РЕФЕРАТ

Отчет 26 с., 8 рис., 4 табл., 13 источн.

ОБУЧЕНИЕ С ПОДКРЕПЛЕНИЕМ, ГЛУБОКОЕ ОБУЧЕНИЕ, ТЕОРИЯ ИГР

Объектом исследования являются алгоритмы обучения с подкреплением в среде с несколькими агентами.

Цель данной научно-исследовательской работы — провести анализ существующих методов обучения с подкреплением в среде с несколькими агентами (MARL).

Для достижения поставленной цели необходимо решить следующие задачи:

— формализовать задачу используя математический аппарат;
— провести анализ предметной области методов обучения с подкреплением для задач игрового искусственного интеллекта;
— сформулировать способы классификации методов;
— классифицировать методы исходя из способов;
— сравнить описанные алгоритмы;
— отразить результаты сравнения и классификации рассмотренных алгоритмов в выводе.

Результатом работы является выявление критериев применимости алгоритмов к задаче игорового искусственного интеллекта.

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

Задача принятия решений является важно нерешенной задачей во многих областях. Одним из подходов в задачах принятия решений является обучение с подкреплением. Проблема формулируется как Марковский процесс принятия решений. Такая формулировка позволяет использовать определенный подход. Классические алгоритмы обучения с подкреплением (Reinforc Learning, RL) подразумевают среду с одним агентом. В данной работе рассматриваются проблемы, в которых несколько агентов. Они должны взаимодействовать друг с другом, при этом взаимодействие может выражаться в кооперации, в конкуренции, или в смешанном варианте.

Примером предметной области применения нескольких агентов являются: управление в робототехнике, менеджмент ресурсов, коллаборативные системы принятия решений, майнинг данных и т. д. Рассматривается предметная область игрового искусственного интеллекта. Она достаточно простая в сравнении. Также в ней нет необходимости взаимодействовать с реальным миром, что позволяет использовать симуляцию.

Средой будем называть игровое пространство, в которой происходит взаимодействие агентов. Среда может подразумевать кооперацию (одна команда), конкуренцию (один против всех), а также смешанную конкуренцию и кооперацию одновременно (несколько команд).

Классификация алгоритмов обучения в среде с несколькими агентами позволит выбрать наиболее подходящий алгоритм для решения конкретной задачи.

Цель данной научно-исследовательской работы — провести анализ

существующих методов обучения с подкреплением в среде с несколькими агентами (MARL).

Для достижения поставленной цели необходимо решить следующие задачи:

— формализовать задачу используя математический аппарат;
— провести анализ предметной области методов обучения с подкреплением для задач игрового искусственного интеллекта;
— сформулировать способы классификации методов;
— классифицировать методы исходя из способов;
— сравнить описанные алгоритмы;
— отразить результаты сравнения и классификации в выводе.

1 Анализ предметной области

Для начала рассмотрим игры, которые будут решаться в данной работе. Сначала даны их виды, рассмотрены проблемы связанные с ними, а затем представлены описания игр. Все игры, рассматриваемые в работе сводятся к Марковским играм (1.4.2), они подробно описаны в конце. Таким образом, предметную область искусственного интеллекта в играх формализуется с помощью математических моделей. Ниже приведена формализация предметной области.

Методика выполнения НИР состоит в выявлении критериев применимости методов решения задач, а также в сравнении этих методов.

1.1 Виды игр

В работе рассматриваются три вида игр. Опишем каждый их видов подробнее.

1.1.1 Кооперативные игры

В полностью кооперативных играх все агенты разделяют общую функцию наград. Такие игры так же называются MDP с несколькими агентами (MMDP). При данном условии, все Q–функции и V–функции агентов совпадают. Таким образом, оптимальная стратегия для каждого агента является совместной стратегией. Можно использовать Q–обучение для обучения совместной стратегии. эквилибриум Нэша достигается.

Альтернативным подходом является использование функции наград для каждого агента в отдельности, награда команды определяется как среднее значение награды каждого агента. Подобный подход подразумевает обучение агентов независимо друг от друга (decentralized MARL), а также использование протокола коммуникации между агентами. Подразумевается, что агенты изобретут схему коммуникации (язык), которая позволит им достичь оптимальной совместной стратегии.

1.1.2 Соревновательные игры

Полностью кооперативные игры обычно моделируются как игры с нулевой сумой, то есть $\sum_{i \in \mathcal{N}} R^i(s, a, s') = 0$. Большинство исследований в области соревновательных игр рассматривают среду с двумя агентами. эквилибриум Нэша описывает стратегию оптимизирующую худшую награду в долгосрочном периоде.

1.1.3 Смешанные игры

Смешанные игры также известны как игры с общей суммой (general sum games). В таких играх сумма наград всех агентов может быть любой, как и отношения между агентами.

1.2 Проблемы применимости к предметной области

Применяя методы обучения с подкреплением к игровому искусственному интеллекту, выявляются следующие проблемы:

— комбинаторная сложность — размер пространства действий растет экспоненциально с увеличением числа агентов;

— оптимизируемые величины многомерны — эффективность обучения агентов нельзя описать одной метрикой;
— проблема нестабильности — агенты, которые являются частью среды, обучаются независимо: динамика среды меняется.
— редкая награда — награда может приходить только в конце матча.

1.3 Рассматриваемые игры

Рассматриваемые игры являются смешанными. Были рассмотрены следующие игры:

— Проблемы повторяющихся матриц (Matrix Games) — игры, награды которых описываются матрицами. Они усложнены тем, что существует локальный минимум на пути к эквилибриуму;
— Частицы с несколькими агентами (MPE) 1.1 — несколько двумерных проблем навигации: Штурман-Искатель, Разносчик, Советчик, Жертва-Хищник;
— StarCraft (SMAC) — сценарий компьютерной игры StarCraft с несколькими агентами;
— Level Based Foraging (LBF) — агенты должны собрать еду на карте, чтобы выжить;
— Robotic Warehouse (RWARE) — агенты должны донести предметы до пункта назначения, а потом вернуться в начало.

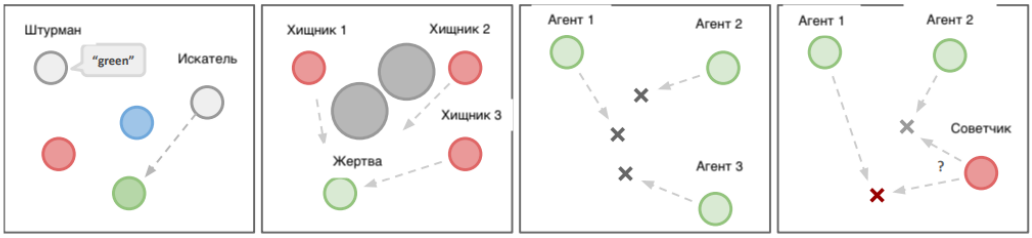


Рисунок 1.1 – Среды MPE, слева–направо: Штурман говорит Искателю к какой точке идти, Жертва скрывается от Хищников в серое убежище, Разносчики идут по разным местам, Советчик координирует Разносчиков.

Таблицы 1.1 и 1.2 сравнивают игры по разным критериям.

Таблица 1.1 – Сравнение игр по наблюдаемости и количеству наград

Игра	Наблюдаемость	Кол–во наград
Матричные игры	Полная	Много
MPE	Полная / Неполная	Много
SMAC	Полная / Неполная	Много
LBF	Полная / Неполная	Крайне мало
RWARE	Полная / Неполная	Крайне мало

Таблица 1.2 – Сравнение игр по сложности и количеству агентов

Игра	# агентов	Главная сложность
Матричные игры	2	Не оптимальный эквилибриум
MPE	2–3	Нестационарность
SMAC	2–10	Большое пространство действий
LBF	2–4	Координирование
RWARE	2–4	Крайне мало наград

1.4 Формализация

Изложенные выше игры формализуются математическим языком согласно [1] с помощью понятия Марковских игр. Ниже представлены описания некоторых важных функций, используемых для решения игр. Решение подразумевает поиск оптимальной стратегии. Определение оптимальной стратегии использует эти функции.

В последующих разделах под функцией будет подразумеваться нейросеть.

1.4.1 Марковский процесс принятия решений

Для среды с одним агентом Марковский процесс принятия решений (MDP) задается кортежем $(\mathcal{S}, \mathcal{A}, \mathcal{P}, R, \gamma)$, элементы которого определяются следующим образом:

— $\mathcal{S}$ — множество состояний среды;

— $\mathcal{A}$ — множество действий агента;

— $\mathcal{P} : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \mapsto [0, 1]$ — вероятность перехода из состояния $s \in \mathcal{S}$ в состояние $s' \in \mathcal{S}$ при выполнении действия $a \in \mathcal{A}$;

— $R : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \mapsto R$[1] — награда за совершение действия $a \in \mathcal{A}$ в состоянии $s \in \mathcal{S}$ и переход в состояние $s' \in \mathcal{S}$;

— $\gamma \in [0, 1]$ — коэффициент дисконтирования, влияет на вероятность предпочтения немедленной награды награде в будущем.

На каждом шаге агент наблюдает состояние среды s_t в момент времени t и принимает действие a_t

Решением задачи является стратегия π, которая определяется следующим образом[1]:

$$\pi(s) = \arg\max_{a \in \mathcal{A}} \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t R(s_t, a_t, s_{t+1}) \mid a_t \sim \pi(\cdot \mid s_t), s_0\right]. \tag{1.1}$$

Определим Q-функцию[2] и V-функцию значений[3] как:

$$Q^{\pi}(s, a) = \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t R(s_t, a_t, s_{t+1}) \mid a_t \sim \pi(\cdot \mid s_t), s_0 = s, a_0 = a\right], \tag{1.2}$$

[1]Стратегия зависит от начального состояния s_0 и от предыдущих принятых действий (т. е. от самой себя). Стратегия максимизирует ожидаемую награду, которая вычисляется по формуле Беллмана.

[2]Какую награду можно ожидать, если на шаге t выполним действие a, а дальше будем придерживаться стратегии?

[3]Какую награду можно ожидать, если будем придерживаться лучших действий согласно стратегии?

$$V^{\pi}(s) = \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t R(s_t, a_t, s_{t+1}) \mid a_t \sim \pi(\cdot \mid s_t), s_0 = s\right]. \tag{1.3}$$

Полным решением проблемы является оптимальная стратегия π^*. В большинстве случаев найти оптимальную стратегию невозможно, и приближенное решение удовлетворительно. Одним из таких приближенных решений является метод итеративного улучшения стратегии (policy iteration).

1.4.2 Марковские игры

Одним из обобщений Марковского процесса принятия решений (MDP) являются Марковские игры (MG). Также используется термин стохастические игры.

Марковская игра определена кортежем $(\mathcal{N}, \mathcal{S}, \{\mathcal{A}^i\}_{i\in\mathcal{N}}, \mathcal{P}, \{R^i\}_{i\in\mathcal{N}}, \gamma)$.

Большинство определений сохраняются из Марковского процесса принятия решений (1.4.1), новые определения следующие:

— $\mathcal{N}$ — множество агентов;

— $\mathcal{S}$ — множество состояний;

— $\mathcal{A}^i$ — множество действий агента i;

— $\mathcal{P}$ — множество вероятностей перехода;

— R^i — функция награды агента i.

Целью агента i является оптимизация его награды в долгосрочном периоде, путем нахождения оптимальной стратегии $\pi^i : \mathcal{S} \mapsto \Delta(\mathcal{A}^i)$[4].

В контексте нескольких агентов, многие величины становятся многомерными. Введем еще несколько определений:

— $s = (s^1, \ \ldots \ , s^n)$ — состояния, в котором находятся агенты;

— $a = (a^1, \ \ldots \ , a^n)$ — действия, принимаемые агентами в состоянии s;

— $\pi : S \mapsto \Delta A$ — совместная стратегия.

[4] Δ означает, что пространство действий поменяется в новом состоянии.

В частности, определим функцию $V^i_{\pi^i,\pi^{-i}}$:

$$V^i_{\pi^i,\pi^{-i}}(s) = \mathbb{E}_{\pi^i,\pi^{-i}}\left[\sum_{t=0}^{\infty}\gamma^t R^i(s_t,a_t,s_{t+1}) \mid a^i_t \sim \pi^i(\cdot|s_t), s_0 = s\right], \tag{1.4}$$

где i означает множество всех агентов кроме i.

Таким образом концепт решения MG отличается от MDP тем, что оптимальная стратегия каждого агента зависит от стратегий всех остальных агентов.

Одним из решений марковских игр является эквилибриум Нэша. эквилибриум Нэша это совместная стратегия $\pi^* = (\pi^{1,*}, \ldots, \pi^{N,*})$ такая, что для каждой $s \in \mathcal{S}$ и $i \in \mathcal{N}$:

$$V^i_{\pi^{i,*},\pi^{-i,*}}(s) \geq V^i_{\pi^i,\pi^{-i,*}}(s),\ \forall\ \pi^i. \tag{1.5}$$

Эквилибриум Нэша характеризует точку равновесия π^*, в которой каждому агенту не выгодно изменить свою стратегию. Иными словами, для любого агента i стратегия $\pi^{i,*}$ является оптимальной стратегией в условиях, когда все остальные агенты играют по стратегии $\pi^{-i,*}$.

Доказано, что существует хотя бы один эквилибриум Нэша для Марковской игры с конечным числом состояний и конечным горизонтом (числом шагов до окончания игры).

Большинство рассматриваемых алгоритмов сходятся к точке эквилибриума Нэша.

1.4.3 Описание задачи

Задачей является поиск этой оптимальной стратегии для агента, описанной выше.

Агенту подается на вход состояние среды (той части, которую он видит), а также награда за предыдущее действие. На выходе агент должен выдать действие, которое он собирается совершить в данном состоянии. Действие может выражаться дискретной величиной, например, направлением движения, или непрерывной величиной, например, углом поворота.

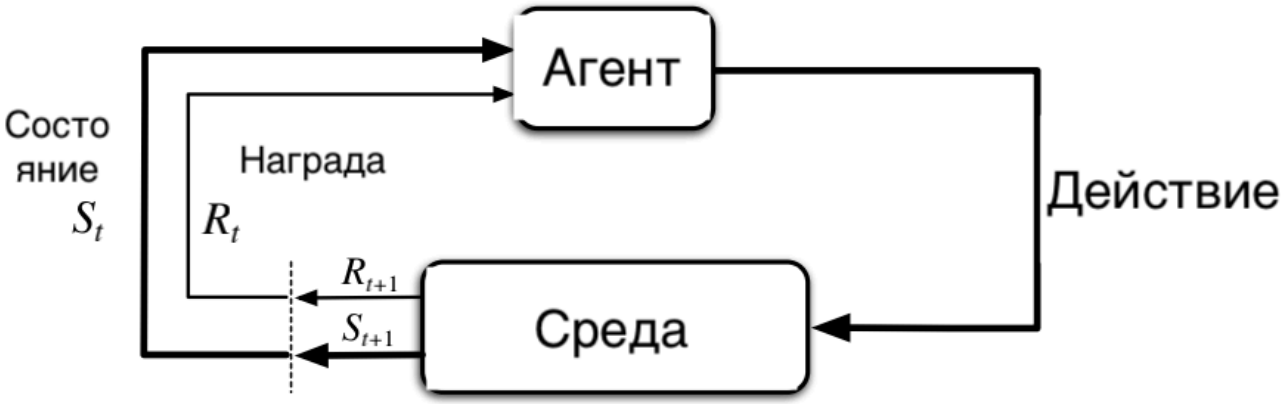


Рисунок 1.2 – Марковский процесс принятия решений

Если у агентов одинаковые возможности (допустим они игроки онлайн шутера), можно использовать среднюю награду за игру, как метрику качества стратегии. Более обобщенной метрикой служит процент побед.

Могут быть введены следующие ограничения:

— количество шагов в игре — горизонт;
— смерть агента — умершему агенту поступают в наблюдения нулевые векторы, а действия игнорируются;
— допустимость действия — недопустимые дискретные действия маскируются и не могут быть выполнены;
— наблюдаемость среды — агент может наблюдать лишь доступную ему часть среды;
— возможность коммуникации — агенты могут или не могут обмениваться информацией.

Для недопущения действий, выражаемых векторами, существует дополнительный класс алгоритмов называющихся безопасными алгоритмами обучения с подкреплением. Их рассмотрение выходит за рамки данной работы.

Вывод

Был произведен анализ предметной области, выделены проблемы игрового искусственного интеллекта, задача формализована с использованием Марковских игр.

2 Описание алгоритмов

В данной главе будут рассмотрены самые популярные алгоритмы обучения с подкреплением. Популярность была определена по наличию программной реализации алгоритма в свободном доступе.

2.1 Выбор алгоритмов для исследования

В ходе проведенного анализа репозиториев и библиотек ([2], [3], [4]) были выделены следующие алгоритмы:

- IQL — независимое Q–обучение [5];
- VDN — сети декомпозиции V–функции [6];
- QMIX — факторизация монотонной Q–функции [7];
- MAVEN —- несколько агентов с вариационным исследованием [8];
- MADDPG — алгоритм DDPG [9] с несколькими агентами [10];
- MAPPO — удивительная эффективность PPO [11] в среде с несколькими агентами [12];
- IPPO — обычный алгоритм PPO из обучения с подкреплением в среде с одним агентом.

2.2 IQL

Алгоритм строится на основе Deep Q Learning [13], для каждый агент контролируется отдельной сетью. Агенты играют независимо друг от друга, но видят одну и ту же среду, и действия друг друга. В качестве среды был выбран модифицированный эмулятор игровой консоли Atari. На примере игры в понг протестированы среды, подразумевающие как коопера-

цию, так и конкуренцию. При кооперации, агенты удерживали мяч на поле наибольшее время. К недостаткам алгоритма можно отнести отсутствие теоретической гарантии достижения эквилибриума [1].

1) дискретные действия;
2) алгоритм для смешанных игр;
3) off–policy;
4) как во время обучения, так и во время тестирования агенты децентрализованы.

В системе реального мира, где агенты только децентрализованы, такой подход может оказаться единственным возможным.

2.3 VDN

VDN [6] пытаются обучить совместную Q–функцию:

$$Q_{tot}(s, a) = \sum i \in \mathcal{N} Q_i(s^i, u^i;\ \theta^i). \tag{2.1}$$

Она отличается от IQL тем, что Q–функциям для каждого агента не поступает информация о состоянии и действиях другого агента. Таким образом достигается большая автономность агентов. Метод уступает по всем метрикам методу, описанному ниже, и по этой причине не используется в сравнениях. Тем не менее, он обладает достаточной методической значимостью, чтобы быть рассмотренным.

Таким образом можно классифицировать алгоритм следующими критериями:

1) дискретные действия;
2) алгоритм для смешанных игр;
3) off–policy;
4) во время обучения агенты централизованны, а во время тестирования децентрализованы;

2.4 QMIX

QMIX строится на базе VDN, но вместо суммы используется факторизация монотонной Q–функции по ограничению:

$$\frac{\partial Q_{tot}}{\partial Q_i} \geq 0, \forall i \in \mathcal{N}. \tag{2.2}$$

Чтобы обеспечить условие выше, QMIX использует 2 дополнительных сети: mixing network, hyper network. Авторы алгоритма демонстрируют его превосходство над VDN на примере кооперативной игры с суммой матрицы. VQN не удается достичь эквилибриума.

Таким образом можно классифицировать алгоритм следующими критериями:

1) дискретные действия;
2) алгоритм для смешанных игр;
3) off–policy;
4) во время обучения агенты централизованны, а во время тестирования децентрализованы;

Алгоритм является улучшением VDN.

2.5 MAVEN

Авторы MAVEN замечают, что ограничения, полагаемые QMIX ведут к плохой стратегии исследования среды, как результат к худшей производительности [8]. MAVEN вводит новый подход совмещающий методы обучения с использованием V-функции и методы с использованием стратегии. Они вводят общее латентное пространство, управляемое иерархической стратегией.

Они предлагают использовать ансамбль генеративных стратегий для управления Q_i, в то же время гарантируя монотонность Q_i по Q_{tot}. Таким образом достигается консистентное по времени исследования среды.

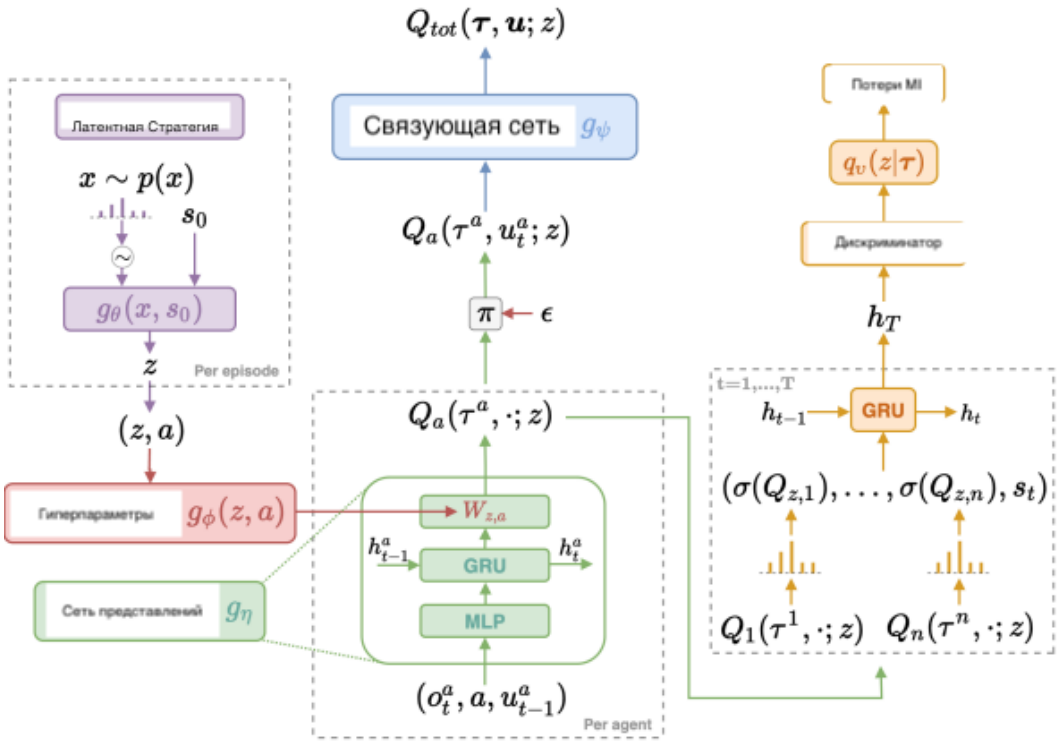


Рисунок 2.1 – Архитектура MAVEN

Таким образом можно классифицировать алгоритм следующими критериями:

1) дискретные действия;
2) алгоритм для смешанных игр;
3) off–policy;
4) во время обучения агенты централизованны, а во время тестирования децентрализованы;

Алгоритм является улучшением QMIX и VDN.

2.6 Традиционные алгоритмы из обучения с подкреплением

В данной секции рассказывается про традиционные алгоритмы. Среди них PPO, берущий за основу метод Actor-Critic. В контексте нескольких агентов, метод получил название IPPO (независимый PPO) [12].

Следующие алгоритмы следуют парадигме «централизованное обу-

чение и децентрализованная игра» (centralized learning and decentralized execution
Это классические алгоритмы DDPG и PPO, оснащенные центральной Q-сетью. Они названы MADDPG и MAPPO соответственно [9]. В сравнениях ниже также представлен MAA2C, который является адаптацией A2C для нескольких агентов. IA2C - это независимая версия A2C.

Классификация IPPO может быть сделана по следующим критериям:

1) смешанные действия;
2) алгоритм для смешанных игр;
3) on–policy;
4) полная децентрализация;

Метод IPPO подходит для сценариев, когда не хранится буфер наблюдений среды, а агенты обучаются по мере игры. Также стоит отметить его простоту реализации, и факт полной децентрализации.

Классификация других методов может быть сделана по следующим критериям:

1) смешанные действия;
2) алгоритм для смешанных игр;
3) централизованное обучение и децентрализованная тестирование;

Вывод

Был проведен анализ релевантности алгоритмов, выделены популярные алгоритмы для анализа в данной работе, приведено их краткое описание, основные свойства, классификация.

3 Классификация алгоритмов

Способы классификации алгоритмов были взяты из [1]. Алгоритмы, рассмотренные ниже, не были полностью классифицированы в статье. Классификация алгоритмов MARL зависит от типа проблемы, а не от подхода к решению, как в области обучения с подкреплением.

3.1 Классификация согласно теории обучения с подкреплением

Классификация по типу действий в среде:

— дискретное действие - прим: Atari дискретными кнопками;

— непрерывное действие - прим: Doom с аналоговым вводом мышки.

По необходимости непосредственного взаимодействия агента со средой:

— необходимо непосредственное взаимодействие агента - On Policy;

— может учиться по записям игр - Off Policy.

Таблица 3.1 показывает классификацию алгоритмов MARL по необходимости непосредственного взаимодействия агента со средой, по типам обучения.

Таблица 3.1 – Классификация алгоритмов MARL по необходимости непосредственного взаимодействия агента со средой, по типам обучения

Алгоритм	Центр. Обучение	On/Off Policy	Q–обучение	Учит стратегию
IQL	x	Off	✓	x
VDN	✓	Off	✓	x
QMIX	✓	Off	✓	x
MAVEN	✓	Off	✓	x
MADDPG	✓	Off	✓	✓
IPPO	x	On	✓	✓
MAPPO	✓	On	✓	✓

Алгоритмы, которые не обучают стратегию, не могут быть применены к играм с континуальным пространством действий.

3.2 Классификация по типу игры

Типы игр были описаны ранее. Приведем их здесь для ссылки:

— кооперативные игры (Cooperative Games);
— соревновательные игры (Competitive Games);
— смешанные игры (Mixed Games).

Рассматриваемые алгоритмы достаточно общие и могут быть применены ко всем типам игр. Тем не менее, существуют другие, не рассматриваемые алгоритмы, которые специализируются на определенных типах игр.

3.3 Классификация по парадигме обучения

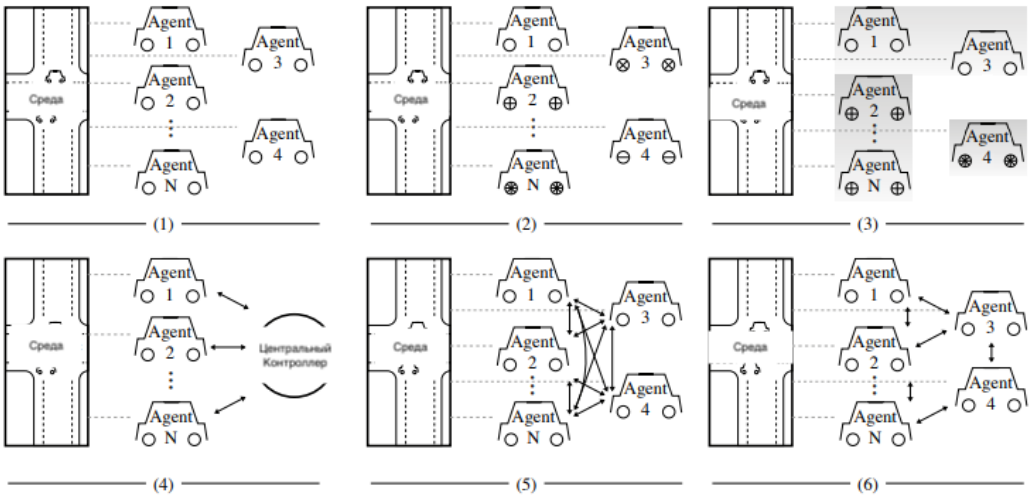


Рисунок 3.1 – Классификация по парадигме обучения

Различают следующие парадигмы обучения [1]:

1) независимые агенты с общей стратегией;
2) независимые агенты с независимой стратегией;
3) независимые агенты с общей стратегией внутри команды;
4) один контроллер всех агентов;
5) во время обучения агенты централизованны, а во время тестирования децентрализованы;
6) во время обучения агенты децентрализованы, но могут обмениваться сообщениями в сети, во время тестирования децентрализованы.

Таблица 3.2 содержит классификацию алгоритмов по парадигме обучения.

Таблица 3.2 – Классификация по парадигме обучения

Алгоритм	Парадигма
IQL	2
VDN	2
QMIX	2
MAVEN	2
MADDPG	5
IPPO	1 или 2
MAPPO	5

Вывод

Были предложены методы классификации алгоритмов, алгоритмы классифицированы согласно предложенным методам.

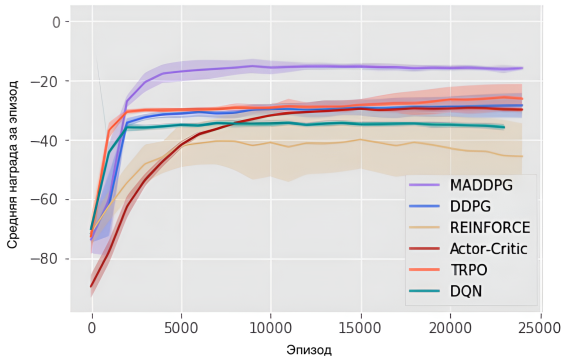
Существует множество алгоритмов обучения агентов в среде с несколькими агентами. Следует помнить что оптимальная стратегии для одного агента может быть не оптимальной для нескольких агентов в совокупности. Не все рассматриваемые алгоритмы гарантируют сходимость к оптимальной стратегии, то есть предоставляют теоретическую гараантию решения проблемы.

Существуют другие дисциплины обучения с подкреплением, например обратное обучение, когда агенту не дают награды. В дальнейших работах было бы интересно рассмотреть эти алгоритмы.

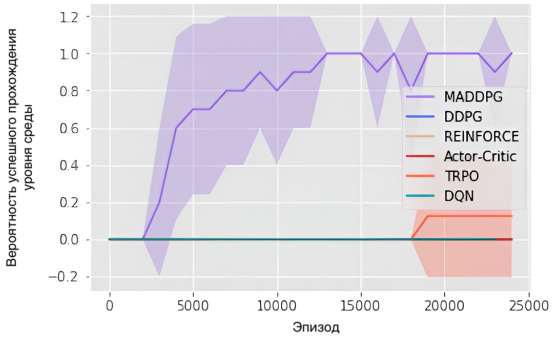
4 Сравнение производительности алгоритмов

В теории игр сравнивать разные алгоритмы достаточно просто - можно посмотреть на награды, полученные алгоритмами за эпизод. Дополнительных метрик и критериев для сравнения не требуется.

Для начала убедимся в необходимости использовать специальные алгоритмы для сред с несколькими агентами. На рисунке 4.1 [9] представлены награды за эпизод следующей игры: один агент говорит другому точку, в которую нужно бежать, а другой агент бежит в эту точку (на время). Видно, что классические алгоритмы не способны понять необходимость кооперации.



(a) Средняя награда за эпизод



(b) Процент ситуаций, когда агент достигает цели

Рисунок 4.1 – Сравнение производительности MARL алгоритма MADDPG и классических RL алгоритмов

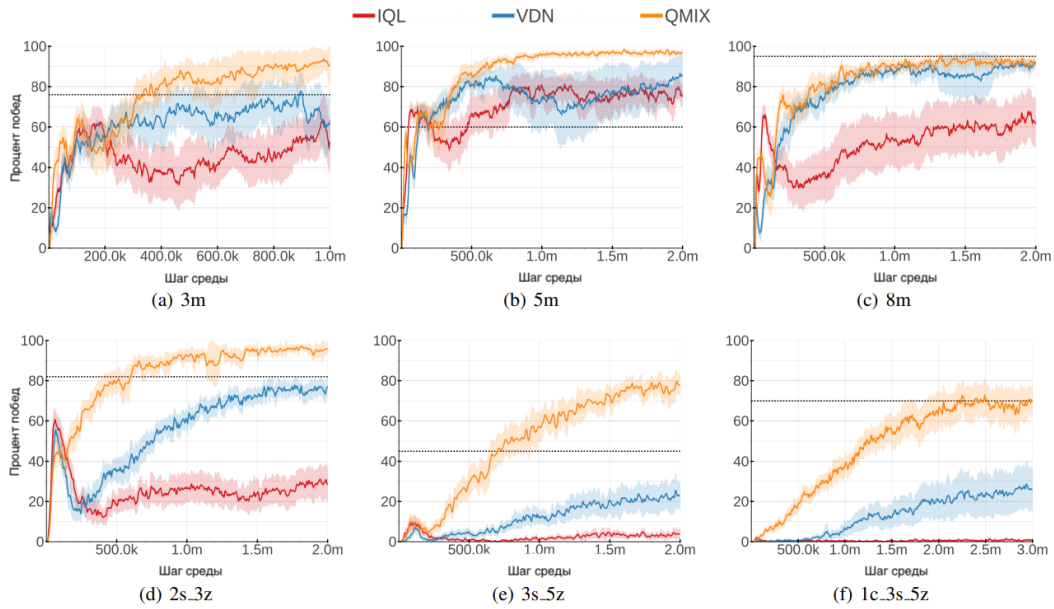


Рисунок 4.2 – Сравнение процента побед QMIX, VDN, IQL в среде StarCraft II в зависимости от кол–ва эпизодов для обучения

Алгоритмы COMA, QTRAN очень похожи на QMIX, и их рассмотрение избыточно. Таким образом, они показывают практически нулевой результат, по сравнению рассмотренными алгоритмами на 4.3 [8].

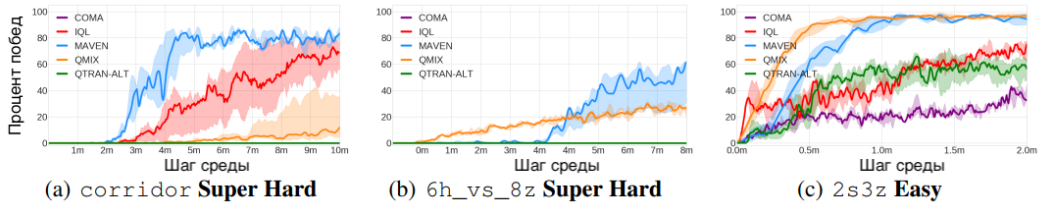


Рисунок 4.3 – Сравнение процента побед MAVEN, QMIX, IQL, COMA, QTRAN, в среде StarCraft II в зависимости от кол–ва эпизодов для обучения

Видно, что MAVEN и QMIX лидируют во всех категориях. Особенно хорошо использовать MAVEN в среде SMAC(StarCraft II), т. к. пространство действий в ней очень большое и необходимо его консистентно исследовать [8].

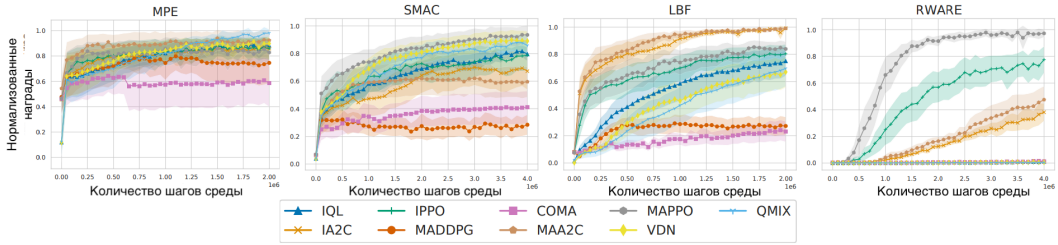


Рисунок 4.4 – Нормализованное вознаграждение разных алгоритмов в на разных средах в зависимости от кол–ва эпизодов для обучения

4.4 [12] включает в себя все рассматриваемые игры. Можно сделать вывод, что MAPPO показывает лучшие результаты в большинстве сред. IPPO, не включающий в себя методы обучения нескольких агентов, показывает хорошие результаты в большинстве сред. Это было подмечено в статье [12].

Модификации классических алгоритмов для среды с несколькими агентами показывают средние результаты, за исключением среды IBF где A2C (как обычный вариант, так и с централизованным обучением Q) показывает наилучшие результаты.

Вывод

Была предложена метрика сравнения алгоритмов (по средней награде в игре, по проценту побед, по нормализованной средней награде), приведены графики сравнения алгоритмов по выбранной метрике. Даны комментарии к графикам, советы по выбору того или иного алгоритма. Можно считать, что обсуждаемые результаты достоверны, исходя из открытых инструментов для сравнения.

ЗАКЛЮЧЕНИЕ

В ходе выполнения данной работы были выполнены следующие задачи:

— задача была формализована используя игры маркова;
— был проведден анализ предметной области;
— сформулированы способы классификации и сравнения методов;
— методы классифицированы по предложенным способам;
— проведено сравнение алгоритмов;
— результат сравнения алгоритмов отражен в выводе.

Современные алгоритмы классического обучения (IPPO, IA2C) с подкреплением показывают хорошие результаты в среде с несколькими агентами. Тем не менее, использование централизованной Q–сети улучшает результаты во всех случаях. Подобные алгориты, основанные а методе Actor-Critic уместнее применять к средам с континуальным действием.

В среде с дискретным действием, таких как данная, лучше использовать алгоритмы без обучения стрегении (Policy Learning), по типу MAVEN, IQL.

Если среда обладает большим пространством действий, то лучше использовать MAVEN, т. к. он гарантирует консистентное исследование среды. Он также показывает лучшую сходимость в других средах.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1 Y. Yang. An Overview of Multi-Agent Reinforcement Learning from Game Theoretical Perspective / Yang Y., Wang J. // CoRR. – 2020. – abs/2011.00583.

2 The StarCraft Multi-Agent Challenge / Samvelyan M. [и др.] // CoRR. – 2019. – 1902.04043.

3 Marl-Algorithms [Электронный Ресурс]. Режим доступа: `https://github.com/starry-sky6688/MARL-Algorithms` (дата обращения: 01.10.2022).

4 Awesome Multiagent Learning [Электронный Ресурс]. Режим доступа: `https://github.com/chuangyc/awesome-multiagent-learning` (дата обращения: 01.10.2022).

5 Multiagent Cooperation and Competition with Deep Reinforcement Learning / Tampuu A. [и др.] // PLoS ONE. – 2015. – 12.

6 Value-Decomposition Networks For Cooperative Multi-Agent Learning / Sunehag P. [и др.] // Adaptive Agents and Multi-Agent Systems. – 2017.

7 QMIX: Monotonic Value Function Factorisation for Deep Multi-Agent Reinforcement Learning / Rashid T. [и др.] // J. Mach. Learn. Res. – 2020. – 21. – 178:1-178:51.

8 MAVEN: Multi-Agent Variational Exploration / Mahajan A. [и др.] // Neural Information Processing Systems (NIPS). – 2019.

9 Continuous control with deep reinforcement learning [Электронный Ресурс]. Режим доступа: https://arxiv.org/abs/1509.02971 (дата обращения: 01.10.2022).

10 Multi-Agent actor-Critic for Mixed Cooperative-Competitive Environments / Lowe R. [и др.] // Neural Information Processing Systems (NIPS). – 2017.

11 Proximal Policy Optimization Algorithms / Schulman J. [и др.] // CoRR. – 2017. – abs/1707.06347.

12 The Surprising Effectiveness of MAPPO in Cooperative, Multi-Agent Games / Yu C. [и др.] // CoRR. – 2021. – abs/2103.01955.

13 Playing Atari with Deep Reinforcement Learning / Mnih V. [и др.] // ArXiv. – 2013. – abs/1312.5602.

ПРИЛОЖЕНИЕ А

Презентация содержит 16 слайдов.