Подход к созданию алгоритма управления в играх Atari

С. А. Беляев[1], Д. А. Коробов[2], А. В. Экало[3]
Факультет компьютерных технологий и информатики,
Санкт-Петербургский государственный электротехнический университет
«ЛЭТИ» им. В.И. Ульянова (Ленина)
[1]bserge@bk.ru, [2]royalruby@yandex.ru, [3]ekalo@nicetu.spb.ru

***Аннотация*. Авторы проводят эксперименты на основе платформы OpenAI Gym. Рассмотрены существующие решения задачи построения алгоритмов управления играми на основе графических входных данных. Предложен подход к построению алгоритма выделения информативных признаков из входных данных для ускорения алгоритмов машинного обучения, решающих задачи управления играми Atari. На этапе предобработки осуществляется обнаружение фона и выделение сгущений – групп связанных и совместно движущихся пикселей, затем распознаются классы объектов и генерируется вектор информативных признаков, который используется как на этапе обучения, так и на этапе применения алгоритма. Представленный подход может быть распространён на другие задачи, использующие машинное обучение.**

Ключевые слова: OpenAI Gym; машинное обучение; вектор информативных признаков; интеллектуальный агент; управление играми Atari

I. Постановка задачи

Современные компьютерные технологии успешно справляются с выполнением достаточно сложных алгоритмов, но при этом не могут превзойти человека в таких задачах, как обработка естественного языка, распознавание изображений и управление в логических и видеоиграх. Тем не менее, в настоящее время наблюдается существенный прогресс в решении данных задач, обусловленный увеличением вычислительной мощности центральных и графических процессоров, а также появлением новых подходов в машинном обучении. Новые технологии позволили вести разработку алгоритмов, которые обучаются как человек, извлекая информацию из неструктурированных данных, что позволяет им реагировать на среду и принимать решения, основанные на предыдущем опыте [1].

В настоящий момент существует открытое программное обеспечение OpenAI Gym [2], позволяющее эмулировать игровую консоль Atari 2600, выпущенную в 1977 году, и предоставляющее возможность использовать её для разработки и тестирования алгоритмов машинного обучения. Авторы OpenAI Gym предлагают проводить эксперименты и сравнивать алгоритмы машинного обучения в единой среде, что позволит получить сравнимые результаты. Задачей данного исследования является разработка интеллектуальных агентов – программ, получающих информацию с помощью датчиков и воздействующих на среду с помощью исполнительных инструментов, способных управлять персонажем в играх Atari. В частности, интерес представляют интеллектуальные агенты, независимые от правил игры, т.е. такие, которые способны проходить несколько различных игр Atari без изменения алгоритма.

В качестве входных данных программа, имитирующая среду Atari, предоставляет последовательность изображений экрана размера 210x160 пикселей, а также общий игровой счёт и индикатор победы или поражения. На основе описанных входных данных модель принимает решение о выборе определённого действия. Обучение агента заключается в выработке правил поведения, которые приведут к получению наибольшего возможного счёта и победе, если игра подразумевает возможность победы.

II. Существующие решения

Обучение с подкреплением является промежуточным между обучением с учителем и обучением без учителя. При обучении с учителем в обучающих выборках имеются метки, позволяющие корректировать модель, а в обучении без учителя метки отсутствуют. Однако в обучении с подкреплением имеются метки с задержкой по времени (награды), с помощью которых агент может обучаться [1]. Обрабатывая предыдущее состояние и награду, агент выбирает действие для выполнения, после чего среда предоставляет следующее состояние и значение награды. В нашем случае состояние – это изображение экрана и индикатор того, является ли состояние терминальным (победа или поражение), а награда – счёт, полученный в результате предыдущего действия.

Существует довольно большое число подходов обучения с подкреплением, которые можно применить к решению задачи обучения интеллектуальных агентов в среде Atari: SARSA, Q-обучение, алгоритмы на основе градиентов по стратегиям, алгоритмы типа «актор-критик» [3, 4]. В последнее время широкое распространение получили алгоритмы, использующие глубокие, свёрточные и рекуррентные нейронные сети, достигнув в некоторых

случаях уровня, превосходящего человека. Значительным ограничением применимости перечисленных алгоритмов является огромная размерность входных данных и ресурсоёмкость задачи обучения при использовании пикселей в качестве входных данных, что делает практически невозможным выполнение алгоритма на низко производительном, например, однопроцессорном, компьютере за разумное время (за несколько часов или дней).

III. Предлагаемый подход

Увеличить скорость и качество обучения может формирование информативных признаков [5]. Использование вектора признаков вместо набора пикселей позволит интеллектуальному агенту ориентироваться лишь на значимые характеристики входных данных и, следовательно, быстрее обучаться. В данной работе предлагается вариант алгоритма генерации вектора признаков, основанный на обнаружении набора классов игровых сущностей (персонажей и окружения) и нахождения положения экземпляров данных классов на каждом изображении экрана. В его основу легли следующие наблюдения: большинство игр Atari имеют статический фон и небольшое число игровых сущностей, перемещающихся по экрану. Все игровые сущности можно разделить на небольшое число непересекающихся классов. При этом задача разбивается на два этапа: на первом этапе для новой игры строится вектор признаков, а на втором этапе – проводится обучение с его использованием.

- Предобработка. Включается в себя обнаружение фона и выделение сгущений – групп связанных и совместно движущихся пикселей. Далее сгущения разделяются на множество классов C.
- Обучение. Как только агент начинает действовать, на каждом шаге распознаются объекты классов $c \in C$, с их помощью происходит генерация вектора признаков.

Для обнаружения фона предлагается использовать гистограммный метод. На заданном наборе кадров для каждого пикселя строится гистограмма, содержащая цвета, которые когда-либо были обнаружены в этой точке. В качестве цвета фона берётся наиболее часто встречающийся цвет пикселя. Далее проводится процедура выделения сгущений, состоящая из двух этапов: сначала помечаются 8-связные области, имеющие одинаковый цвет, а затем сливаются объекты, содержащие несколько составных частей. Под классом объектов понимается множество всех возможных форм (фигур), задающих игровую сущность, где формы представляют различные кадры в анимации объекта. Классы могут содержать обрезанные вариации формы, например, при частичном выходе объекта за пределы экрана.

Общий подход выделения классов заключается в последовательном просмотре изображений экранов $[X_1, X_2, \dots X_n]$, и для каждого объекта o_t на экране X_t осуществляется поиск соответствующего ему объекта o_{t-1}' на экране X_{t-1}. Если объект o_{t-1}' найден, и он отличается от o_t, то можно сделать вывод, что обе формы принадлежат одному и тому же классу. В данном случае предполагается, что объект o_{t-1}' соответствует объекту o_t, если выполняются свойства близости и схожести форм. Схожесть по расстоянию: $dist(o_{t-1}', o_t) < d$. Схожесть по подобию формы: $shape\_similarity(o_{t-1}', o_t) > s$.

В результате, чтобы найти объект o_{t-1}', требуется найти все объекты на экране X_{t-1}, находящиеся на расстоянии менее d от объекта o_t. Далее из полученных объектов, удовлетворяющих формуле схожести по подобию формы, находится тот, для которого это значение является наибольшим среди всех найденных.

Метод распознавания объектов классов использует классы, полученные на предыдущем этапе. Формально ставится задача нахождения списка $\Psi_t(c)$, содержащего координаты и скорости всех экземпляров класса c на экране X_t для каждого класса $c \in C$. Сначала строится список всех возможных фигур всех классов, затем упорядочивается по размеру от самой большой фигуры к самой маленькой. Затем изображение экрана сканируется на предмет наличия каждой фигуры o из списка. Если совпадение найдено, то пиксели помечаются распознанными и в дальнейшем не учитываются. Центр масс распознанных пикселей добавляется в список $\Psi(o.class)$.

Следующим этапом за распознаванием объектов классов является генерация вектора признаков для обучения модели. Вектор признаков в результате должен содержать следующие данные.

- Количество объектов каждого класса. В зависимости от наличия или отсутствия определённых объектов действия агента могут различаться.
- Абсолютное положение каждого объекта. Координаты объектов – основная информация, необходимая для построения эффективной стратегии.
- Векторы скорости объектов. Направление движения играет значительную роль при принятии решения о выборе следующего действия. Скорость определяется разностью координат объектов ψ_t и ψ_{t-1}.

На каждом экране класс c может иметь от 0 до $n_c(t)$ экземпляров, где $n_c(t)$ не является постоянным и зависит от этапа игры. Однако, все алгоритмы обучения с подкреплением требуют на входе вектор фиксированной длины. Следовательно, требуется найти способ эффективной генерации вектора информативных признаков фиксированной длины из распознанных объектов.

Пусть $N_c = \max(n_c(t))$ – максимальное возможное число объектов класса c, которые встретились в последовательности изображений экрана $[X_1, X_2, \dots X_n]$. Тогда для нахождения числа N_c для каждого класса $c \in C$ необходимо на этапе предобработки также сохранять

максимальное количество встретившихся объектов каждого класса.

Используя данный подход, вектор информативных признаков будет иметь размер, зависящий от суммы максимального количества объектов каждого класса. Помимо количества объектов n класса c, в векторе признаков содержатся 2*n координат (x и y) и 2*n скоростей (v_x и v_y). Следовательно, итоговый размер вектора будет равен: $N = size(C) + 4 * \sum N_c$.

Множества классов C и значения N_c вычисляются на этапе предобработки и зависят от каждой конкретной игры. При этом алгоритм создания вектора информативных признаков может реализовывать следующую последовательность шагов для каждого изображения экрана.

Шаг 1. Выделение и классификация объектов в заданном изображении, определение их координат.

Шаг 2. Вычисление скоростей объектов (сравнение с данными предыдущего изображения).

Шаг 3. Для каждого объекта выполнение следующих действий.

Шаг 3.1. Добавление в вектор информативных признаков количества объектов в данном изображении.

Шаг 3.2. Добавление в вектор информативных признаков координат объекта по горизонтали и вертикали.

Шаг 3.3. Добавление в вектор информативных признаков скоростей объекта по горизонтали и вертикали.

Разработанный алгоритм опробован на нескольких играх Atari (платформы OpenAI Gym): Pong, Breakout, SpaceInvaders, MsPacman. Выполнено сравнение с известными подходами: Reinforce, DQN и A3C. В целом все алгоритмы позволили решить исследуемые задачи. Полученный результат позволяет утверждать, что при использовании вектора информативных признаков в среднем сокращается время обучения нейронной сети и увеличивается общий счёт по итогам каждой игры.

IV. Заключение

Предложенный подход к созданию и использованию вектора информативных признаков позволяет увеличить скорость и качество обучения интеллектуальных агентов в среде Atari при использовании подходов обучения с подкреплением. Реализация метода распознавания образов даёт возможность производить обработку видеопотока в режиме реального времени.

Предложенный подход может быть распространён и на другие среды моделирования процессов взаимодействия интеллектуальных агентов [6, 7], единственное ограничение в том, что данные среды должны предоставлять алгоритму набор пикселей, обеспечивая тем самым имитацию изображения на экране.

Список литературы

[1] Yannakakis G., Togelius J. Artificial Intelligence and Games. Springer International Publishing AG 2018. ISBN 978-3-319-63519-4. p. 350.

[2] Беляев С.А., Михнович А.Г. Современные подходы к решению задачи стабилизации перевернутого маятника // Программные продукты, системы и алгоритмы. 2017. №2. С. 2.

[3] Саттон Р.С., Барто Э.Г. Обучение с подкреплением. М.: Бином. Лаборатория знаний, 2017. 400 с.

[4] Николенко С., Кадурин А., Архангельская Е. Глубокое обучение. СПб: Издательский дом «Питер», 2018. 480 с.

[5] Liu D.R., Li H.L., Wang D. Feature selection and feature learning for high-dimensional batch reinforcement learning: A survey // International Journal of Automation and Computing. 2015,.12(3). p. 229–242.

[6] Беляев С.А., Черепкова Ю.С. Архитектура среды моделирования для проведения экспериментов с интеллектуальными агентами // Программные продукты, системы и алгоритмы. 2017. №3. С.4.

[7] Опыт создания среды имитационного моделирования / Беляев С.А., Матросов В.В. // Теоретические и прикладные проблемы развития и совершенствования автоматизированных систем управления военного назначения: Тез. докл. III Всерос.науч.-практ.конф., СПб, 22 ноя. 2017 / СПб.: Военно-космическая академия имени А.Ф. Можайского. 2017. С. 232.