ИПМ им.М.В.Келдыша РАН • Электронная библиотека

Препринты ИПМ • Препринт № 170 за 2018 г.

ISSN 2071-2898 (Print)
ISSN 2071-2901 (Online)

Князятов С.А., Малинецкий Г.Г.

Решение задачи распознавания блефа в игре «верю – не верю» с помощью алгоритмов обучения с подкреплением

Рекомендуемая форма библиографической ссылки: Князятов С.А., Малинецкий Г.Г. Решение задачи распознавания блефа в игре «верю – не верю» с помощью алгоритмов обучения с подкреплением // Препринты ИПМ им. М.В.Келдыша. 2018. № 170. 21 с. doi:10.20948/prepr-2018-170
URL: http://library.keldysh.ru/preprint.asp?id=2018-170

О р д е н а Л е н и н а
ИНСТИТУТ ПРИКЛАДНОЙ МАТЕМАТИКИ
имени М.В.Келдыша
Р о с с и й с к о й а к а д е м и и н а у к

С. А. Князятов, Г. Г. Малинецкий

Решение задачи распознавания блефа в игре «верю – не верю» с помощью алгоритмов обучения с подкреплением

Москва — 2018

Князятов С.А., Малинецкий Г.Г.

Решение задачи распознавания блефа в игре «верю – не верю» с помощью алгоритмов обучения с подкреплением

В работе исследуется возможность построения алгоритма на основе обучения с подкреплением для задачи распознавания и использования блефа в карточной игре «верю — не верю». Построенный алгоритм обладает «интеллектуальной способностью» перестраивать свою стратегию поведения и оценивать возможные ходы, основываясь на предыдущем опыте. Данный класс алгоритмов используется для принятия решений в быстроменяющихся средах. Описаны способ и результаты сравнения алгоритмов между собой, результаты игр лучших алгоритмов с реальным соперником. Обнаружен эффект «переобучения» — увеличение количества обучающий партий в ряде случаев не улучшает, а ухудшает качество работы алгоритма.

Ключевые слова: обучение с подкреплением, математическое моделирование, Q-обучение, метод SARSA(λ), алгоритм распознавания блефа, имитация блефа, нейронные сети, высокоскоростное принятие решений

Knyazyatov S.A., Malinetskiy G.G.

The solution of the problem of bluff detection in the game «I-doubt-it» based on reinforcement learning

In this paper we consider the construction of an algorithm based on reinforcement learning for the problem of recognizing and using a bluff on the example of a card game «I-doubt-it». The constructed algorithm has the "intellectual ability" to restructure its behavior strategy and to evaluate possible moves based on previous experience. This class of algorithms used to make decisions in rapidly changing environments. The method and results of comparing algorithms among themselves, the results of games of the best algorithms with a real opponent are obtained. The effect of "overfitting" is detected, increasing the number of training batches, in some cases, does not improve, but worsens the quality of the algorithm.

Key words: reinforcement learning, mathematical modeling, Q-learning, SARSA(λ) method, bluff detection algorithm, bluff imitation, neural networks, high-speed decision making

Работа выполнена при поддержке РФФИ проект 16-01-00342

Введение

Есть задачи, которые можно эффективно решать с помощью методов обучения с подкреплением. Обучение с подкреплением представляет собой вычислительный подход к пониманию и реализации целенаправленного обучения и принятия решений. Данные методы помогают построить стратегию поведения системы, определяющую её взаимодействие с окружающей средой для получения некоторых бонусов, которые в обучении с подкреплением называются наградами. Поиск стратегии для максимизации награды можно отнести к задаче управления, но методы обучения с подкреплением имеют ряд преимуществ над стандартными методами управления. Например, в случае стохастической среды или среды с большим количеством состояний методам обучения с подкреплением не нужно полное описание среды для поиска оптимальной стратегии. Обучение с подкреплением использует формальную основу, которая определяет взаимодействие между обучающимся агентом и средой в терминах состояний, действий и вознаграждений [1].

Игра «верю — не верю» и все её модификации — это простейшая игра с неполной информацией, ориентированная на использование блефа. Она интересна с точки зрения моделирования тем, что является простейшей моделью блефа и позволяет тестировать новые подходы и алгоритмы. Важной особенностью данной задачи является то, что она легко обобщается и результаты, полученные для базовой задачи, работают и в обобщенной.

Рассматриваемые в данной работе методы основываются на классических подходах к построению самообучающихся алгоритмов, описанных в работах М.Цейтлина и И.Гельфанда. Первоначально данный класс рассматривался как задачи теории игр. В работе [7] представлена классификация таких игр и используемых для них автоматов, рассмотрен пример игры двух автоматов. Этот подход был развит в работе [8]. В ней вводятся такие понятия, как самонастраиваемые системы и предлагаются принципы их настройки, рассматривается понятие скорости адаптации и подчеркивается её важность в данном классе задач. Проблема моделирования сложных систем и построения управляющих воздействий для них описаны в работе [9]. Фундаментальной работой по построению линейных автоматов является работа [10], в ней вводятся фундаментальные подходы к рассмотрению среды, способу взаимодействия с ней в случае, когда нет априорной информацией о численных значениях параметров среды. В эссе [11]

описаны построение самоорганизующихся сред с использованием «простых» агентов и способы моделирования природы с помощью агентного моделирования. Главным отличием методов обучения с подкреплением от классической теории линейных автоматов является предмет моделирования. В случае линейных автоматов главный предмет анализа — это архитектура агента и способы его перехода из одного состояния в другое. В методах обучения с подкреплением внимание фокусируется на обработке информации об опыте взаимодействия агента со средой.

В теории обучения с подкреплением имеют место несколько подходов к обработке информации об опыте взаимодействия алгоритма со средой. В данной работе будут рассматриваться направления, связанные с семейством методов SARSA(λ) и методом Q-обучения.

Для обобщения полученного опыта взаимодействия со средой мы будем использовать нейронные сети. По предыдущим примерам взаимодействия со средой нейронная сеть обучается так же, как в задачах обучения с подкреплением. В работе будет исследована возможность использования решающих правил вместо нейронной сети, будет проведено их сравнение.

В настоящее время в виртуальном пространстве для информационного управления общественным сознанием широко используются боты, дающие как достоверную, так и ложную информацию. Это делает задачу автоматизированного распознавания блефа особенно актуальной.

Структура препринта следующая. В первом разделе описаны правила игры «верю — не верю», во втором — формализация задачи и описание её в терминах среды и взаимодействия со средой, в третьем разделе описаны используемые методы и их предпосылки, в четвертом разделе описана целесообразность аппроксимации функции ценности состояний-действий с помощью нейронной сети. В последующих разделах описывается процесс моделирования и способы повышения качества работы алгоритмов. В моделировании использовался эволюционных подход к улучшению стратегии и архитектуры алгоритма. В начале работы при описании среды и действий использовался такой же подход, если бы среду и действия в них нужно было бы описать человеку; как показано далее, у такого подхода узкая область применения, и качество построенного алгоритма оставляет желать лучшего. Далее будет проведен анализ метода обучения, будут выделены способы повышения качества алгоритма, после использования которых будет достигнуто ожидаемое качество. Критерием верификации работы алгоритмов являлись партии с людьми, финальная оценка считалась следующим образом. Количество очков у игрока выбиралось как медиана множества количества

очков игроков во всех партиях с данным алгоритмом. Далее из множества удалялись все партии с отличным количеством очков у игроков от медианного значения. Количество очков у алгоритма равняется максимуму среди оставшихся в множестве очков алгоритма.

Постановка задачи

Рассмотрим простую карточную игру для двух игроков. Колода (36 карт) раздается двум игрокам. Далее каждый по очереди кладет карту рубашкой вверх и заявляет, карту какого достоинства он положил, например, «десятки». Этим он утверждает, что карта, которую он положил является «десяткой». Игрок может сказать: «Верю». И тогда он, в свою очередь, может положить, объявив, свою карту поверх той, которую положил соперник. Тем самым он утверждает, что положенная карта имеет то же достоинство, то есть в данном случае «десятка». Если он говорит «не верю», то последняя положенная карта вскрывается. Если она оказывается такой, как объявил игрок, то он забирает все выложенные карты, иначе все выложенные карты забирает соперник. Проигравшим становится тот, кто соберет у себя все карты.

Оригинальная игра «верю – не верю» и все её модификации рассчитаны на трех и более игроков, поэтому следует ввести некоторые правила, отсутствующие в исходной постановке задачи. Первая особенность, которая возникает при игре двух человек, — это то, что по картам из своей руки можно узнать карты противника, что в случае 36 различных карт делает игру бессмысленной. Поэтому будем считать, что в игре используется стандартная игровая колода карт. Вторая особенность, которая возникает при игре двух человек, это то, что при наличии у одного из игроков четырех карт одного наименования, например четырех девяток разной масти, у него появляется преимущество, т.к. он может наверняка знать, что этих карт нет у соперника. Поэтому предлагается штрафовать игроков одним очком штрафа за сбор комбинации из четырех одинаковых карт и отправлять эти карты в сброс. Следовательно, стартовую раздачу стоит организовать таким образом, чтобы количество карт разных наименований у игроков совпадало. Стоит отметить, что в случае, когда в игре остается всего 4 карты, игру нужно останавливать, т.к. это патовая ситуация, и после этого следует объявлять ничью. Игра останавливается, когда один из игроков набирает 5 и более штрафных очков, именно он считается проигравшим. Если у одного из игроков заканчиваются в руках карты, карты со стола пересдаются.

Формализация задачи

Для того чтобы использовать те или иные алгоритмы обучения с подкреплением, нам нужно формализовать игру. Выберем одного игрока, будем называть его агентом, второй игрок и крупье будут выполнять роль среды. Агент и окружающая среда взаимодействуют на каждом из дискретных временных шагов (ходы игрока), составляющих последовательность, $t = 0,1,2,\ldots T$, где T — конец игры или терминальное состояние. На каждом временном шаге t агент получает некоторое описание состояния окружающей среды $s_t \in S$, где S — множество возможных состояний среды, и на основании этого описания выбирает действие, где $A(s_t)$ — множество действий, возможных в состоянии s_t.

В данной задаче состояние s_t представляет собой вектор размерности 11. Первые 9 координат принимают значения от 0 до 4, каждая позиция связана с одним наименованием карт, число в данной позиции обозначает количество карт этого наименования в руке у игрока. В десятой координате записано количество карт в руке соперника. В последней координате записана категория карты, которую озвучил игрок, начавший данный раунд. Последняя координата принимает значения от -1 до 8, где -1 она равняется в случае, если карта еще не озвучена, в остальных случаях она соответствует наименованию озвученной карты.

Формат действий в данной задаче зависит от состояния игрока. Если игрок находится в состоянии озвучивания карты первого хода, то действие — это скаляр, и оно принимает значение от 0 до 8 и зависит от тех карт, что есть у игрока. В случае если это не озвучивание первой карты для хода, то действие — это вектор размерности 10, каждая координата вектора может принимать значения 0 и 1. Первые 9 координат вектора действий соотносятся с ходом картой соответствующего наименования, последняя координата — с проверкой последней положенной карты.

Введем функцию $Q(s, a)$, которая парам состояние-действие соотносит число. Данное число называется ценностью состояния-действия. Также на каждом временном шаге t агент получает вознаграждение r_t. В данной задаче вознаграждение выдается после каждого сделанного хода. Если первый и второй игрок не проверяли последнюю карту, то первому игроку (тому, кто ходил первым) выдается вознаграждение, равное нулю. В случае если одним из игроков была проверена верхняя карта, то при распознавании блефа ему выдается награда в виде количества штрафных очков у соперника, а сопернику

выдается отрицательная награда того же номинала. В случае ошибочной проверки знаки наград меняются между игроками.

Описание используемых методов

Перед тем как описывать способы определения ценностей для пар состояний-действий, стоит указать используемые стратегии выбора действий. В данной задаче есть два принципиально разных класса состояний, следовательно, и стратегии для них тоже должны быть разные. В случае, когда производится выбор карты для озвучивания, карта выбирается следующим образом. В руке находятся наименования с наибольшим количеством карт, далее из них случайно выбирается одно наименование. Во втором случае используется -жадная стратегия, она заключается в жадном выборе действия (действие, которое максимизирует $Q(s,a)$) с вероятностью $(1-\epsilon)$, в остальных случаях действие выбирается случайно.

Все используемые в работе методы построения оценки функции ценности пар состояний-действий основаны на методе временных различий (TD — Temporal-Difference). В TD-методах процесс обучения основывается на опыте взаимодействия агента со средой без использования модели среды. Расчетные оценки состояний (в случае задачи управления состояний-действий) в TD-методах обновляются, основываясь на других полученных оценках, т.е. они самонастраиваются [2]. Классический TD-метод используют для построения оценок ценности состояния среды. Опишем его, перед тем как перейти к случаю управления.

В данной работе будут использоваться идеи многошагового TD-метода, так же известного как метод TD(λ), и одношагового метода, или метода TD(0), который является частным случаем многошагового. В многошаговом методе имеется переменная памяти $e(s)$, соответствующая каждому состоянию. Она называется следом приемлемости [2]. На каждом временном шаге следы приемлемости для всех состояний, кроме текущего, убывают с коэффициентом $\lambda \cdot \gamma$, а след приемлемости для посещаемого на данном шаге состояния увеличивается на 1, где λ — параметр затухания следа, γ — коэффициент приведения. След приемлемости все время регистрирует, посещение каких состояний имело место недавно, где смысл понятия "недавно" определяется с помощью коэффициента$\lambda \cdot \gamma$. Процесс оценки состояний проходит следующим образом. Во время обучения при переходе из состояний s_t в состояние s_{t+1}вычисляется величина δ

$$\delta = r + \gamma V(s_{t+1}) - V(s_t), \quad (1)$$

где $V(s_t)$ — функция ценности состояния, аналогичная функции ценности пар состояний-действий $Q(s,a)$. Далее для всех состояний производится корректировка их ценности с использованием следов приемлемости

$$V(s) := V(s) + \alpha\delta e(s), \quad (2)$$

где α — коэффициент обучения. Соответственно, в случае одношагового метода никаких следов приемлемости нет, т.к. $\lambda = 0$, поэтому на каждом шаге производится только корректировка ценности состояния s_t, что можно записать в виде

$$V(s) := V(s) + \alpha\big(r + \gamma V(s_{t+1}) - V(s_t)\big). \quad (3)$$

Одним из наиболее важных достижений в обучении с подкреплением стало развитие управления по TD-методу с разделенной оценкой ценности стратегий, известного как Q-обучение. В данной работе используется простейший одношаговый алгоритм корректировки ценностей пар состояние-действие, который основывается на одношаговом методе TD

$$Q(s,a) := Q(s,a) + \alpha\big(r + \gamma max_{a'} Q(s',a') - Q(s,a)\big), \quad (4)$$

с штрихами здесь состояния и действия s_{t+1} и a_{t+1}, без штрихов s_t и a_t. В этом случае искомая функция ценности действия Q непосредственно аппроксимирует оптимальную функцию ценности действий, независимо от применяющейся стратегии. [3].

Альтернативой методам Q-обучения является метод SARSA (State-Action-Reward-State-Action), который основывается на модели обобщенной итерации по стратегиям с использованием TD-метода в оценочной или предсказательной части. В данной работе используется TD-метод управления с интегрированной оценкой ценности стратегий. Последовательность действий в методе SARSA(λ)

базируется на двух шагах. Первый шаг заключается в изучении функции ценности действий. Для этого необходимо оценить функцию $Q(s,a)$ для состояния s и всех действий a. Далее выбирается действие a и производится переход в следующее состояние. Второй шаг повторяет первый, только в конце шага вместо перехода производится корректировка ценностей всех пар состояний-действий [4]. По аналогии с методом TD(λ) находится величина δ

$$\delta = r + \gamma Q(s_{t+1}, a_{t+1}) - Q(s_t, a_t), \quad (5)$$

а далее для всех пар состояний-действий производится корректировка оценок

$$Q(s_t, a_t) := Q(s_t, a_t) + \alpha\delta e(s_t, a_t) \quad (6)$$

и корректировка всех следов приемлемости.

Численная реализация

В классической реализации методов обучения с подкреплением опыт взаимодействия со средой хранят в виде таблицы, у которой напротив каждого состояния-действия записана его ценность. Состояния-действия, которые агент еще не использовал, инициализируются в зависимости от задачи, чаще всего нулями. В задачах с большим количеством пар состояний-действий или в задачах с непрерывными коэффициентами такую таблицу составить проблематично или вовсе невозможно, поэтому ключевым вопросом является вопрос обобщения. Для его решения исходную функцию Q аппроксимируют с помощью нейронной сети прямого распространения. Это позволяет в случае наличия истории взаимодействия построить оценки для тех состояний-действий, которые агентом еще не посещались [5]. Задача построения аппроксимирующей функции по набору значений неизвестной функции является задачей обучения с учителем. В случае использования нейронной сети задача имеет следующую формулировку. Требуется подобрать такой набор весов в нейронной сети, при котором на некотором множестве примеров величина отклонения от известных значений была минимальна. Формальная постановка представлена ниже

$$\sum_{i=0}^{K} \sum_{a \in A(s_i)} \| Q'(s_i, a, w) - r_i \| \to \min_w, \quad (7)$$

где Q'(s, a, w) — аппроксимация функции ценности состояния-действия, w — вектор весов нейронной сети, K — количество примеров.

В данной работе использовалась нейронная сеть прямого распространения с двумя скрытыми слоями, в первом скрытом слое использовалось 160 нейронов, во втором скрытом слое — 40. Функции активации нейронов — гиперболический тангенс. Во время обучения для разных примеров на первом слое игнорировались 10% нейронов, а во втором 15%, это производилось для того, чтобы избежать эффекта переобучения [6]. Данные параметры подбирались следующим образом. С помощью одного из алгоритмов обучения с подкреплением была набрана выборка состояний-действий, состоящая из 500 элементов. По 450 элементам данной выборки были обучены нейронные сети с различными параметрами. Далее на вход обученным нейросетям подавались оставшиеся 50 примеров, и для них вычислялась средняя ошибка в евклидовой метрике. У вышеперечисленных параметров была наименьшая ошибка. Используемая нейронная сеть схематично изображена на рис. 1.

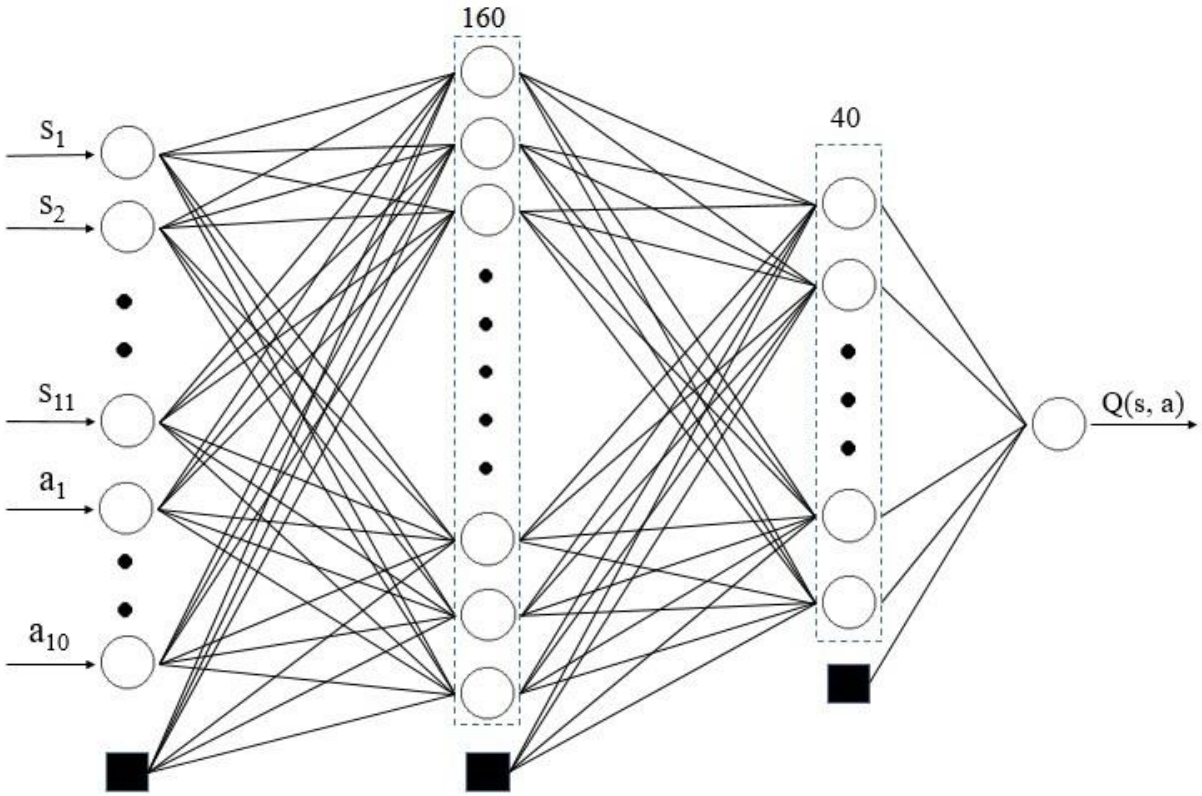

Рис.1 Схематичное изображение архитектуры используемой нейронной сети

В используемых методах достаточно много настраиваемых коэффициентов, влияющих на результат работы алгоритма. Помимо настройки коэффициентов используемого метода, можно предобучать нейронную сеть. Предобучение будем производить с помощью игр с агентом, выбирающим действия случайно, и, после нескольких игр, с предыдущими версиями алгоритма. Количество партий для предобучения будет параметром задачи. Как описывалось выше, для выбора действий будет использоваться-жадный алгоритм. Параметр жадности также будет настраиваемым параметром.

Для того чтобы проследить связь качества алгоритмов в зависимости от используемых параметров, следует полностью перебрать все параметры с некоторыми шагами и сравнивать их между собой. Это большая вычислительная задача, решение которой можно получить более экономичным способом. Отбор алгоритмов будет произведен в два этапа.

На первом этапе отбора была составлена сетка параметров для каждого типа алгоритмов, множества значений каждого из параметров представлены ниже.

- параметр λ использовался только для алгоритмов SARSA(λ), он принимал следующие значения: 0.0, 0.2, 0.4, 0.75, 1.0;
- параметр γ принимал значения: 0.7, 0.8, 0.9;
- параметр α принимал значения: 0.1, 0.2;
- параметр ε принимал значения: 0.1, 0.3, 0.5;
- частота дообучения нейронной сети принимала следующие значения: 20, 40, 60, 100;
- для предобучения использовались: 0, 15, и 30 игр.

Согласно данному списку были сформированы 1296 различных векторов параметров, на основе которых были построены соответствующие алгоритмы. Полученные алгоритмы проверялись при игре со случайной стратегией, из них были отобраны те алгоритмы, которые «уверенно» выигрывали у этой стратегии. Под «уверенной» победой здесь понимается победа минимум в 6 партиях из 7. После данной операции осталось 50 алгоритмов. Это количество все еще слишком велико для того, чтобы проводить ручную проверку качества построенных алгоритмов.

На втором этапе отбора производилось сокращение количества алгоритмов с помощью турнирной системы с выбыванием после двух поражений. Вначале алгоритмы были разбиты на 10 групп по 5 алгоритмов, в каждой группе производились игры каждый с каждым. За победу начислялось 3 очка, за ничью 1, за проигрыш со счетом 1:6 и 0:7 начислялось -1 очко. В каждой группе в следующий этап переходили 2 алгоритма с наибольшим количеством очков. Полученные в конце вышеописанной процедуры алгоритмы будем считать лучшими с точки зрения такого отбора.

Таким способом были выявлены 8 лучших алгоритмов. Для ранжирования оставшихся алгоритмов были проведены игры каждый с каждым. Параметры алгоритмов и количество набранных очков представлены в таблице 1.

Таблица 1. Результат сравнения различных стратегий

Алгоритм	λ	γ	α	ε	Частота дообучения нейронной сети	Количество игр для предобучения	Очки
Q-обучение	-	0.9	0.1	0.3	100	15	16
SARSA(λ)	1.00	0.8	0.2	0.5	60	15	16
Q-обучение	-	0.9	0.1	0.5	60	15	13
SARSA(λ)	0.00	0.8	0.2	0.5	100	15	13
Q-обучение	-	0.8	0.2	0.5	40	0	11
SARSA(λ)	0.75	0.7	0.1	0.5	100	30	6
SARSA(λ)	0.4	0.7	0.1	0.5	100	30	3
SARSA(λ)	0.75	0.9	0.2	0.5	20	15	0

Результаты игр с людьми

Для проверки качества были сыграны партии с тремя наилучшими алгоритмами.

При игре с первым алгоритмом игроку удалось победить в 6 партиях из 7. Данный алгоритм показывал хорошую адаптивность к предыдущим стратегиям. Придерживаясь одних и тех же действий, выигрывать не удавалось. Алгоритм показывал разнообразие стратегий в начале партии, но к концу партии он использовал 1-2 стратегии, узнав о которых, игроку удавалось выиграть.

При игре со вторым алгоритмом игроку удалось победить в 4 партиях из 7. Алгоритм показывал хорошую, но медленную адаптивность к предыдущим стратегиям. После 4 выигрышей дальше выигрывать не удавалось.

При игре с третьим алгоритмом игроку удалось победить во всех партиях. Алгоритм реализовал стратегию набора максимально возможного количества карт для укрепления своего стратегического преимущества, но реализовать это преимущество ему не удалось.

Ни один из алгоритмов не смог обыграть человека, из этого можно сделать вывод, что либо способ отбора лучших алгоритмов неприменим в данной задаче, либо параметры моделей требуют настройки, либо требуется изменить награды. Начнем с изменения наград, обучим лучшие алгоритмы с измененными наградами. В первом случае увеличим награду за победу до 100 очков. Во втором случае уменьшим награду за поражения до -100. Из-за несимметричных наград алгоритмы получались переобученными, они либо проверяли каждую карту, либо избегали проверок.

Повышение качества алгоритмов

Повышение качества алгоритмов производилось с помощью двух методов. Первый метод — изменение процедуры предобучения. Во время предобучения алгоритм проводит партии не со случайной стратегией, а с алгоритмами, использующими элементарные стратегии, такие агенты используют одинаковую стратегию на протяжении всей партии. Далее, для краткости, будем называть их элементарными стратегиями. Стратегии можно описать последовательностью действий, например FTC, что означает сыграть неправильную карту в первом ходу (F — False), сыграть правильную во втором (T — True) и проверить в третьем (C — Check). Лучшие алгоритмы сравнивали со следующими стратегиями: FC, TC, FTC, TFC, FFC, TTC, FFFC, FFTC, FTFC, TFFC, FTTC, TFTC, TTFC, TTTC. Количество очков у каждого из алгоритмов представлено в таблице 2.

Второй метод — сокращение размерности векторов состояний и действий. В большинстве задач машинного обучения сокращения размерности «входов» алгоритма ведет либо к повышению качества, либо к повышению скорости сходимости, поэтому этот прием следует применить и здесь. В данной задаче от описанного выше вектора состояния можно перейти к вектору размерности 5. На первых трех позициях следует записать количество наименований карт с одной, двумя, тремя картами в руке у агента. В четвертой позиции записано количество карт «правдивого» наименования. В пятой позиции — количество карт у оппонента. Все действия в данной игре можно свести к 3, тогда к 3 сводится и размерность вектора действий. Результаты игр алгоритмов после преобразования пар состояний-действий с элементарными стратегиями представлены в таблице 2.

Таблица 2. Результат партий с элементарными стратегиями до и после преобразования пар состояний-действий

Алгоритм	λ	γ	α	ε	Частота дообучения нейронной сети	Количество игр для предобучения	Очки	Очки (после преобр.)
Q-обучение	-	0.9	0.1	0.3	100	15	-12	15
SARSA(λ)	1.00	0.8	0.2	0.5	60	15	-13	7
Q-обучение	-	0.9	0.1	0.5	60	15	-14	10
SARSA(λ)	0.00	0.8	0.2	0.5	100	15	-9	-1
Q-обучение	-	0.8	0.2	0.5	40	0	-14	7
SARSA(λ)	0.75	0.7	0.1	0.5	100	30	-12	8
SARSA(λ)	0.4	0.7	0.1	0.5	100	30	-11	-3
SARSA(λ)	0.75	0.9	0.2	0.5	20	15	-12	2

Скомбинируем оба метода повышения качества алгоритмов и опишем общий способ обучения. Для каждого из алгоритмов перейдем в пространство меньшей размерности, выберем некоторый порог p и количество итераций I. На каждой итерации обучающийся алгоритм будет проводить партии против

элементарной стратегии, пока его частота выигрышей не превысит *p*. На каждой итерации элементарная стратегия меняется. Каждые 10 партий алгоритмы играют партии со стратегиями: FC, TC, FTC, TFC, FFC, TTC. Среднее и медиана частот выигрышей обучающегося алгоритма с данными стратегиями являются показателями качества обучения данного алгоритма. Длины цепочек были выбраны после анализа игр против людей, люди не использовали стратегии больше 2-3 ходов.

Для того чтобы компьютерный расчет сходился быстрее, использовались некоторые эвристики. Пока обучающийся алгоритм не сыграет 5 партий, его частота выигрышей не рассчитывалась. Для того чтобы уменьшить время расчета, был поставлен искусственный порог в виде 250 партий, после которых при любой частоте выигрышей алгоритм переходил к следующей стратегии.

Процессы сходимости трех наилучших алгоритмов из таблицы 2 изображены на рис. 2-4. Для сравнения результатов алгоритмов была получена одна величина качества, для этого был составлен вектор из медианы и среднего и подсчитана его евклидова норма, деленная на корень из двух. Здесь и далее, для сокращения, частота дообучения будет обозначена lf.

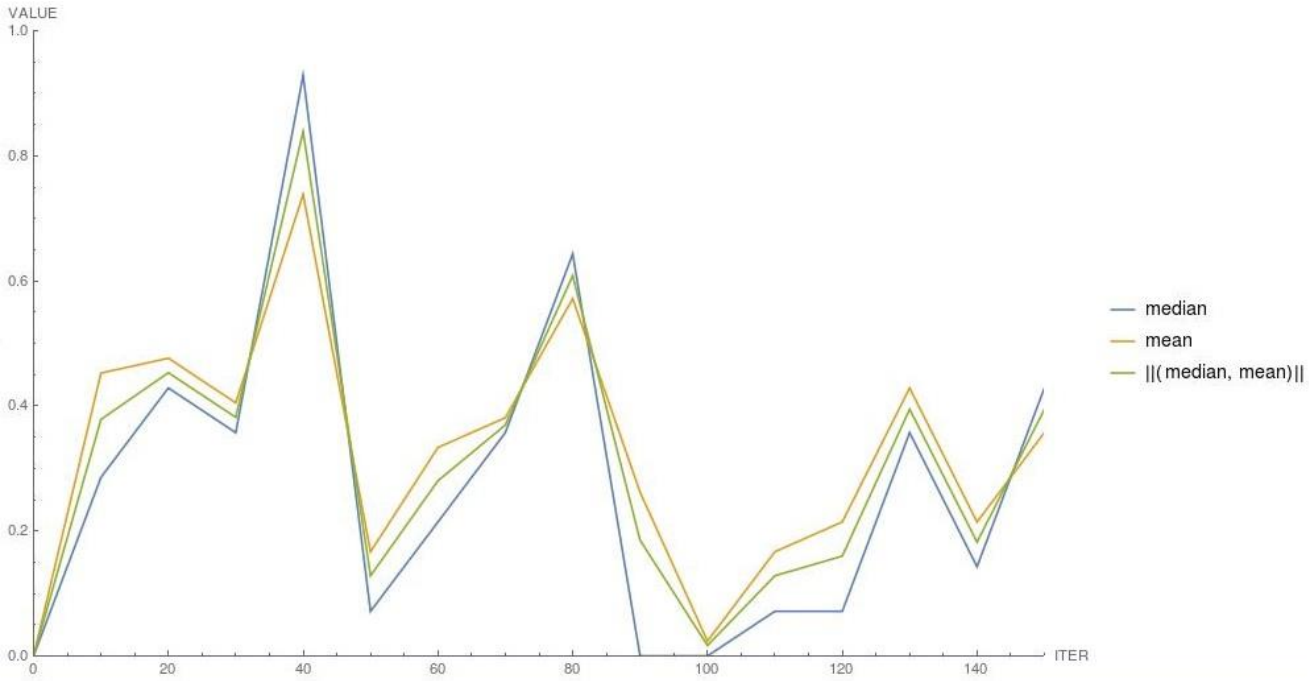

Рис. 2. Сходимость метода Q-обучение с параметрами: $\gamma = 0.9$, $\alpha = 0.1$, lf = 100, $\varepsilon = 0.3$, I = 150, p = 0.25

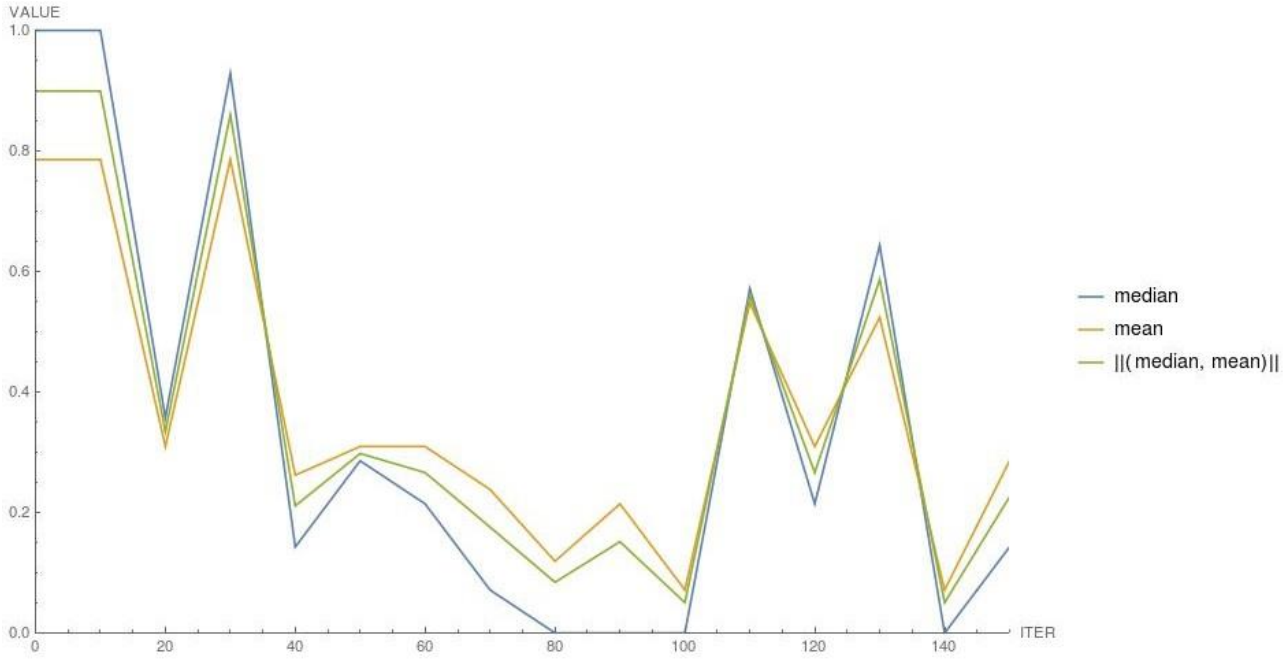

Рис. 3. Сходимость метода SARSA(λ) с параметрами: $\lambda = 1$, $\gamma = 0.8$, $\alpha = 0.2$, lf = 60, $\varepsilon = 0.5$, I = 150, p = 0.75

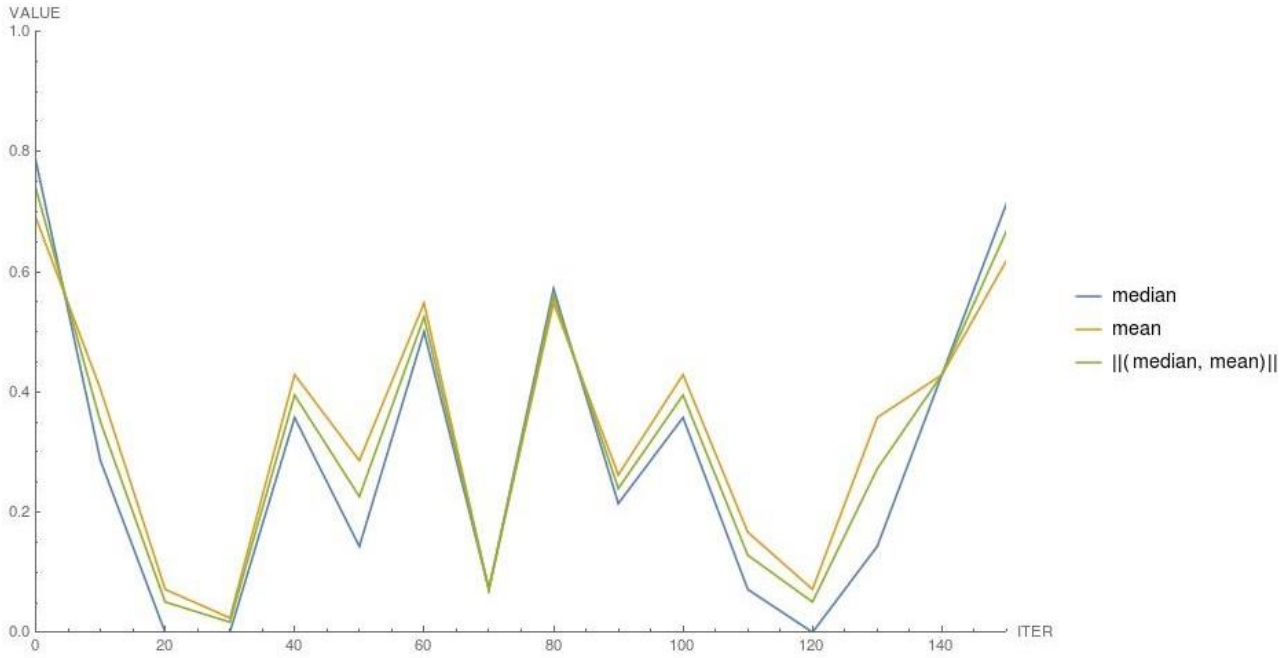

Рис. 4. Сходимость метода Q-обучение с параметрами: $\gamma = 0.9$, $\alpha = 0.1$, lf = 60, $\varepsilon = 0.5$, I = 150, p = 0.25

На рисунках выше видно, что ни о какой сходимости к одной оптимальной стратегии не может быть и речи. Более того, можно сделать вывод

о том, что в данной задаче не существует одной оптимальной стратегии, которая доминировала бы над всеми остальными. Видно, что, адаптируясь к одному классу стратегий, агент начинал проигрывать остальным. Данный результат может показаться парадоксальным, но в дисциплине, связанной с моделированием агентов, он встречается довольно часто. В классической работе по линейным автоматам данная проблема возникала из-за малой глубины автомата [10], в данном случае это та же самая проблема. Аналогично живым существам, у агента имеется ограниченная память, и чем выше скорость адаптации агента к новым импульсам от среды, тем быстрее забывается предыдущий опыт взаимодействия, что приводит к подобным эффектам.

Результаты игр улучшенных алгоритмов с людьми

Как и раньше, для проверки качества были сыграны партии с тремя наилучшими алгоритмами после их обучения с помощью партий с элементарными стратегиями. Для партий были выбраны оценки функции ценности состояний-действий, у которых были наибольшие значения медианы и среднего.

При игре с первым и третьем алгоритмами игроку удалось победить в 5 партиях из 7. Данные алгоритмы были схожи в своей стратегии игры. Они быстро адаптировались к выбранной стратегии и наращивали и реализовывали преимущество по картам. Из-за того что алгоритмы изменяли только последние состояния при получении наград, они не достаточно быстро адаптировались к используемым игроком стратегиям. Так же как и в предыдущем случае, агенты, корректирующие ценности только последних состояний-действий, неконкурентоспособны.

При игре со вторым алгоритмом игроку удалось победить в 2 партиях из 7. Данный алгоритм, как и предыдущая его версия, показывал хорошую адаптивность к предыдущим стратегиям игрока. Отличием от предыдущей версии является скорость сходимости к оптимальной, с точки зрения текущего противника, стратегии. После двух выигрышей игроку не удавалось выиграть у алгоритма.

Получение интерпретируемой стратегии

В этом разделе будет исследована возможность построения интерпретируемой стратегии на основе базы данных полученных оценок

состояний-действий с помощью нейронной сети. При построении стратегии поведения для системы, в которой окончательное решение принимает человек, очень важно использовать интерпретируемые методы. Данный класс методов позволяет выполнить сразу две задачи. Логику результатов можно объяснить в терминах задачи, что позволяет использовать дополнительное экспертное знание и определять потенциальные риски. Решение можно представить в виде набора инструкций, что позволяет увеличить его доступность и массовость.

В качестве интерпретируемого метода было выбрано дерево решений с фиксированной глубиной. Выбор обусловлен близостью результата к обычной (человеческой) логике, т.е. выбора тех или иных действий при превышении порогов некоторых величин. Деревья будут аппроксимировать функцию $Q(s,a)$ и будут обучены по оценкам полученных с помощью нейронных сетей.

Помимо получения простых правил в виде инструкции, требуется знать, насколько полученные с помощью них результаты будут хуже по сравнению с результатами исходных алгоритмов. Для этого построенные деревья играли партии с исходными алгоритмами. Полученные результаты, глубина дерева и алгоритм, чьи значения оценок функции ценности использовались при построении, указаны в таблице 3.

Таблица 3. Результат сравнения стратегий, основанных на деревьях решений, со стратегиями, основанными на нейронных сетях

тип алгоритма	параметры алгоритма							глубина дерева	Очки
	λ	γ	α	ε	lf	I	p		
Q-обучение	-	0.9	0.1	0.3	100	150	0.25	2	-9
Q-обучение	-	0.9	0.1	0.3	100	150	0.25	3	-9
Q-обучение	-	0.9	0.1	0.3	100	150	0.25	4	-9
SARSA(λ)	1.0	0.8	0.2	0.5	60	150	0.75	2	-9
SARSA(λ)	1.0	0.8	0.2	0.5	60	150	0.75	3	-9

тип алгоритма	параметры алгоритма							глубина дерева	Очки
SARSA(λ)	1.0	0.8	0.2	0.5	60	150	0.75	4	-9
Q-обучение	-	0.9	0.1	0.5	60	150	0.25	2	-9
Q-обучение	-	0.9	0.1	0.5	60	150	0.25	3	-9
Q-обучение	-	0.9	0.1	0.5	60	150	0.25	4	-9

Из результатов видно, что в данной задаче не существует единого набора правил для получения гарантированного результата. Более того, при фиксировании единой стратегии агент становится неконкурентноспособен по сравнению с агентами с адаптивной стратегией.

Заключение

В работе описаны и построены алгоритмы, основанные на методах обучения с подкреплением, найдены оптимальные параметры алгоритмов. Описан способ увеличения обобщающей способности алгоритмов с помощью нейронных сетей. На примерах показана важность обучения алгоритмов на играх с элементарными стратегиями.

При анализе зависимости качества алгоритмов от длительности предобучения были зафиксированы скачки качества алгоритма, этот эффект проявляется из-за ограниченной памяти и глубины рассматриваемых агентов. Данный эффект согласуется с результатами, полученными для линейных автоматов.

Исследована возможность построения алгоритма, поведение которого основано на простых интерпретируемых методах. Было показано, что такие алгоритмы неконкурентоспособны, т. к. в них пропадает свойство адаптивности и они слишком простые для рассмотренной игры.

Мы считаем приятным долгом поблагодарить Махова С.А. за обсуждения и полезные замечания. Мы постарались их учесть.

Список литературы

1. Обучение с подкреплением / Р. С. Саттон, Э. Г. Барто ; пер. с англ. — М.: БИНОМ. Лаборатория знаний, 2011. — 399 с.
2. Sutton R.S. Learning to predict by the method of temporal differences // Machine Learning. – 1988. – №3 – p.9-44.
3. Watkins C.J. C. H. Learning from Delayed Rewards. Ph.D. thesis, Cambridge University // Natural Science. – 2014. – Vol.6. – №13.
4. Rummery G.A. Problem Solving with Reinforcement Learning. Ph.D. thesis, Cambridge University. – 1995.
5. Bertsekas D.P., Tsitsiklis J.N. Neuro-Dynamic Programming. Athena Scientific, Belmont, M.A. – 1996.
6. Dropout: A Simple Way to Prevent Neural Networks from Overfitting / Srivastaya N., Hinton G., Krizhevsky A., Sutskever I., Salakhutdinov R. // Journal of Machine Learning Research — 2014. — №15. — С. 1929
7. Гельфанд И. М., Пятецкий-Шапиро И. И., Цетлин М. Л. О некоторых классах игр и игр автоматов // Докл. АН СССР, 1963, том 152, номер 4, С. 845-848.
8. Гельфанд И. М., Цетлин М. Л. Принцип нелокального поиска в системах автоматической оптимизации // Докл. АН СССР, 1961, том 137, номер 2, С. 295-298.
9. Гельфанд И. М., Цетлин М. Л. О некоторых способах управления сложными системами // УМН, 1962, том 17, выпуск 1(103), С. 3-25.
10. Цетлин М. Л. Исследования по теории автоматов и моделированию биологических систем. — М.: Наука. Главная редакция физико-математической литературы, 1969. — 316 с.
11. Варшавский В. И., Поспелов В. А. Оркестр играет без дирижера: размышления об эволюции некоторых технических систем и управлении ими. — М.: Наука. Главная редакция физико-математической литературы, 1984. — 208 с.

Содержание