DOI: 10.15593/2499-9873/2019.2.04
УДК 004.89

А.И. Мейер

Национальный исследовательский университет «Высшая школа экономики»,
Пермский филиал, Пермь, Россия

АЛГОРИТМЫ, ПРОГРАММНЫЕ ПАКЕТЫ И ПЛАТФОРМЫ АВТОНОМНОГО УПРАВЛЕНИЯ ПОРТАТИВНЫМ ТРАНСПОРТНЫМ СРЕДСТВОМ В УСЛОВИЯХ НЕОПРЕДЕЛЕННОГО ДИНАМИЧЕСКОГО ЛАНДШАФТА: ЛИТЕРАТУРНЫЙ ОБЗОР

Целью внедрения системы автопилота в транспортное средство является автоматизация его передвижения на динамическом ландшафте. Задачу автопилота можно рассмотреть как разновидность задачи классификации, в которой осуществляется подбор наиболее оптимального набора действий в определенный момент времени в зависимости от состояния, в котором находится объект. За последние годы область обучения такого рода систем получила огромное развитие. Рассказано об актуальных на текущий момент подходах к реализации системы автопилота с акцентированием внимания на портативных аналогах. В частности, охватываются алгоритмы нейронных сетей, пригодные для такого проекта, и проводится сравнительный анализ с использованием доступных на момент написания статьи библиографических материалов. В качестве сравнительных характеристик были выбраны: качество работы, скорость обучения, гарантия сходимости и охват обучения системы. В результате теоретического анализа лучшим алгоритмом для интеграции как в виртуальные, так и в реальные автономные транспортные средства был выбран DQN (глубокое Q-обучение). Также в данной работе приведена информация о полезных для разработчиков готовых решениях: виртуальных платформах и программных пакетах.

Ключевые слова: нейронная сеть, автопилот, автономное управление, киберфизические системы, портативные транспортные средства, глубокое обучение, виртуальные платформы, программные пакеты.

A.I. Meyer

National Research University Higher School of Economics, Perm Branch,
Perm, Russian Federation

METHODS, FRAMEWORKS AND VIRTUAL PLATFORMS IN DOMAIN OF PORTABLE SELF-DRIVING VEHICLES CONTROL ADJUSTED FOR UNDETERMINED DYNAMIC ENVIRONMENT CONDITIONS: LITERATURE REVIEW

The goal of implementing autopilot system into vehicle is the automation of its movement on a dynamic landscape. Vehicle movement task can be considered as a particular case of a classification task, because the problem of every autopilot system is an optimal set of agent actions selection

depending on the vehicle state. Over the recent years a great success has been achieved in self-driving vehicles learning. This article is about current approaches to implementation of autopilot system with the focus on portable vehicles. Particularly, it covers neural networks' algorithms, appropriate for such project, and analyses them between each other according to available state-of-the-art reference sources. Work quality, learning speed, convergence guarantee and AI tasks stack coverage extent were chosen as the comparative characteristics. As a result, DQN (Deep Q-Learning) algorithm was chosen as the best option for implementation in terms of both virtual and real portable self-driving vehicles. This research paper also contains basic details about useful software solutions: virtual platforms and frameworks.

Keywords: neural network, autopilot, autonomous control, cyber-physical systems, portable vehicles, deep learning, virtual platforms, frameworks.

Введение

В последние годы в области машинного обучения особое внимание уделяется транспортным средствам с автопилотом (подсистемой, которая автономно выполняет задачи управления). Это можно объяснить тем, что их внедрение является, согласно описанному в [1] мнению, одной из первых и при этом самой глубокой интеграцией роботов для использования в личных целях, что позволит при развитии данной области в долгосрочной перспективе повысить степень доверия между машиной и человеком. Такая глубина внедрения обосновывается прежде всего тем фактом, что при передаче управления автономному транспортному средству человек доверяет ему свою жизнь и здоровье.

Не меньший интерес представляют и упрощенные портативные аналоги. Транспорт уменьшенного масштаба помимо удовлетворения потребностей отдельных людей может быть использован в более общих интересах: например, при тушении пожаров или транспортировке небольших грузов внутри помещения. Поэтому данную подобласть стоит рассмотреть отдельно.

Целью данного обзора является определение наилучшего алгоритма для самостоятельного принятия решения вышеописанной системой, а также выявление примеров существующих виртуальных платформ и программных пакетов, которые могут быть полезны при реализации искусственного интеллекта.

Исходя из поставленной цели, были выделены следующие **задачи**:

– на основе библиографических материалов составить список подходящих под рассматриваемую систему алгоритмов;

– составить список сравнительных характеристик подобранных алгоритмов;

– провести относительную оценку каждого алгоритма;

– сравнить алгоритмы и выбрать среди них наилучший;

– на основе библиографических материалов сделать обзор, описывающий основные преимущества некоторых бесплатных программных пакетов, способных при их использовании повлиять на качество работы рассматриваемой системы;

– на основе библиографических материалов сделать обзор примеров бесплатных виртуальных платформ, пригодных для проведения экспериментов и тестирования написанных для системы алгоритмов;

– на основе библиографических материалов сделать обзор примеров существующих в рассматриваемой области проектов с выявлением типичных отличительных признаков.

1. Алгоритмы

Как правило, на систему автономного управления транспортным средством возложена задача принятия решения в условиях динамично меняющегося окружающего мира. Его динамика обеспечивается, например, изменениями погоды, освещения, времени суток, сезонов и т.д. Также особая сложность кроется в передвижении объекта управления внутри неопределенной искусственной среды, а также влиянии человеческого фактора на ее функционирование. Тем не менее система должна быть в достаточной степени устойчива к таким изменениям в любой момент времени.

В силу отсутствия конкретизированных данных, описывающих условия, в которых будет работать система автоматического управления, оптимальным выбором архитектурной основы выглядит искусственная нейронная сеть. Из ее основных преимуществ перед традиционными алгоритмами можно выделить универсальность и отказоустойчивость, упомянутые в [2]. Отсутствие необходимости пересмотра алгоритма решения задачи после изменения ее условий, способность к поиску зависимостей в условиях неопределенности и устойчивость к избыточности входных данных являются ключевыми преимуществами перед классическими методами решения трудноформализуемых задач автопилота. Также они позитивно влияют на расширяемость системы, что важно при ее интеграции с различными видами транспортных средств.

Набор входных и выходных данных будет зависеть от охвата обучения системы. Также предполагается использование второстепенных нейронных сетей. Они применимы, например, для преобразования

сложных оптических данных в информацию о том, что находится вблизи транспортного средства. Но выбор алгоритма будет осуществляться именно для основной нейросети, подбирающей оптимальное действие, а информация, описанная выше, будет априори присутствовать как часть ее входа.

Нейросеть – это система, работа которой сильно зависит от адекватности обучающей модели. Для корректной работы автопилота необходимо прежде всего понять, какие топологии способны минимизировать разницу между потенциально возможными и идеальными значениями характеристик, которые будут перечислены далее.

Задачу управления можно рассматривать как частный случай задачи классификации (отнесение состояния объекта, состоящего из набора входных условий, к определенному классу, который является одним из вариантов решения). В работе [3], ссылаясь на возможности программного пакета Statistica Neural Networks 4.0E, для решения задач классификации предлагается использовать одну из четырех видов архитектуры нейросети: многослойный персептрон, сеть радиально-базисных функций, вероятностная нейронная сеть и нейронная сеть Кохонена.

В работе [4] рассказывается про сети радиально-базисных функций, которые имеют более быструю сходимость, чем у персептрона, но при использовании большого обучающего множества и огромного количества входных и выходных параметров сеть физически разрастается очень быстро, что влияет на быстродействие и размер нейросети. Вероятностная сеть обладает теми же недостатками, так как является частным случаем RBF-сети [3]. Нейронная сеть Кохонена больше подходит под задачу кластеризации потому, что заранее неизвестен набор классов, а сеть рассчитана на обучение без учителя [3].

В работе [5] в 2013 г. проектом DeepMind был предложен алгоритм с использованием сверточных нейронных сетей. Сам алгоритм основан на Q-обучении – виде обучения с подкреплением на основе марковского процесса принятия решений, в котором агент, используя нейросеть, рассчитывает результат *функции качества* (Q-функции), с помощью которой он получает сведения о полезности выполнения того или иного действия в текущем внутреннем состоянии [6]. Работа [5] предлагает так называемое глубокое Q-обучение (DQN), приводя примеры обучения на играх Atari 2600, используя на входе изображение экрана.

В работе [7] также были проведены эксперименты с данным алгоритмом. В частности, было сделано сравнение с алгоритмами на основе градиентов по стратегиям (REINFORCE) и асинхронного актора-критика (A3C). В результате был сделан вывод, что алгоритму DQN нужно наименьшее количество эпизодов для обучения, хотя A3C требует меньшее количество шагов обучения. Вероятно, это связано с тем, что A3C формирует поведение сразу на нескольких агентах, что делает стоимость шага кратно выше. Автор работы [7] делает вывод, что алгоритм DQN хорошо справляется с обучением в средах с простыми правилами, ограниченным числом сущностей и тесной связью между оптимальным решением и состоянием экрана. В динамических средах алгоритму приходится уже намного сложнее, но эту проблему можно решить с помощью метода присвоения признаков на основе распознавания образов, что позволит использовать более простую структуру нейронной сети [7].

В работе [8] рассказывается о преимуществах использования глубокого обучения и обучения с подкреплением в задачах автопилота. Глубокое обучение хорошо справляется при преобразовании сенсорных данных в знания [8]. Метод обучения с подкреплением, в отличие от обучения с учителем, заключается в переборе возможных действий в случае, когда агент не имеет достаточного количества информации о внутреннем состоянии, что позволяет возложить процесс понимания окружения на нейронную сеть [8].

В то же время работа [9] критикует алгоритм DQN. Хотя автор и признает, что глубокое обучение с подкреплением Reinforcement Learning (RL) – это набор методов, который при грамотном подходе может быть первым шагом к общему искусственному интеллекту (ИИ), он утверждает, что на данный момент этот вид обучения не работает. К основным тезисам, приведенным в работе, можно отнести неэффективность алгоритма по сравнению с другими методами, сложность реализации правильной функции вознаграждения и вероятность странного поведения после обучения.

Что касается неэффективности, то автор приводит в противовес традиционные алгоритмы. Например, метод Монте-Карло в играх Atari и управление с прогнозирующими моделями при генерации анимации человеческого тела в реальном времени. Но, учитывая преимущества нейросетей, упомянутые ранее, эти алгоритмы не будут давать хоро-

ших результатов в рамках рассматриваемой системы. Странное поведение же возникает при некорректном выводе формулы качества. К примеру, в статье приводится случай с компьютерной игрой Coastrunners 7, симулятором соревнований на моторных лодках. Энтузиасты пытались обучить нейросеть победе в гонках, используя алгоритм DQN. Значение функции качества становилось выше при прохождении контрольных точек и сборе бонусов, позволяющих быстрее завершить игровой уровень. Когда же управление было передано нейросети, вместо логичного прохождения трассы она стремилась собрать как можно больше бонусов, съезжая с траектории и бесконечно подбирая их после повторного появления. Поэтому при использовании DQN важно задать функцию качества, соответствующую поставленной разработчиком цели.

Работа [10] предлагает шесть расширений для алгоритма DQN, которые позволяют улучшить производительность и качество обучения нейронной сети: Double DQN, Prioritized DQN, Dueling DQN, A3C, Distributional DQN и Noisy DQN. В частности, Prioritized DQN предлагает внедрение краткосрочной памяти как части состояния объекта, что кардинально повысило шанс сходимости целевой функции нейросети [11]. К тому же, согласно работе [8], алгоритм глубокого обучения с подкреплением, в котором учитывается прошлый опыт при принятии решения, становится более устойчивым к динамике среды. Комбинация с использованием всех расширений была названа Rainbow DQN, которая показала наилучшие среди рассматриваемых расширений результаты в скорости обучения и способности состязаться с человеком.

Также стоит упомянуть DRQN – DQN с использованием рекуррентной нейросети. На первый взгляд, как и Prioritized DQN, сеть учитывает предыдущий опыт. Однако, если Prioritized DQN использует experience replay, которая внедряет краткосрочную память, то с DRQN появляется зависимость результата от долгосрочной памяти. Комбинация обоих подходов, согласно результатам работы [12], также позволяет улучшить характеристики системы.

Используя рассмотренные материалы, составили таблицу алгоритмов (табл. 1), характеризующую каждый из них при обучении и работе нейросети. Она описывает основные особенности сетей, а также приводит относительную оценку характеристик, набор и значения которых перечислены ниже.

Таблица 1

Характеристики алгоритмов, пригодных для решения задачи автопилота на динамическом ландшафте

Алгоритм	Характер обучения	Характер связей	Охват обучения системы	Скорость обучения	Качество работы	Гарантия сходимости
Алгоритм многослойного персептрона	Обучение с учителем	Прямое распространение	Низкий	Низкая	Низкое	Есть
Алгоритм сети радиально-базисных функций (RBF)	Смешанное обучение	Радиально-базисные функции	Низкий	Низкая	Низкое	Есть
Алгоритм вероятностной нейронной сети	Смешанное обучение	Радиально-базисные функции	Низкий	Высокая	Низкое	Есть
Алгоритм нейронной сети Кохонена	Обучение без учителя	Самоорганизующиеся карты	Низкий	Низкая	Низкое	Есть
Алгоритм асинхронного актора-критика (A3C)	Смешанное обучение	Прямое распространение	Высокий	Низкая	Высокое	Нет
Алгоритм на основе градиентов по стратегиям (Reinforce)	Смешанное обучение	Прямое распространение	Высокий	Низкая	Высокое	Нет
Алгоритм глубокого Q-обучения (DQN)	Смешанное обучение	Прямое распространение	Высокий	Низкая	Высокое	Есть

1. Характер обучения классифицирует алгоритмы нейросети по способности обучаться с учителем или без него:

– обучение с учителем,

– смешанное обучение (в том числе обучение с подкреплением),

– обучение без учителя.

2. Характер связей классифицирует алгоритмы нейросети по направлению сигнала между нейронами и его влиянию на обучение и работу сети:

– прямое распространение,

– радиально-базисные функции,

– самоорганизующиеся карты.

3. Охват обучения системы описывает количество функций из стека ИИ, представленного в [8], которые может взять на себя нейросеть с данной архитектурой:

– низкий (не более 2–3 функций),

– высокий (более 2–3 функций).

4. Скорость обучения характеризует способность текущей нейросети относительно остальных обучаться за минимально возможный временной промежуток: низкая, высокая.

5. Качество работы описывает оперативность и точность принятия решения текущей нейронной сетью: низкое, высокое.

6. Гарантия сходимости описывает наличие стремления разницы между текущим и оптимальным значениями целевой функции к нулю (минимизация ошибки тестирования): есть, нет.

На основе выбранных характеристик составим формулу оценки алгоритма:

$$R = 2 \cdot \mathrm{Cov} + \mathrm{Lrn} + 2 \cdot \mathrm{Wrk} + 3 \cdot \mathrm{Con},$$

$$\mathrm{Cov} \in \{0;1\},$$

$$\mathrm{Lrn} \in \{0;1\},$$

$$\mathrm{Wrk} \in \{0;1\},$$

$$\mathrm{Con} \in \{0;1\},$$

где R – общая оценка алгоритма; Cov – охват обучения системы; Lrn – скорость обучения; Wrk – качество работы; Con – гарантия сходимости.

Диапазон значений переменных выбран в соответствии с набором возможных значений соответствующих характеристик. Обоснование выбранных коэффициентов для переменных приведено в табл. 2.

Таблица 2

Обоснование коэффициентов формулы оценки алгоритма

Переменная	Приоритет	Обоснование
Con	Высокий	Отсутствие гарантии сходимости может повлечь за собой стохастическое поведение вне зависимости от адекватности построенной модели и полноты обучающего набора данных, что ставит под вопрос обучаемость сети даже на уровне эксперимента
Cov	Средний	Учитывая высокую сложность формализации рассматриваемой задачи, количество операций, которые можно возложить на нейросеть, является немаловажным, хотя и не критичным параметром алгоритма (часть операций можно реализовать традиционными методами)
Wrk	Средний	Качество системы с точки зрения пользователя и качество работы нейросети находятся в прямой корреляции, хотя оперативность и точность поведения могут быть достигнуты, например, уменьшением диапазона задач, которые может выполнять рассматриваемая система
Lrn	Низкий	Длительность процесса обучения незначительна на фоне его результата. Иногда высокая скорость обучения вносит некоторые ограничения. Например, метод Левенберга – Маркара в многослойном персептроне чувствителен к избыточности данных и их большому количеству, а также применим для сетей с единственным выходным элементом [13]. В такой нетривиальной задаче, как управление транспортом, подобная оптимизация неприменима

Применяя полученную формулу, оценим рассматриваемые алгоритмы нейросетей (табл. 3).

Таблица 3

Сравнение алгоритмов, пригодных для решения задачи автопилота на неопределенном динамическом ландшафте (авторские результаты)

Алгоритм	Cov	Lrn	Wrk	Con	R
Алгоритм многослойного персептрона	0	0	0	1	3
Алгоритм сети радиально-базисных функций (RBF)	0	0	0	1	3
Алгоритм вероятностной нейронной сети	0	1	0	1	4
Алгоритм нейронной сети Кохонена	0	0	0	1	3
Алгоритм асинхронного актора-критика (A3C)	1	0	1	0	4
Алгоритм на основе градиентов по стратегиям (REINFORCE)	1	0	1	0	4
Алгоритм глубокого Q-обучения (DQN)	1	0	1	1	7

Для начала было решено сделать оценку алгоритма многослойного персептрона. Метод обратного распространения ошибки в данном случае будет наиболее предпочтительным из-за сложности и нетривиальности рассматриваемой задачи, так как не ставит дополнительных ограничений при моделировании нейросети, упомянутых для алгоритмов второго порядка в [13]. Поэтому анализ архитектуры будет производиться с учетом использования данного метода. В работе [14] упоминаются такие недостатки многослойного персептрона, как большое количество вычислений и связанных нейронов при обучении и решении задач с соответствующим объемом используемой информации, а также игнорирование топологии входных данных. Следовательно, данная сеть в системе автопилота будет работать медленно и некорректно, а также иметь значительные требования к объему памяти. Доказательство гарантии сходимости приведено в [15].

Работа сети RBF, радиально-базисных функций, в задачах классификации, согласно результатам работы [16], зависит от объема обучающей выборки и типов данных. Многослойный персептрон наиболее точно решает задачу классификации вместе с увеличением количества обучающих данных. Наряду с этим, согласно статье [17], эта нейросеть обучается быстрее RBF. Следовательно, проблемы производительности и обучения в случае с сетью радиально-базисных функций

становятся еще более значительными. В статье [18] даны примеры алгоритмов на базе RBF-сетей, которые имеют глобальную сходимость. В остальном же сеть имеет тот же набор преимуществ и недостатков, что и персептрон.

Вероятностная нейронная сеть, как было сказано ранее, является частным случаем RBF-сети, в основном с теми же плюсами и минусами. Ее основное преимущество заключается в особенности обучения, процесс которого сводится к внедрению входных данных в сеть, что значительно увеличивает скорость обучения по сравнению с уже рассмотренными нейросетями. С другой стороны, материал [13] говорит, что из-за того, что нейросеть фактически содержит весь обучающий набор данных, то ее размер при огромной выборке будет соответствующим. Также, согласно [19], сходимость при использовании данной архитектуры гарантируется при использовании достаточного количества данных для обучения.

Алгоритм сети Кохонена, с точки зрения задачи классификации, пригоден больше для поиска наиболее близких классов, чем для оперативного отнесения данных к какому-либо классу [13]. С другой стороны, можно использовать распознавание с учителем, но, как и в предыдущих алгоритмах, будет остро стоять проблема размера нейросети. Что касается скорости обучения, то она сопоставима с алгоритмом RBF [20]. В работе [21] был получен показатель, благодаря которому можно оценить сходимость смоделированной нейросети на базе самоорганизующихся карт.

Согласно [8], при использовании алгоритмов глубинного обучения увеличивается потенциальный охват процедур стека ИИ, так как соответствующие нейросети способны действовать в более общей парадигме, без ориентации на конкретный тип задач. Также в статье [22] указывается на преимущество таких нейросетей в производительности и точности на фоне классических алгоритмов. При обучении же эти сети показывают себя хуже, так как отсутствие адаптации под определенные задачи требует значительного количества времени для формирования сети. В статье [7] были приведены в пример алгоритмы A3C, REINFORCE и DQN, использующие методы глубокого обучения с подкреплением. В ней же упоминается, что при их использовании отсутствует гарантия сходимости. Но для алгоритма DQN было предложено использовать описанное ранее расширение Prioritized DQN,

гарантирующее сходимость и, следовательно, дающее преимущество на фоне остальных двух алгоритмов.

Оценив вышеперечисленные алгоритмы, можно сделать вывод, что DQN наиболее подходит для реализации автономного управления транспортным средством. Несмотря на чувствительность алгоритма к ошибкам, допущенным при реализации, DQN имеет потенциал как топология нейросети, способная, теоретически, работать в общей парадигме, что необходимо при реализации системы автопилота. Конечно, стоит помнить, что архитектура еще недостаточно изучена, так как была предложена еще не так давно. Но, с другой стороны, первоначальный вариант DQN уже имеет множество улучшений, предложенных за относительно короткий период, которые повышают его способность качественно работать.

2. Программные пакеты

Разработчику системы автономного управления транспортного средства будут полезны готовые программные решения для оптимизации ее функционирования. Ниже приведены примеры существующих программных пакетов в данной области.

В работе [23] описывается решение Torch, предоставляющее поддержку алгоритмов машинного обучения и использующее мощности GPU, которые, в отличие от CPU, способны быстро выполнять множество одинаковых операций, что полезно при распараллеливании вычислений. В библиотеке поддерживается язык сценариев Lua, а также доступно большое количество расширений для совместимости с разными языками и технологиями. В работе [24] на базе данной библиотеки приводится пример реализации DQN.

Материал [25] описывает программный пакет OpenCV, дающий доступ к методам компьютерного зрения для распознавания образов. Его функциональность может быть использована для формализации сенсорных данных.

3. Виртуальные платформы

Для предварительного тестирования алгоритмов и проведения экспериментов с ИИ подходит виртуальная платформа – симулятор реального мира, который может предоставить доступ к функциональному набору для управления агентами в динамической среде с имитацией физики в трехмерном пространстве.

Работа [26] описывает CARLA – симулятор с открытым исходным кодом для исследований в области автономного управления транспортным средством. CARLA позволяет отслеживать процесс управления, работать с несколькими агентами и использовать специально смоделированные свободно распространяемые объекты для заполнения ландшафта среды. Также платформа позволяет симулировать работу различных датчиков для получения сенсорных данных.

Материал из [27] рассказывает об Autoware. Это программный продукт от Tier IV с открытым исходным кодом, предлагающий многомодульную платформу для внедрения автопилота в целях эксплуатации на открытых урбанистических зонах. Autoware позволяет как создать реальный полноразмерный автомобиль с внедренным автопилотом, так и проводить эксперименты в специальной виртуальной среде. На базе данной платформы уже существует несколько проектов от разных компаний.

Авторы работы [28] рассказывают о платформе AirSim от корпорации Microsoft. AirSim построен на базе игрового движка Unreal Engine и работает как симулятор беспилотных летательных аппаратов и автомобилей. Разработчики AirSim говорят о том, что платформа создается как площадка для исследований в сфере ИИ и экспериментов с алгоритмами глубокого обучения, компьютерного зрения и обучения с подкреплением.

Симулятор Gazebo, описанный в работе [29], работает в более общей области. Разработчики позиционируют данное решение как платформу с уклоном в робототехнику. Она позволяет создавать в трехмерном пространстве среду с поддержкой нескольких роботов одновременно, используя сложную систему просчетов физики, возможность интеграции ОС ROS, набор виртуальных датчиков и библиотеку трехмерных моделей.

Также есть компьютерные игры, которые можно отнести к доступным любому разработчику платформам. В статье [30] рассказывается о модификациях к игре Euro Truck Simulator 2, через которые можно, в том числе, управлять игровым ИИ и транспортом. В материале [31] же описывается сетевая модификация Multi Theft Auto, работающая на базе игры Grand Theft Auto: San Andreas. Данная модификация нетребовательна к характеристикам ЭВМ, поддерживает функции для построения пользовательского интерфейса и ИИ, а также под-

держивает сетевую игру, что также можно использовать при экспериментах.

4. Существующие проекты

Если говорить о транспорте уменьшенного масштаба с системой автопилота, то сейчас подобные проекты разрабатываются в основном на промежуточных этапах, перед созданием реального прототипа. Программная часть некоторых проектов в сфере автономного управления основана на специально разработанной для сферы робототехники операционной системе ROS (библиотека, ориентированная на Unix-подобные системы), о которой рассказывается в книге [32].

В материалах [33]–[35] энтузиаст Sanyam Bhutani занимается разработкой автопилота. Изначально проект разрабатывался на базе игрушечного автомобиля. Затем он решил сменить направление в сторону автопилота для летающих машин. Sanyam строит вычислительные мощности на базе Raspberry Pi.

В работе [36] рассказывается про симулятор автономного вождения от LG Silicon Valley Lab, построенный на базе Unity. Симулятор имеет поддержку интеграции TierIV Autoware и Baidu Apollo. Данный проект является виртуальным отражением проекта реального многоагентного динамического ландшафта MIT DuckieTown, представляющего собой уменьшенную копию города с передвигающимися по нему двухколесными транспортными средствами, именующимися DuckieBot, с фронтальными камерами в качестве датчиков.

Другой энтузиаст из Техаса по имени Kinsley Harrison разрабатывает автопилот на базе игры Grand Theft Auto V с использованием языка программирования Python. Проект называется PYGTAV. Свой выбор платформы он мотивирует тем, что сама игра является реализацией открытого виртуального мира с уже реализованной физикой и динамическим ландшафтом, что делает ее идеальной для подобных экспериментов. В рамках данного проекта существует ряд статей на сайте PythonProgramming, в которой автор рассказывает об успехах, проблемах и решениях с примерами кода и отчетами в видеоформате. Первая часть приведена в [37]. Основные особенности упомянутых проектов были вынесены в табл. 4.

Таблица 4

Основные особенности рассматриваемых проектов
в области автономных портативных транспортных средств

Название проекта	Вид транспортного средства	Разработчик виртуальной системы	Платформа виртуальной системы	Разработчик реальной системы	Основа агента реальной системы	Имеющиеся сенсоры
A Self Driving Year	Летающий автомобиль	Sanyam Bhutani	Euro Truck Simulator 2 / CARLA	Sanyam Bhutani	Raspberry Pi 3 для тестирования и последующий переход на NVIDIA Jetson TX-2	Не определены
PYGTAV	Наземный автомобиль	Kinsley Harrison	Grand Theft Auto V	–	–	Игровая фронтальная камера (в виртуальной системе)
A ROS/ROS2 Multi-robot Simulator for Autonomous Vehicles	Наземный автомобиль	LG Silicon Valley Lab	Unity	MIT DuckieTown Foundation	DuckieBot	Фронтальная камера

Также есть множество других любительских и профессиональных иностранных проектов на базе уменьшенных моделей транспортных средств, каждый со своим подходом в зависимости от поставленных целей. Подробное рассмотрение каждого из них выходит за пределы задач, поставленных в начале этой статьи.

Заключение

В рамках данной статьи были рассмотрены алгоритмы работы нейронных сетей, пригодных для разработки системы автономного управления портативным транспортным средством. Наилучшей для разработки системы была выбрана сеть DQN.

Также был сделан обзор бесплатных решений в виде виртуальных платформ и программных пакетов. Библиотека Torch пригодна для оптимизации работы нейросетей, а инструмент OpenCV предоставляет функциональность для работы со сложными оптическими сенсорными данными. Они могут пригодиться при непосредственной разработке и тестировании как виртуальной, так и реальной системы искусственного интеллекта, уменьшив объем работы для команды разработчиков.

В конце был проведен обзор и анализ существующих в рассматриваемой области проектов. Все они отличаются различными подходами и выбором составляющих элементов систем.

Все достигнутые в рамках данной статьи результаты могут пригодиться в дальнейшем разработчикам и архитекторам при проектировании и реализации прототипа автономного портативного транспортного средства, равно как и ее виртуальной интерпретации. Также существуют перспективы для более узкого и глубокого сравнительного анализа в этой области.

Список литературы

1. Bhutani S. MIT 6.S094: Deep Learning for Self-Driving Cars 2018 Lecture 1 Notes. – URL: https://hackernoon.com/mit-6-s094-deep-learning-for-self-driving-cars-2018-lecture-1-notes-807be1a50893 (accessed 01 May 2019).

2. Головинов А.О., Климова Е.Н. Преимущества нейронных сетей перед традиционными алгоритмами // Экспериментальные и теоретические исследования в современной науке: сб. ст. по материалам V междунар. науч.-практ. конф. – Новосибирск, 2017. – № 5. – С. 11–15.

3. Сетлак Г. Использование искусственных нейронных сетей для решения задач классификации в менеджменте // Радіоелектроніка, інформатика, управління. – 2004. – № 1. – С. 127–135.

4. Некипелов Н. Введение в RBF-сети [Электронный ресурс]. – URL: https://basegroup.ru/community/articles/rbf (дата обращения: 01.05.2019).

5. Playing Atari with Deep Reinforcement Learning / V. Mnih, K. Kavukcuoglu, D. Silver [et al.] // NIPS Deep Learning Workshop. – Lake Tahoe, USA, 2013.

6. Рассел С., Норвиг П. Искусственный интеллект. Современный подход. – М.: Вильямс, 2007. – 1408 с.

7. Коробов Д.А., Беляев С.А. Современные подходы к обучению интеллектуальных агентов в среде Atari // Программные продукты и системы. – 2018. – Т. 31, № 2. – С. 284–290.

8. Bhutani S. MIT 6.S094: Deep Learning for Self-Driving Cars 2018 Lecture 3 Notes: Deep Reinforcement Learning. – URL: https://hackernoon.com/mit-6-s094-deep-learning-for-self-driving-cars-2018-lecture-3-notes-deep-reinforcement-learning-fe9a8592e14a (accessed 01 May 2019).

9. Irpan A. Deep Reinforcement Learning Doesn't Work Yet. – URL: https://www.alexirpan.com/2018/02/14/rl-hard.html (accessed: 01 May 2019).

10. Rainbow: Combining Improvements in Deep Reinforcement Learning / M. Hessel, J. Modayil, T. Schaul [et al.] // AAAI. – 2018. – P. 3215–3222.

11. Schaul T., Quan J., Antonoglou I., Silver D. Prioritized experience replay // ICLR. – Hilton, San Juan, Puerto Rico, 2016.

12. Schulze C., Schulze M. ViZDoom: DRQN with Prioritized Experience Replay, Double-Q Learning, & Snapshot Ensembling. – URL: https://arxiv.org/pdf/1801.01000 (accessed 01 May 2019).

13. StatSoft, Inc. Электронный учебник по статистике. – URL: http://www.statsoft.ru/home/textbook/default.htm (дата обращения: 01.05.2019).

14. Солдатова О.П., Гаршин А.А. Применение сверточной нейронной сети для распознавания рукописных цифр // Компьютерная оптика. – 2010. – Т. 34, № 2. – С. 252–259.

15. Rumelhart D., McClelland J. Parallel distributed processing: explorations in the microstructure of cognition. Vol. 1: Foundations. – MIT Press Cambridge, 1986. – 547 p.

16. Sug H. Performance Comparison of RBF networks and MLPs for Classification // Proceedings of the 9th WSEAS International Conference on applied Informatics and Communications (AIC′09). – 2009. – P. 450–454.

17. Kayri M. An Intelligent Approach to Educational Data: Performance Comparison of the Multilayer Perceptron and the Radial Basis Function Artificial Neural Networks // Educational Sciences: Theory & Practice. – 2015. – Vol. 15, № 5. – P. 1247–1255.

18. Wild S., Shoemaker C. Global Convergence of Radial Basis Function Trust Region Derivative-Free Algorithms // SIAM J. Optim. – 2011. – Vol. 21, № 3. – P. 761–781.

19. Devillers J. Endocrine Disruption Modeling. – CRC Press, 2017. – 424 p.

20. Gil D., Johnsson M. Supervised SOM Based Architecture versus Multilayer Perceptron and RBF Networks // Linköping Electron Conf. – 2010. – P. 15–24.

21. Benjamin O. A convergence criterion for self-organizing maps // Dissertations and Master's Theses (Campus Access). – 2012. – Paper AAI1508312.

22. Seif G. Deep Learning vs Classical Machine Learning. – URL: https://towardsdatascience.com/deep-learning-vs-classical-machine-learning-9a42c6d48aa (accessed 01 May 2019).

23. Collobert R., Kavukcuoglu K., Farabet C. Torch7: A Matlab-like Environment for Machine Learning // BigLearn, NIPS Workshop. – 2011.

24. Gordon M. Deep Q-Network Training. – URL: https://github.com/soumith/cvpr2015/blob/master/DQN%20Training%20iTorch.ipynb (accessed 01 May 2019).

25. Обработка изображений с помощью OpenCV / Г. Буэно Гарсия, О.Д. Суарес, Х.Л. Эспиноса Аранда [и др.]. – М.: ДМК Пресс, 2016. – 210 с.

26. CARLA: An Open Urban Driving Simulator / A. Dosovitskiy, G. Ros, F. Codevilla [et al.] // CoRL. – Mountain View, California, 2017. – P. 11–16.

27. Autoware on Board: Enabling Autonomous Vehicles with Embedded Systems / S. Kato, S. Tokunaga, Y. Maruyama [et al.] // ICCPS. – Porto, Portugal, 2018. – P. 287–296.

28. AirSim: High-Fidelity Visual and Physical Simulation for Autonomous Vehicles / S. Shah, D. Dey, C. Lovett, A. Kapoor // FSR. – Zürich, Switzerland, 2017.

29. Koenig N., Howard A. Design and Use Paradigms for Gazebo, An Open-Source Multi-Robot Simulator // IROS. – Sendai, Japan, 2004. – P. 2149–2154.

30. Comiskey C. The Very Best Euro Truck Simulator 2 Mods. URL: https://www.geforce.com/whats-new/articles/the-very-best-euro-truck-simulator-2-mods (accessed: 01.05.2019).

31. Moss R. Why A Million People Still Play Multiplayer Grand Theft Auto: San Andreas Every Month. – URL: https://www.rockpapershotgun.com/2016/09/15/why-a-million-people-still-play-multiplayer-grand-theft-auto-san-andreas-every-month/ (accessed 01 May 2019).

32. Lentin J. Robot Operating System (ROS) for Absolute Beginners: Robotics Programming Made Easy. – Apress, 2018 – 282 p.

33. Bhutani S. A Self Driving New Year № 1. – URL: https://hackernoon.com/a-self-driving-new-year-33284e592f35 (accessed 01 May 2019).

34. Bhutani S. A Self Driving New Year № 2. – URL: https://hackernoon.com/a-self-driving-new-year-2-d1bbc5a83570 (accessed 01 May 2019).

35. Bhutani S. A Self Driving and Flying (New) Year № 3. – URL: https://hackernoon.com/a-self-driving-and-flying-new-year-3-30d5ecd375e8 (accessed 01 May 2019).

36. Zelenkovsky D. A ROS/ROS2 Multi-robot Simulator for Autonomous Vehicles. – URL: https://github.com/lgsvl/simulator/blob/master/README.md (accessed 01 May 2019).

37. Harrison K. Reading game frames in Python with OpenCV – Python Plays GTA V. – URL: https://pythonprogramming.net/game-frames-open-cv-python-plays-gta-v/ (accessed 01 May 2019).

References

1. Bhutani S. MIT 6.S094: Deep Learning for Self-Driving Cars 2018 Lecture 1 Notes, available at: https://hackernoon.com/mit-6-s094-deep-learning-for-self-driving-cars-2018-lecture-1-notes-807be1a50893 (date of access: 01.05.2019).

2. Golovinov A.O., Klimova E.N. The advantages of neural networks over traditional algorithms, Experimentalniye i teoreticheskiye issledovaniya v sovremennoy nauke: sb. st. po mater. V mezhdunar. nauch.-prakt. konf. № 5(5), Novosibirsk, 2017, pp. 11–15 (in Russian).

3. Setlak G. The use of artificial neural networks to solve classification tasks in management field, Radioelektronika, informatika, upravlenye, 2004, № 1, p. 127–135 (in Russian).

4. Nekipelov N. Introduction to RBF networks, available at: https://basegroup.ru/community/articles/rbf (date of access: 01.05.2019).

5. V. Mnih, K. Kavukcuoglu, D. Silver et al. Playing Atari with Deep Reinforcement Learning, NIPS Deep Learning Workshop, Lake Tahoe, USA, 2013.

6. Russell S., Norvig P. Iskustvennii Intellekt. Sovremennii podhod (Artificial Intelligence: A Modern Approach), Moscow, Williams Publishing, 2007, 1408 p. (in Russian).

7. Korobov D.A., Belyaev S.A. Modern training approaches of intelligent agents inside Atari environment, Programmnye produkty i systemy, 2018, Vol. 31, № 2, P. 284–290 (in Russian).

8. Bhutani S. MIT 6.S094: Deep Learning for Self-Driving Cars 2018 Lecture 3 Notes: Deep Reinforcement Learning, available at: https://hackernoon.com/mit-6-s094-deep-learning-for-self-driving-cars-2018-lecture-3-notes-deep-reinforcement-learning-fe9a8592e14a (date of access: 01.05.2019).

9. Irpan A. Deep Reinforcement Learning Doesn't Work Yet, available at: https://www.alexirpan.com/2018/02/14/rl-hard.html (date of access: 01.05.2019).

10. Hessel M., Modayil J., Schaul T. et al. Rainbow: Combining Improvements in Deep Reinforcement Learning, AAAI, 2018, p. 3215–3222.

11. Schaul T., Quan J., Antonoglou I., Silver D. Prioritized experience replay, ICLR, Hilton, San Juan, Puerto Rico, 2016.

12. Schulze C., Schulze M. ViZDoom: DRQN with Prioritized Experience Replay, Double-Q Learning, & Snapshot Ensembling, available at: https://arxiv.org/pdf/1801.01000 (date of access: 01.05.2019).

13. StatSoft, Inc. Electronic textbook on statistics, available at: http://www.statsoft.ru/home/textbook/default.htm (date of access: 01.05.2019) (in Russian).

14. Soldatova O.P., Garshin A.A. The use of convolutional neural network for handwritten digits recognition, Komputernaya optika, 2010, Vol. 34, № 2, pp. 252–259 (in Russian).

15. Rumelhart D., McClelland J. Parallel distributed processing: explorations in the microstructure of cognition, vol. 1: foundations – MIT Press Cambridge, 1986 – 547 p.

16. Sug H. Performance Comparison of RBF networks and MLPs for Classification, Proceedings of the 9th WSEAS International Conference on applied Informatics and Communications (AIC ’09), 2009, P. 450–454.

17. Kayri M. An Intelligent Approach to Educational Data: Performance Comparison of the Multilayer Perceptron and the Radial Basis Function Artificial Neural Networks, Educational Sciences: Theory & Practice, 2015, Vol. 15, № 5, pp. 1247–1255.

18. Wild S., Shoemaker C. Global Convergence of Radial Basis Function Trust Region Derivative-Free Algorithms // SIAM J. Optim, 2011, Vol. 21, № 3, pp. 761–781.

19. Devillers J. Endocrine Disruption Modeling. – CRC Press, 2017. – 424 p.

20. Gil D., Johnsson M. Supervised SOM Based Architecture versus Multilayer Perceptron and RBF Networks, Linköping Electron Conf, 2010, pp. 15–24.

21. Benjamin O. A convergence criterion for self-organizing maps // Dissertations and Master's Theses (Campus Access). – 2012. – Paper AAI1508312.

22. Seif G. Deep Learning vs Classical Machine Learning, available at: https://towardsdatascience.com/deep-learning-vs-classical-machine-learning-9a42c6d48aa (date of access: 01.05.2019).

23. Collobert R., Kavukcuoglu K., Farabet C. Torch7: A Matlab-like Environment for Machine Learning, BigLearn, NIPS Workshop, 2011.

24. Gordon M. Deep Q-Network Training, available at: https://github.com/soumith/cvpr2015/blob/master/DQN%20Training%20iTorch.ipynb (date of access: 01.05.2019).

25. Gloria Bueno García, Oscar Deniz Suarez, José Luis Espinosa Aranda et al. Obrabotka izobrazheniy s pomoshyu OpenCV (Learning Image Processing with OpenCV), Moscow, DMK Press Publishing, 2016, 210 p. (in Russian).

26. Dosovitskiy A., Ros G., Codevilla F. et al. CARLA: An Open Urban Driving Simulator, CoRL, Mountain View, California, 2017, pp. 11–16.

27. Kato S., Tokunaga S., Maruyama Y. et al. Autoware on Board: Enabling Autonomous Vehicles with Embedded Systems, ICCPS, Porto, Portugal, 2018, p. 287–296.

28. Shah S., Dey D., Lovett C., Kapoor A. AirSim: High-Fidelity Visual and Physical Simulation for Autonomous Vehicles, FSR, Zürich, Switzerland, 2017, pp. 2149–2154.

29. Koenig N., Howard A. Design and Use Paradigms for Gazebo, An Open-Source Multi-Robot Simulator, IROS, Sendai, Japan, 2004.

30. Comiskey C. The Very Best Euro Truck Simulator 2 Mods, available at: https://www.geforce.com/whats-new/articles/the-very-best-euro-truck-simulator-2-mods (date of access: 01.05.2019).

31. Moss R. Why A Million People Still Play Multiplayer Grand Theft Auto: San Andreas Every Month, available at: https://www.rockpapershotgun.com/2016/09/15/why-a-million-people-still-play-multiplayer-grand-theft-auto-san-andreas-every-month/ (date of access: 01.05.2019).

32. Lentin J. Robot Operating System (ROS) for Absolute Beginners: Robotics Programming Made Easy, Apress, 2018, 282 p.

33. Bhutani S. A Self Driving New Year #1, available at: https://hackernoon.com/a-self-driving-new-year-33284e592f35 (date of access: 01.05.2019).

34. Bhutani S. A Self Driving New Year #2, available at: https://hackernoon.com/a-self-driving-new-year-2-d1bbc5a83570 (date of access: 01.05.2019).

35. Bhutani S. A Self Driving and Flying (New) Year #3, available at: https://hackernoon.com/a-self-driving-and-flying-new-year-3-30d5ecd375e8 (date of access: 01.05.2019).

36. Zelenkovsky D. A ROS/ROS2 Multi-robot Simulator for Autonomous Vehicles, available at: https://github.com/lgsvl/simulator/blob/master/README.md (date of access: 01.05.2019).

37. Harrison K. Reading game frames in Python with OpenCV - Python Plays GTA V, available at: https://pythonprogramming.net/game-frames-open-cv-python-plays-gta-v/ (date of access: 01.05.2019).

Получено 22.05.2019

Сведения об авторе

Мейер Артем Игоревич (Пермь, Россия) – бакалавр, Национальный исследовательский университет «Высшая школа экономики», Пермский филиал (614070, Россия, г. Пермь, ул. Студенческая, 38, e-mail: meyer59@mail.ru).

About the author

Artem I. Meyer (Perm, Russian Federation) – Bachelor, National Research University Higher School of Economics, Perm Branch (38, Studencheskaya st., Perm, 614070, Russian Federation, e-mail: meyer59@mail.ru).