс на индекс РТС, данные для экспериментальной части получены с одной крупной российской биржи. Комиссия учитывается согласно тарифам данной биржи для срочной торговли.

Создание алгоритма основано на использовании обучения с подкреплением (reinforcement learning), так как оно наилучшим образом подходит для задач с отложенной наградой. В отличие от методов обучения с учителем, оно не требует создания правил, по которым определенное действие должно считаться верным и с каким весом и позволяет использовать метрики, подсчитываемые для каждой стратегии на продолжительных промежутках времени, например коэффициент Шарпа [4]. В работе используется модифицированный алгоритм асинхронного исполнителя-критика (asynchronous advantage actor-critic [12]). В качестве функции аппроксимации были исследованы несколько архитектур искусственных нейронных сетей (ИНС). Показана зависимость результатов от вариации глубины сети, количества параметров (нейронов), а также добавления рекуррентного слоя, а именно долгой краткосрочной памяти (long short-term memory, LSTM [2]). Данные в виде обезличенных заявок агрегированы эмпирически в вектор признаков. Действие избирается раз в 60 секунд.

Отметим, что применимость всех разработанных методов была проверена экспериментально на реальных данных. Более подробно основные полученные результаты описываются в следующих подразделах.

В настоящее время существует множество алгоритмов обучения с подкреплением [5] и часть из них нашле применение в алгоритмической торговле, например Q-learning [6], Deep Q-learning [1, 7], recurrent reinforcement learning и policy gradient методы [8], [6], [9], REINFORCE [10] а также другие методы исполнитель-критик [5, 11]. Однако область быстро развивается и появляются новые алгоритмы.

В данной работе мы строим типичную для задач обучения с подкрепления среду [3], состоящую из **состояний**, набора возможных **действий** и функции **награды**. Кроме того, предполагается наличие марковского свойства у данного процесса. В качестве метода обучения был применен алгоритм **asynchronous advantage actor-critic (A3C)** [12], показавший эффективность на множестве датасетов, включая Atari 2600 [13]. Применение данного алгоритма к задаче биржевой торговли не было найдено в литературе. В процессе улучшения работы алгоритма был проведен поиск архитектуры искусственной нейронной сети, лежащей в основе метода, включая исследование использования рекуррентных нейронных сетей и ряд других оптимизций. Алгоритм был обучен и протестирован на реальных обезличенных данных торгов фьючерса индекса РТС на Московской Бирже (MOEX:RTSI).

Главные результаты работы:

- Было исследовано применение алгоритма обучения с подкреплением на основе глубокой нейронной сети.
- Modern asynchronous advantage actor-critic (A3C) algorithm is applied to these tasks.
- Был проведен поиск архитектуры ИНС
- Различные архитектуры метода были экспериментально опробированы и сравнены. Лучшая из них продемонстрировала устойчивую выигрышную стратегию с доходностью 66% годовых на 6 месяцах теста (с учетом комиссии биржи).

2. ПРЕДВАРИТЕЛЬНЫЕ СВЕДЕНИЯ

2.1. Формулирование задачи биржевой торогвли в терминах обучения с подкреплением

Рассмотрим процесс торговли на бирже более детально. Зафиксируем финансовый инструмент (фьючерс на индекс РТС), его возможный обьем, а также условимся принимать решение о купле/продаже раз в квант времени - одн минуту. Таким образом, мы получим систему, в которой на каждом шаге требуетсся выбрать одно из трех действий - желаемую позицию. Позиция может быть "длинная"(long), когда мы владеем фьючерсом, нейтральная, когда все активы переведены в деньги или "коротая"(short), когда мы занимаем зафиксированный ранее обьем с обязательством выкупить его позже по той цене, которая будет в будущем. Кроме того, на каждом шаге доступны новые данные, которые можно агрегировать в виде вектора состояния.

Зададим среду как марковский процесс принятия решений [14]: $M = \{\mathrm{S}, \mathrm{A}, \mathrm{P}, \gamma, \mathrm{R}\}$, где:

- $\mathbb{S} \in \mathbb{R}^m$ - пространство **наблюдаемых состояний**. На каждом шаге система биржа-агент находится в $s_t \in \mathrm{S}$. Конструктивно s_t в работе представлен как агрегация текущих заявок или как внутреннее состояние ячейки памяти LSTM. В последнем случае можно надеяться на большую информативность;
- $\mathbb{A} = \{-1, 0, 1\}$ - пространство **действий**. В задаче торговли действие - это желаемая позиция - длинная (держать ед. инструмента, лонг, 1), нейтральная (выход в наличные, 0) или

короткая (взять в долг ед. инструмента, шорт, -1).

На каждом шаге действие $a_t \in \mathrm{A}$ выбирается исходя из выработанной **политики** $\pi(a|s)$ - вероятности выбрать a в s;

– $\mathrm{P}(s'|s,a)$ - **переходная вероятность** предполагаемого марковского процесса;
– $\mathrm{R}(s,a)$ - функция **награды**. На каждом шаге агент получает награду, зависящую не только от текущего, но и предыдущих действий $r_t = \mathrm{R}(s_t, a_t)$
– $\gamma \in [0,1]$ - мультипликатор затухания, с которым следующая награда суммируется в общую награду за действие $R_t = \sum_{i=0}^{T} \gamma^i r_{t+i}$;

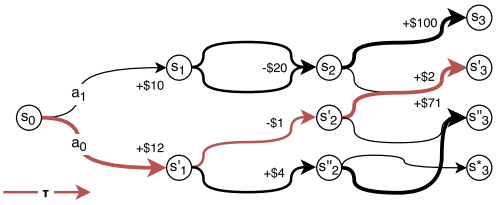

Рис. 1. τ-красный трек. a - действие, s - состояние, r - награда

Утверждение 1. Задача алгоритма - *выработать стратегию* $\pi : \mathbb{S} \to \mathbb{A}$, *максимизирующую математическое ожидание награды* ρ^π *:*

$$\rho^\pi = \rho^\pi(\theta) = \mathbb{E}[R|\pi] \to \max_\pi$$

$$\rho^\pi = \mathbb{E}[R|\pi(\theta)] = \int_{\mathrm{T}} p(\tau|\pi(\theta))R(\tau)d\tau \to \max_\theta$$

где трек $\tau = \{s_t, a_t\}_{t=0}^{T} \in \mathrm{T}$ - *реализация одной игры/эпизода (см рис. 1);*
$R(\tau)$ - *суммарная награда*

В качестве награды используется ее оценка - функция от действия a и состояния s $R(s,a)$:

$$R(s,a) \approx V^\pi(s) + A^\pi(s,a)$$

где $A^\pi(s,a)$ - функция преимущества(advantage) - насколько лучше избранное действие a средней оценки полезности состояния s:

$$A^\pi(s,a) := R(s,a) - V^\pi(s)$$

и $V^\pi(s)$ - оценка функции ценности состояния s для политики $\pi : \mathbb{S} \to \mathbb{A}$:

$$V^\pi(s) = \mathbb{E}_\pi R^\pi(s,a)$$

Таким образом, архитектура исполнителя определяет какой действие выбрать $a = \pi(s)$. Функция потерь для данной подсети пропорционален значению функции приемущества $A(s_t, a_t) = [r_t + \gamma V^\pi(s_{t+1})] - V(s)$. Градиент функции потерь для данной части сети будет равен (вывод можно посмотреть в [12]):

$$\text{Исполнитель: } \nabla_\theta J(\theta) \approx \mathrm{E}_\pi \left[\nabla_\theta \log \pi_\theta(a|s) A^\pi(s,a)\right]$$

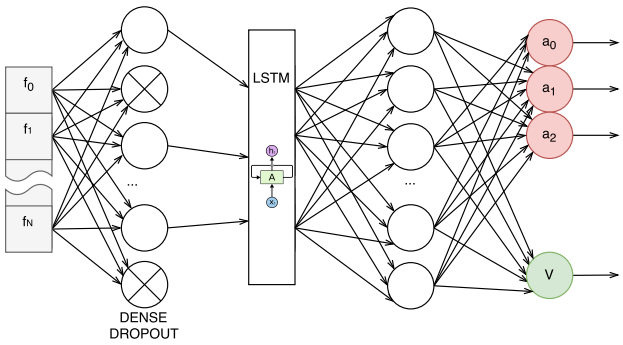

Рис. 2. Искусственная нейронная сеть, определяющая функцию ценности $V(s) = V(s|\theta)$ и функцию политики (исполнитель) $\pi(a|s) = \pi(a|s, \theta)$

Архитектура критика предсказывает то, какая награда ожидается в данном состоянии без разделения оценок по действиям $V^\pi(s_t) = \mathrm{E}_\pi R_t(\pi)$. Функция потерь для критика - ошибка в предсказании $V^\pi(s_t)$:

$$\text{Критик: } L_v = \sum_{i=1}^{T}(V_{target_i} - V_i)^2$$

$$V_{target_i} = \sum_{k=i}^{T} \gamma^{k-i} r_i$$

В рассматриваемой задаче P и R неизвестны, функция моментальной награды отражает изменение стоимости портфеля и учитывает комиссию за покупку/продажу инструмента. Т.е. если c_t - это стоимость всех активов агента на шаге t (количество денег, если бы он сейчас все продал, не учитывая комиссию), то $r_t = c_t - c_{t-1} - fee \cdot \mathbb{I}[a_t = a_{t-1}]$

Вся сеть тренируется методом обратного распространения ошибки. Функция потерь для всей сети общая и является линейной комбинацией функций потерь для исполнителя и критика (параметр $\alpha \in [0, 1]$):

$$L = \alpha \sum_{i=1}^{T}(V_{target_i} - V_i)^2 - \log \pi_\theta(a_i|s_i) A_i^\pi \tag{1}$$

В конечном итоге алгорим пытается максимизовать кумулютивную дисконированную награду

$$R_t = \sum_{i=0}^{T} \gamma^i r_{t+i},$$

где r_{t+i} - изменение портфеля за шаг. Количество шагов $T \sim 200$, т.к. большее количество членов не внесет существенного вклада в оценку ($\gamma^T \to 0$)

3. ПРОГРАММНАЯ РЕАЛИЗАЦИЯ

Запуск алгоритма исполнитель-критик (advantage actor-critic) в несколько асинхронных потоков показывает существенный прирост в качестве работы [12]. В целях исследований авторами была построена программная система обучения и тестирования. Система реализована на языке Python с использованием фреймворка TensorFlow [18]. Система состоит из двух глобальных процессов: обучения и тестирования(см. рис 3). Они связаны с помощью сохранения модели на жесткий диск и загрузки архитектуры и выученных параметров в системе тестирования. Выбор архитектуры является ключевой задачей и детально рассмотрен в секции численных результатов.

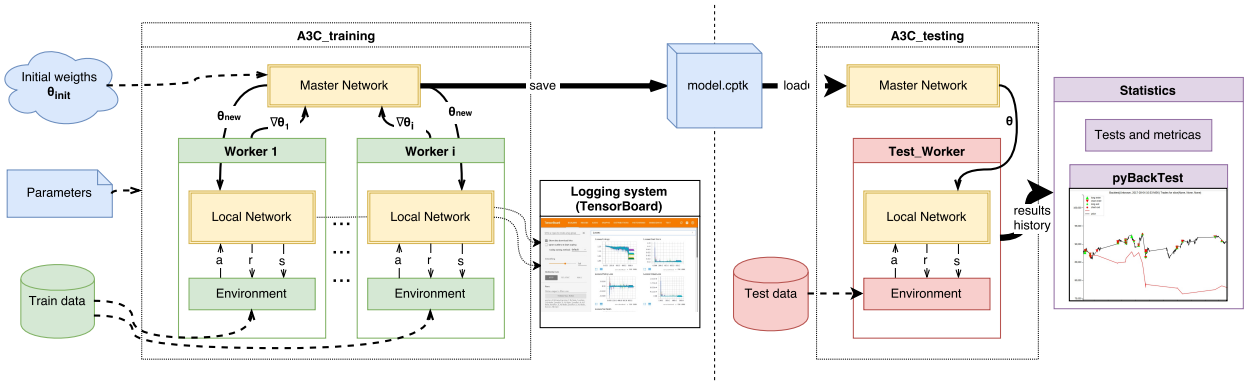

Рис. 3. Схема тренировочно-тестирующей системы

Обучение нейронной сети проводится каждые $N_{steps} = 200$ шагов. N_{worker} это номер параллельных процессов и в данной работе был зафиксирован и равен 10. Каждая эпоха обучения проводилась на трехмесячных данных, содержащих 50'000 одноминутных шагов. Сходимость при размере шага около 10^{-3} требует порядка 1000 эпох.

Для ускорения тестирования системы, была реализована тестирующая подсистема без обучения и записей в журнал. Вероятностный подход в тесте заменен функцией arg max. Функция dropout также отключена в тестировании.

4. ЧИСЛЕННЫЕ РЕЗУЛЬТАТЫ

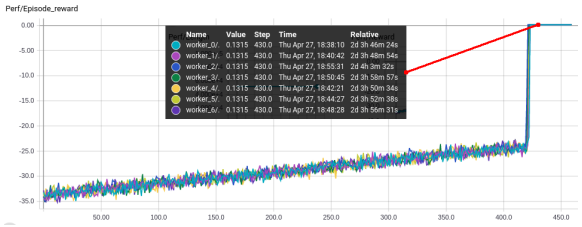

Рис. 4. Кивая обучения алгоритма, отмечено попадание в экстремум, соответствующий стретегии *не торговать*

Численные эксперименты производились на реальных исторических данных (полный лог обезличенных заявок для RTSI). В качестве обучающей выборки использовались заявки с 15.09.2015 до 15.12.2015. Тест производился на следующих 6 месяцах: с 15.12.2015 до 15.06.2016.

Комиссия для покупки или продажи фиксировалась как 1.25 руб. за операцию, т.е. 2.50 руб за каждую сделку. Очевидно, что стратегия должна быть более прибыльной, чем 2.50 руб/трейд. Для увеличения прибыльности каждой сделки была искусственно завышена комиссия в модели награды. Также функция награды была изменена для избежания попадания в ловушку "купить и держать"(см рис. 4), т.е. отсутствие активных торгов. Для этого был введен штраф за длительное повторение действия.

Важной задачей является разработка архитектуры нейронной сети. В качестве начальной точки была выбрана самая простая ИНС с одним общим скрытым слоем, линейной функцией V и линейным слоем с активацией softmax для выбора действия $\pi(s)$. В качестве гипотез, улучшающих качество метода были выдвинуты следующие предположения:

Предположение 1. *Использовать другую функцию награды;*

Предположение 2. *Добавить рекуррентный слой (LSTM);*

Предположение 3. *Добавить слой dropout;*

Предположение 4. *Увеличить количество нейронов в скрытых слоях;*

Предположение 5. *Использовать более сложную архитектуру функции ценности*

Предположение 6. *Объединить в один вектор признаки за несколько минут;*

Исходя из предположений, мы сконструировали несколько архитектур из комбинации одних и тех же слоев, но с разными параметрами, в т.ч. и с отсутствием слоя (см. таблицу 1 и рисунок 2):

Name	**Depth**	**Dense**	**Dropout**	**LSTM**	**Dense V**	**Dense A**
5	6	-	0.5	64	-	-
8	6	-	0.5	128	-	-
5coolV	6	-	0.5	64	32	-
9	1	-	0.5	64	32	-
12	1	-	0.5	64	32	32
5noLSTM	20	-	-	-	-	-
6	6	128	-	128	-	-

Таблица 1. Архитектуры

1. Depth - количество соединенных последовательно векторов признаков, используемых как входной вектор;
2. Dense - Этот атрибут - количество нейронов в полностью связанном первом слое (например, 128) или отсутствие такого слоя (-);
3. Dropout - Вероятность отсева (dropout) (например, 0,5) или отсутствие (-);
4. LSTM - Число нейронов в LSTM слое, связанном с первым слоем (например, 64) или отсутствием такого слоя (-);
5. Dense V - Число нейронов в полносвязном слое, предшествующем выходному линейному нейрону критика ($V(s)$);
6. Dense A - Число нейронов в полносвязном слое, предшествующем выходному softmax слою исполнителя ($\pi(a|s)$);

Для исследования зависимости от добавления слоя отсева (dropout), было выбрано две архитектуры под названиями 6 и 8. По результатам теста (см. таблицу 3) видно, что dropout существенно улучшает ситуацию. Для сравнения зависимости результата от количества нейронов в LSTM слое рассматривались архитектуры 8 и 5, зависимости от усложнения аппроксиматора функции ценности - 5 и 5$_cool_V$. Чтобы проверить зависимость от количество соединенных последовательно векторов признаков, используемых как входной вектор были рассмотрены архитектуры 5$_cool_V$ и 9 (кривые обучения и теста см. на рис. 5,6,7).

Основная сложность в экспериментальной оптимизации архитектуры - время обучения одной модели. Оно варьируется в зависимости от загруженности сервера, но в основном составляет десятки часов. Выбор скорости обучения также важен для оптимизации и сходимости алгоритма [17].

Ниже приведены таблицы с экономическими метриками, важными для принятия решения о инвестиционной привлекательности алгоритма, отражающие прибыльности и риск.

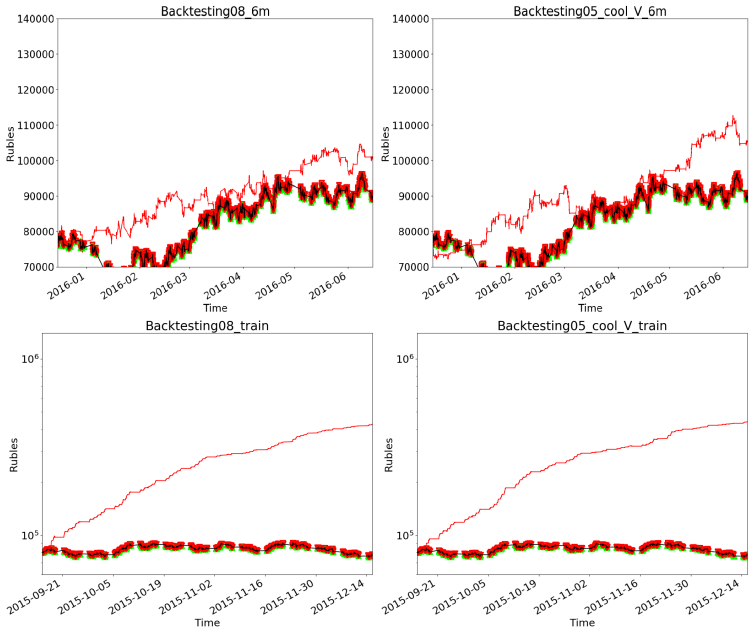

Рис. 5. Сравнение моделей "8"и "5 cool V": как влияет использование линейного приближения функции ценности состояния $V(s)$ (слева) против двуслойной сети с функцией активации *tanh* на первом слое. Остальные параметры идентичны (см. таблицу 1).

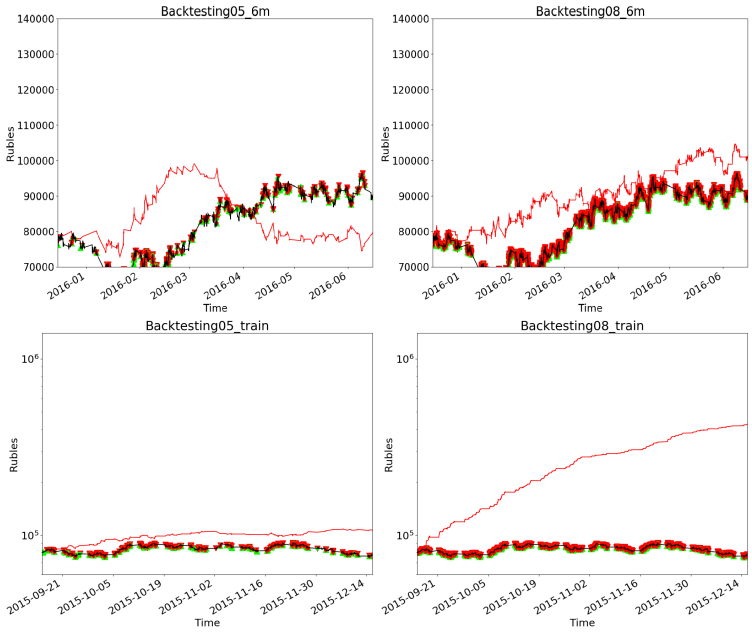

Рис. 6. Сравнение моделей "5"и "8": как влияет использование 64 нейронов в слое LSTM (слева) против 128 нейронов. Остальные параметры идентичны (см. таблицу 1).

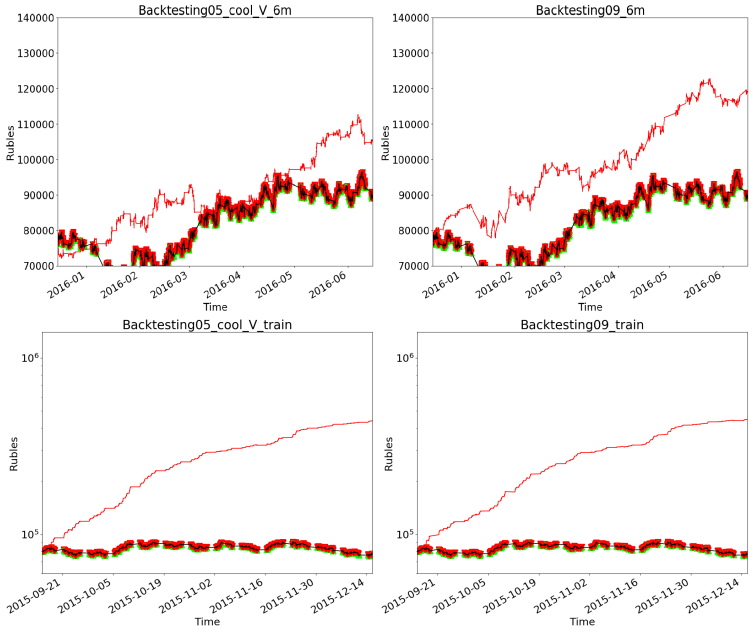

Рис. 7. Сравнение моделей "5 cool V"и "9": как влияет соединение признаков за 6 шагов в один входной вектор(слева) против использования только признаков за 1 шаг. Остальные параметры идентичны (см. таблицу 1).

1. Прибыль % годовых - это прибыльность в процентах годовых, т.е. $\frac{profit}{begin\_price} \cdot \frac{365}{number\_of\_days}$ где $number\_of\_days = 90$ для 3 месяцев и 180 для 6. $begin\_price$ - это стоимость фин. инструмента в начале торговли.
2. Прибыль % годовых (комисс.)- это прибыльность в процентах годовых с учетом комиссии, т.е. $\frac{profit - n\_trades \cdot fee}{begin\_price} \cdot \frac{365}{number\_of\_days}$ где $fee = 2.5rub.$
3. Коэф. Шарпа - это коэффициент Шарпа $\frac{\mathbb{E}(profit)}{\sigma(profit)}$ - отношение прибыльности к отклонениям в обе стороны.
4. Средняя сделка, руб. - средняя прибыль за сделку. Этот показатель крайне важен при учете комиссий.

Имя	**Прибыль % годовых**	**Прибыль % годовых (комисс.)**	**Коэф. Шарпа**	**Доля выигрышных сделок, %**	**Средняя сделка, руб.**
5	**99.0**	**94.9**	**3.23**	**60.32**	**59.59**
8	50.3	16.4	1.73	51.69	3.7
5coolV	44.6	-3.5	1.46	54.14	2.32
9	**86.1**	47.3	**2.18**	53.58	5.55
6	22.6	-121.6	0.58	51.14	0.39

Таблица 2. Результат работы на 3х месячных тестовых данных

5. ЗАКЛЮЧЕНИЕ

Главный результат работы - создание потенциально прибыльного и привлекательного для инвестиций с эконометрической точки зрения алгоритма биржевой торговли, основанного на методе Advantage Actor-Critic. Так, лучшая архитектура достигла показателей прибыльности в 110% годовых без учета комиссии или 66% годовых с учетом комиссии в 2.5 рубля за сделку по фьючерсу РТС (подсчитано на 6 месяцах исторических данных 2016 года).

Имя	Прибыль % годовых	Прибыль % годовых (комисс.)	Коэф. Шарпа	Доля выигрышных сделок, %	Средняя сделка, руб.
5	8.8	5.2	0.29	**55.82**	**6.15**
8	64.0	28.4	2.14	52.57	4.5
5coolV	75.6	20.7	**2.68**	53.24	3.44
9	**110.5**	**66.5**	**3.2**	**54.21**	**6.27**
6	8.6	-143.7	0.25	50.84	0.14

Таблица 3. Результат работы на 6 месячных тестовых данных

Имя	Прибыль % годовых	Прибыль % годовых (комисс.)	Коэф. Шарпа	Доля выигрышных сделок, %	Средняя сделка, руб.
5	10^{-3}	139.1	5.11	58.13	47.12
8	$5 \cdot 10^{-3}$	1784.5	5.11	68.91	70.28
5coolV	10^{-3}	1855.4	24.1	67.95	57.22
9	10^{-3}	1894.9	22.57	70.26	68.21
6	10^{-3}	7385.5	54.29	88.22	92.11

Таблица 4. Результат работы на 3 месячных тренировочных данных

В ходе оптимизации алгоритма были экспериментально проверены несколько гипотез, что позволило существенно улучшить характеристики метода и создать представление о применимости ряда идей:

- Использовать другую функцию награды -спорно. С одной стороны, позваляет избежать застревания в локальном минимуме отсутствия торговли, с другой - оптимизируется уже не целевая функция торговца;
- Объединить в один вектор признаки за несколько минут - неверно;
- Добавить рекуррентный слой (LSTM) - верно;
- Добавить слой dropout - верно;
- Увеличить количество нейронов в скрытых слоях - спорно;
- Использовать нейронную сеть в несколько слоев для аппроксимации *функции ценности* - верно;

В результате работы также была реализована удобная среда для дальнейших экспериментов с торговлей на бирже, выдержанная в принятом de-facto стиле. Мы верим, что это позволит проще экспериментировать с применением различных методов к данной задаче.
Дальнейшее развитие работы мы видим в оптимизации архитектуры и применении в реальной торговой системе.

СПИСОК ЛИТЕРАТУРЫ

1. Y. Deng, Y. Kong, F. Bao, and Q. Dai, Sparse Coding-Inspired Optimal Trading System for HFT Industry. IEEE Transactions on Industrial Informatics, vol. 11, no. 2, pp. 467–475, 2015.
2. Long Short-term Memory Schmidhuber, Sepp Hochreiter and Jürgen Neural Computation 8 vol.9, pp.1735–1780, 1997
3. Sutton, Richard S. and A. G. Barto, Reinforcement Learning: An Introduction. IEEE Transactions on Neural Networks Neural Networks, vol. 9, p. 1054, 1998.
4. W. F. Sharpe, The Sharpe Ratio. The Journal of Portfolio Management, vol. 21, no. 1, pp. 49–58, oct 1994.
5. Y. Li Deep Reinforcement Learning: An Overview. CoRR, vol. abs/1701.0, pp. 1–30, 2017.
6. J. Moody and M. Saffell Learning to trade via direct reinforcement IEEE Transactions on Neural Networks, vol. 12, no. 4, pp. 875–889, 2001.

7. Y. Deng, F. Bao, Y. Kong, Z. Ren, and Q. Dai Deep Direct Reinforcement Learning for Financial Signal Representation and Trading. IEEE Transactions on Neural Networks and Learning Systems, vol. 28, no. 3, pp. 653–664, 2017.

8. J. Moody, L. Wu, Y. Liao, and M. Saffell Performance functions and reinforcement learning for trading systems and portfolios. Journal of Forecasting, vol. 17, no. 56, pp. 441–470, 1998.

9. K. L. Xin Du, Jinjian Zhai Algorithm Trading using Q-Learning and Recurrent Reinforcement Learning, positions, 1, p.1., 2009.

10. R. J. Williams Simple Statistical Gradient-Following Algorithms for Connectionist Reinforcement Learning Machine Learning, vol. 8, pp. 229–256, 1992.

11. S. D. Bekiros Heterogeneous trading strategies with adaptive fuzzy Actor-Critic reinforcement learning: A behavioral approach Journal of Economic Dynamics and Control, vol. 34, no. 6, pp. 1153–1170, jun 2010.

12. V. Mnih, A. P. Badia, M. Mirza, A. Graves, T. P. Lillicrap, T. Harley, D. Silver, and K. Kavukcuoglu, "Asynchronous Methods for Deep Reinforcement Learning," 2016.

13. Y. Zhan, H. B. Ammar, and M. E. Taylor Theoretically-grounded policy advice from multiple teachers in reinforcement learning settings with applications to negative transfer IJCAI International Joint Conference on Artificial Intelligence, vol. 2016-Janua, no. 7540, pp. 2315–2321, 2016.

14. A. A. Markov The Theory of Algorithms Journal of Symbolic Logic, vol. 18, no. 4, pp. 340–341, 1953.

15. R. Bellman A Markovian decision process Journal Of Mathematics And Mechanics, vol. 6, no. 4, pp. 679–684, 1957.

16. M. Riedmiller and H. Braun A direct adaptive method for faster backpropagation learning: The RPROP algorithm IEEE International Conference on Neural Networks - Conference Proceedings, vol. 1993-Janua, pp. 586–591, 1993.

17. G. E. Hinton, N. Srivastava, and K. Swersky RMSProp: Lecture 6a- overview of mini-batch gradient descent COURSERA: Neural Networks for Machine Learning, p. 31, 2012.

18. M. Abadi, P. Barham, J. Chen, Z. Chen, A. Davis, J. Dean, M. Devin, S. Ghemawat, G. Irving, M. Isard, M. Kudlur, J. Levenberg, R. Monga, S. Moore, D. G. Murray, B. Steiner, P. Tucker, V. Vasudevan, P. Warden, M. Wicke, Y. Yu, X. Zheng, and G. Brain TensorFlow: A System for Large-Scale Machine Learning TensorFlow: A system for large-scale machine learning 12th USENIX Symposium on Operating Systems Design and Implementation (OSDI '16), 2016, pp. 265–284.

19. W. Zaremba, I. Sutskever, and O. Vinyals Recurrent Neural Network Regularization International Conference on Learning Representations, no. 2013, pp. 1–8, 2014.

Algorithmic trading with reinforcement learning

E.S. Ponomarev, E.V. Oseledets, A.S. Cichocki

The development of reinforced learning methods has opened wide boundaries on their use in many areas, including algorithmic trading. In this paper trading on the stock exchange is interpreted in terms of a game with a Markov property consisting of states, actions, and rewards.

We propose and experimentally tested a system for trading a fixed volume of a financial instrument based on the asynchronous advantage actor-critic method using a number of neural network architectures. The use of recurrent layers in this approach is investigated. We use real anonymized data for experiments.

The best architecture demonstrated a trading strategy for the RTS Index futures (MOEX: RTSI) with a profitability of 66% per annum, taking into account the commission. Project source code is available via the link: http://github.com/evgps/a3c_trading

KEYWORDS: Algorithmic trading, reinforcement learning, neural networks, recurrent neural networks.