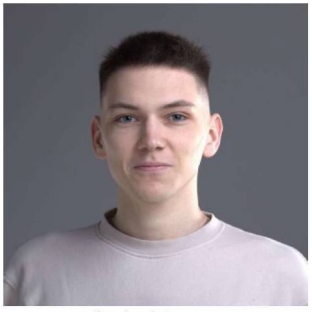
УДК 004.8

ВЛИЯНИЕ МЕТОДОВ ОБУЧЕНИЯ С ПОДКРЕПЛЕНИЕМ НА СТРАТЕГИЮ ПОВЕДЕНИЯ АВТОНОМНЫХ АГЕНТОВ



А.А. Гулис
Студент факультета компьютерных систем и сетей БГУИР
antongulis43@gmail.com

М.А. Калугина
Доцент кафедры информатики, кандидат физико-математических наук, доцент
marina_kalugina@list.ru

А.А. Гулис

В 2022 году окончил Лицей города Ивацевичи. Интересы направлены на глубокое обучение, обучение с подкреплением в сложных средах, компьютерное зрение, а также генеративные нейросетевые модели для синтеза 2D- и 3D-контента.

М.А. Калугина

Окончила Белорусский государственный университет. Область научных интересов связана с исследованием проблем метрической теории диофантовых приближений зависимых величин и приложений математических методов к нейросетевому анализу

Аннотация. В статье приведены результаты исследования влияния различных архитектур и алгоритмов глубокого обучения с подкреплением (Deep Reinforcement Learning, DRL) на формирование

стратегий поведения автономных агентов. В качестве испытательной среды используется актуальный клиент многопользовательской игры Tanks Blitz. Рассматривается мультиагентное кооперативно-соревновательное взаимодействие в формате «2 на 2» с использованием парадигмы обучения агента против собственных исторических версий (Self-Play). Обучение в данной среде осложнено частичной наблюдаемостью и ограничениями пропускной способности. Предложена распределенная архитектура сбора данных и проанализированы алгоритмы PPO, Discrete SAC (DSAC), Stable DSAC (SDSAC), R2D2 и гибридная модель Hybrid SAC. Выявлены феномены «взлома вознаграждения» (reward hacking), взрыва температурного коэффициента, коллапса энтропии, а также предложены конкретные методы их устранения.

Ключевые слова: глубокое обучение с подкреплением, self-play, PPO, SAC, DSAC, SDSAC, Hybrid SAC, формирование вознаграждения.

Введение. Методы глубокого обучения с подкреплением (RL) зарекомендовали себя как эффективный инструмент для создания автономных агентов – автономных систем принятия решений –, способных действовать в сложных динамических средах [1]. В отличие от задач классификации, где нейронная сеть обучается на заранее размеченном наборе данных, в RL агент самостоятельно генерирует данные путем проб и ошибок, получая от среды скалярный сигнал – вознаграждение.

Целью данного исследования является сравнительный анализ того, как выбор математического аппарата алгоритма (On-policy, Off-policy, рекуррентные и гибридные сети) влияет на итоговую стратегию поведения агента, управляющего боевой машиной в формате «2 на 2».

Ключевой особенностью работы является отказ от использования упрощенных симуляторов: взаимодействие происходит напрямую с актуальным игровым клиентом Tanks Blitz через специально разработанный WebSocket-интерфейс. Это накладывает ряд критических ограничений:

1 Низкая пропускная способность. Взаимодействие с 3D-клиентом ресурсоемко. На высокопроизводительной системе (CPU 24 ядра, GPU RTX 4080 Super, 64 GB RAM) параллельно запускается около 30 экземпляров игры (в зависимости от выбранного алгоритма), что в сумме дает генерацию лишь около 300 кадров (шагов) в секунду.

2 Разреженная награда. Терминальным победным состоянием является полное уничтожение команды противника. Вероятность случайного выполнения этой задачи стремится к нулю, поэтому на ранних этапах обучения нейронная сеть не получает значимого обучающего сигнала.

3 Частичная наблюдаемость (POMDP). Агенту доступна лишь локальная информация. В игре реализованы механики циклов перезарядки и «тумана войны» (противник не передается клиенту, если находится вне радиуса радиосвязи или за укрытием), что нарушает марковское свойство среды, требуя от сети наличия памяти.

Описание среды и распределенной архитектуры. Агенты внутри одной команды обучаются с использованием парадигмы разделения параметров: оба танка управляются одной и той же нейронной сетью, однако получают индивидуальные векторы наблюдений и формируют независимые действия [2].

На начальных этапах исследования в качестве входных данных использовались пиксельные изображения (кадры из игры), обрабатываемые сверточными сетями (CNN). Однако для повышения скорости сходимости был осуществлен переход к векторным представлениям. Для обеспечения свойства трансляционной инвариантности (независимости поведения от абсолютных координат на карте) вектор состояния строится исключительно на относительной геометрии и статусах модулей.

Формируемый вектор включает:

1 Сложную геометрию целей: расчет дистанций, нормализованных векторов направлений (орудия, камеры, корпуса) и их косинусных расстояний. Дополнительно с помощью матриц поворота вычисляются проекции направлений агента на границы корпуса и башни противника.

2 Статус модулей и орудия: здоровье внутренних модулей агента и противников (гусеницы, двигатель, боеукладка, механизм поворота башни), вероятность пробития брони, текущий разброс орудия и статус прицеливания (в танк, в землю или в небо).

3 Сенсоры препятствий: массив виртуальных лучей, сканирующих пространство вокруг танка на предмет укрытий и препятствий. Лучи запускаются на ограниченную дистанцию – 50 метров. Всего испускается 8 лучей: вперед, назад, влево, вправо и в промежуточные направления.

4 Историю действий: к вектору состояния конкатенируется *one-hot* кодировка предыдущего дискретного действия, а при использовании гибридных политик – еще и маскированные смещения.

Для обеспечения корректной работы с актуальным клиентом игры необходимо исключить отправку физически невозможных команд (например, выстрел в момент перезарядки). Среда возвращает бинарную маску допустимых действий. На стороне алгоритма реализовано маскирование: логиты (сырые выходы нейросети) недопустимых действий заменяются на минимально возможное значение перед вычислением функции вероятности Softmax. Это математически корректно обнуляет вероятность выбора запрещенного действия, не искажая расчет энтропии в функции потерь.

Для непрерывного обучения спроектирована асинхронная архитектура Actor-Learner [3]. Множество процессов-сборщиков на CPU собирают переходы агентов в среде и отправляют их в очередь Redis. Центральный процесс на GPU непрерывно извлекает данные, вычисляет градиенты и обновляет веса (рисунок 1).

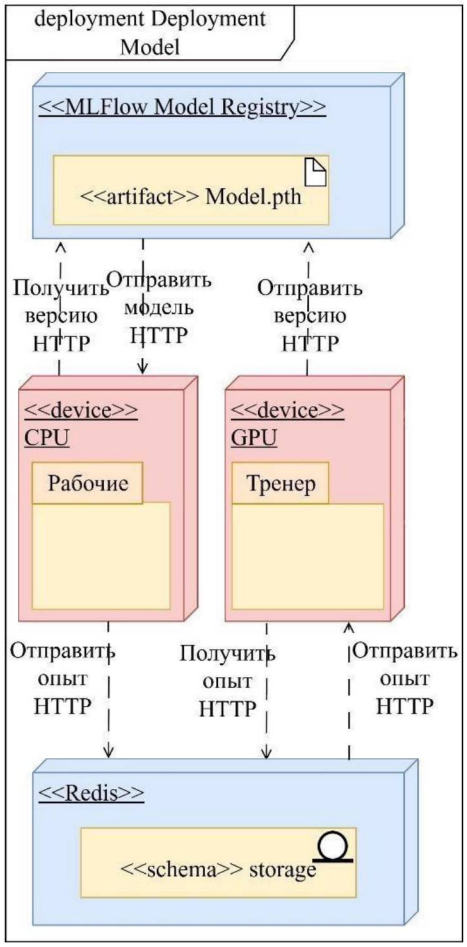


Рисунок 1. Диаграмма распределенной архитектуры

Для предотвращения эффекта «катастрофического забывания», когда агент выучивает узкую тактику против текущего оппонента, применяется смешанная выборка. Вражеская команда формируется по правилу: в 80% случаев загружаются самая последняя версия нейросети, а в 20% случаев случайным образом выбираются предыдущие версии.

Формирование стратегии через систему вознаграждений. Для обеспечения стабильного самообучения функция вознаграждения спроектирована по принципу игры с нулевой суммой (Zero-sum reward): любое положительное вознаграждение агента зеркально отражается штрафом для противника. Это предотвращает неформальную кооперацию агентов-оппонентов, когда агент может найти лазейку, увеличивая награды в одном состоянии, оставляя другого агента с нулевой или несопоставимой наградой.

Поскольку базовая награда крайне разрежена, был применен метод формирования вознаграждения на основе потенциалов (Potential-Based Reward Shaping, PBRS). Метод теоретически гарантирует, что добавление плотного сигнала не изменяет глобальную оптимальную политику [4].

Однако первоначальные эксперименты выявили классическую проблему «взлома вознаграждения» [5]. Если агент поощрялся исключительно за нанесенный урон, он находил локальный оптимум: поднимал камеру в небо, либо опускал орудие и стрелял в землю, так как противник зачастую находился рядом.

Для решения проблемы награда за урон была масштабирована в зависимости от дистанции до цели. Чем выше дистанция между агентом и танком противника в момент попадания, тем выше награда. Для лучшего усвоения новой системы наград было разработано обучение по расписанию (Curriculum Learning) [6]. Суть расписания заключалась в увеличении дистанции между командами по мере продвижения обучения. В начале, когда расстояние между агентами близкое, они изучали то, как ведет себя среда, какую они награду получают за свои действия, а затем, постепенно отдаляясь, начинали вырабатывать стратегии нападения и защиты.

Влияние алгоритмов на поведение. В ходе исследования был протестирован ряд современных DRL алгоритмов. Оценивалось их поведение в условиях низкой пропускной способности среды.

Первым был реализован On-policy алгоритм PPO [7]. Первоначальные эксперименты с алгоритмом проводились в упрощенной изолированной постановке: бои проходили в формате «1 на 1», при этом агенты были лишены возможности передвигаться. Танки появлялись в случайных позициях и на различных дистанциях друг относительно друга, управляя исключительно наведением орудия и стрельбой. В этой ограниченной среде PPO успешно и достаточно быстро решил задачу, выработав оптимальную политику точного прицеливания и быстрого уничтожения противника.

Успех в базовом статичном сценарии послужил мотивацией для перехода к полноценной и значительно более сложной среде – боям «2 на 2» с возможностью полномасштабного маневрирования. Для работы в этой среде исследовались две архитектуры сети: полносвязная и рекуррентная (LSTM). Эксперименты показали, что LSTM-архитектура справляется с задачей значительно лучше благодаря способности агрегировать исторический контекст (например, отслеживать перемещения противника за укрытиями).

Однако при масштабировании задачи алгоритм столкнулся со сложностями: агенты не могли выучить комплексное поведение на дальних дистанциях. Для решения этой проблемы была успешно внедрена стратегия Curriculum Learning. Изначальная дистанция между командами при появлении искусственно занижалась, а затем постепенно увеличивалась по мере выпуска новых, более обученных версий агента. Этот подход позволил PPO успешно и уверенно освоить среду 2v2, включая бои на дальних дистанциях. Впоследствии стратегия постепенного увеличения дистанции доказала свою эффективность и была интегрирована в обучение всех последующих алгоритмов.

Поскольку алгоритм обновляет веса только на свежесобранных данных, к его архитектуре были применены стабилизирующие практики [8]. Были исключены методы регуляризации, характерные для задач компьютерного зрения (такие как Batch Normalization или Dropout). Связано это с тем, что регуляризация должна предотвратить переобучение модели, когда она не может обобщить свои знания на данные, которых она не видела. Данная проблема не появляется в методах DRL. Также использовалась ортогональная инициализация весов, нормализация входных наблюдений и целевых значений с помощью бегущего среднего.

Критическим дополнением стала ранняя остановка обучения на эпохе: если KL-дивергенция (мера расхождения между старой и новой политикой) превышала порог 0.045, обновление прерывалось.

Благодаря Curriculum Learning алгоритм PPO успешно решил задачу 2v2 и оказался наиболее стабильным из всех исследованных методов. Однако выявились два критических недостатка:

1 Экстремальная чувствительность к подбору гиперпараметров.

2 Крайне низкая сэмпл-эффективность. Из-за On-policy природы алгоритма и низкой пропускной способности среды время сбора новых данных делало процесс обучения неприемлемо долгим.

Именно невозможность переиспользования старого опыта при жестких лимитах на генерацию новых кадров стала главной причиной отказа от масштабирования PPO и перехода к исследованию Off-policy методов.

Переход к алгоритму DSAC [9] потребовал реструктуризации пространства действий. В алгоритме PPO агенты использовали многомерное дискретное пространство, состоящее из пяти независимых ветвей: стрельба (2 варианта), поворот камеры по оси X (5 вариантов), по оси Y (3 варианта), движение шасси (3 варианта) и поворот корпуса (3 варианта). Архитектура PPO естественным образом поддерживает такой формат через несколько независимых выходных слоев (голов) нейросети. Однако классические дискретные Q-функции, на которых базируется DSAC, требуют оценки ценности для каждого возможного действия. Чтобы избежать создания сложной ветвящейся архитектуры Q-сети (критика), многомерное пространство было преобразовано в единое одномерное дискретное пространство, содержащее 270 уникальных комбинаций действий.

На этапе тестирования базового DSAC алгоритм продемонстрировал математическую нестабильность, вызванную спецификой функции вознаграждений:

1 Изначально была предпринята попытка использовать алгоритм нормализации целевых значений PopArt [10], который часто применяется в мультиагентных средах. Однако из-за высокой разреженности наград (подавляющее большинство шагов агента приносят нулевое вознаграждение) дисперсия в нормализаторе стремилась к нулю. Деление на околонулевую дисперсию вызывало градиентные взрывы и деградацию весов. От PopArt пришлось отказаться в пользу статического масштабирования (деления всех наград на константу).

2 В алгоритмах семейства SAC используется параметр, который динамически регулирует баланс между исследованием (максимизацией энтропии) и эксплуатацией (максимизацией награды) [11]. Механизм автоматической настройки вычисляет ошибку на основе разницы между текущей и целевой энтропией. В условиях нашей среды агенты получали нулевые награды на протяжении тысяч шагов. Следовательно, Q-значения оставались околонулевыми и не создавали градиентного «противовеса» для энтропии. В попытке заставить сеть исследовать среду (достичь целевой энтропии), оптимизатор бесконтрольно увеличивал значение регулятора до экстремальных величин (рисунок 2). Это приводило к тому, что политика становилась абсолютно случайной и теряла способность к какому-либо осмысленному поведению.

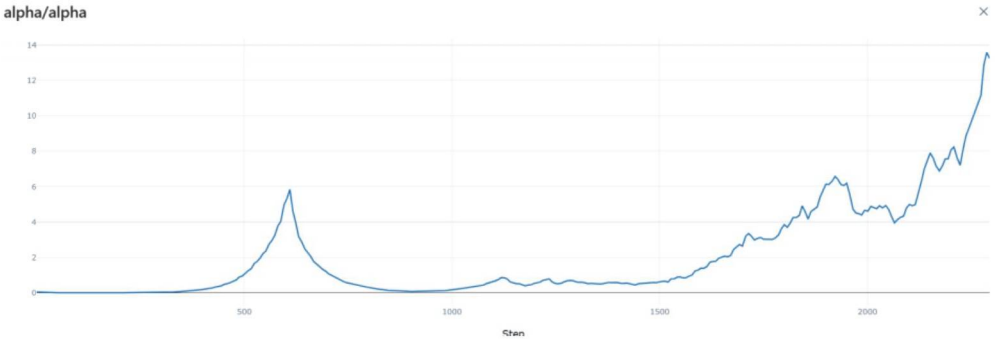


Рисунок 2. График параметра регуляции энтропии в алгоритме DSAC

Алгоритм удалось временно стабилизировать лишь путем экстремального снижения скорости обучения, однако скорость обучения оказалась неприемлемо низкой, а агент так и не смог выучить полезные тактики. Для решения проблем базового DSAC алгоритм был модернизирован до версии Stable DSAC (SDSAC) [12]. Основным архитектурным изменением стало вычисление целевого Q-значения. В классическом SAC используется минимум из двух критиков для предотвращения переоценки (overestimation). В дискретных пространствах это часто ведет к излишнему пессимизму. В SDSAC целевое значение вычислялось как среднее между критиками, что существенно стабилизировало процесс оценки состояний. Особое внимание было уделено горизонту предвидения – использованию n-step returns. Вместо обновления Q-функции на основе только одного следующего шага, алгоритм аккумулировал дисконтированную награду за несколько шагов вперед. Эксперименты проводились с параметрами n=3 и n=5.

Было установлено, что переход к n=5 обеспечивает значительно более плавное падение энтропии политики по мере обучения. В контексте обучения с подкреплением энтропия отражает степень неопределенности агента. Плавное и медленное падение энтропии означает, что агент дольше сохраняет высокий уровень исследования и медленнее переходит к эксплуатации. С одной стороны, это предотвращает преждевременную сходимость к субоптимальным тактикам. С другой стороны, это свидетельствует о значительном замедлении процесса обучения: агенту требуется гораздо больше времени и вычислительных мощностей, чтобы обрести уверенность в правильности своих действий и начать применять жесткие детерминированные стратегии поведения.

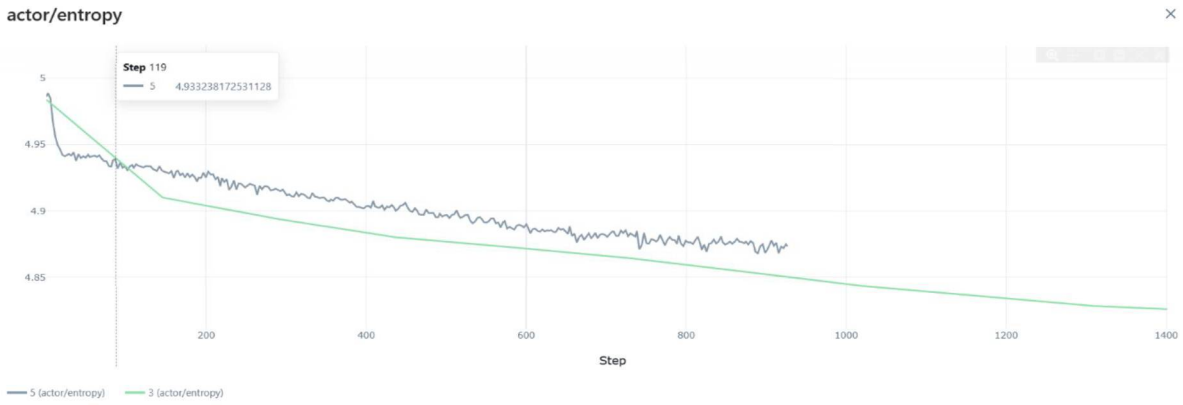


Рисунок 3. Сравнение динамики энтропии для параметра n-step

Благодаря этим модификациям SDSAC позволил агентам выработать базовые тактические паттерны, недоступные ранее. В частности, агенты освоили правильное

позиционирование – научились разворачивать лобовую (наиболее защищенную) проекцию брони к противнику перед выстрелом. Тем не менее, общая медлительность обучения (из-за затянутой фазы падения энтропии) послужила триггером для поиска еще более продвинутых архитектур. Для решения проблемы частичной наблюдаемости был внедрен алгоритм R2D2 [13], основанный на рекуррентной архитектуре. Для обучения сети исторические данные разбивались на фазу «прогрева» для инициализации скрытых состояний и фазу обучения. Наличие памяти (LSTM) позволило агенту запоминать положение противника. Агенты успешно научились набирать и удерживать дистанцию, максимизируя функцию вознаграждения. Недостатком стало появление артефактов поведения: агент начинал циклично повторять микро-движения, визуально «замедляясь» по сравнению с безрекуррентными моделями.

Наиболее перспективным направлением стала реализация алгоритма Hybrid SAC [14]. Суть подхода заключается в отказе от огромного дискретного пространства:

1 Управление шасси и стрельбой остается дискретным.

2 Управление башней переводится в непрерывное пространство, за которое отвечает отдельный выход сети.

Это радикально снизило размерность задачи, однако потребовало функции потерь сети стратегии (актора) с двумя независимыми регуляторами энтропии. Архитектура нейросети была перепроектирована и теперь включает два независимых выходных слоя: первый отвечает за выбор тактических маневров уклонения, а второй генерирует точные параметры для плавного сопровождения цели орудием. Для обеспечения надежности обучения дискретной части агента в гибридную модель были успешно интегрированы механизмы из ранее протестированного алгоритма SDSAC.

В процессе настройки была выявлена крайняя чувствительность алгоритма к гиперпараметрам. Во-первых, обучение критически зависит от выбора целевой энтропии: некорректное значение приводит к необратимой деградации температурного коэффициента, который сходится к нулю или уходит в максимум, полностью останавливая прогресс агента. Во-вторых, эффективность гибридного подхода сильно зависит от коэффициентов масштабирования камеры. При слишком малых значениях агент становится медлительным и почти неподвижным, поскольку любой маневр начинает требовать избыточного количества микро-действий. При завышенных масштабах предсказания сети перестают нести полезную нагрузку, так как они упираются в жесткие физические ограничения среды на скорость вращения башни. Тонкая калибровка этих параметров позволила раскрыть потенциал архитектуры. В отличие от полностью дискретных алгоритмов, где наведение происходило ступенчато и часто сбивалось при движении шасси, гибридная модель продемонстрировала максимально плавное и точное наведение, наиболее приближенное к действиям реального игрока.

Заключение. В рамках данного исследования была разработана и успешно внедрена сложная распределенная система обучения с подкреплением, позволяющая обучать автономных агентов непосредственно в актуальном клиенте игры Tanks Blitz. Проведенный сравнительный анализ алгоритмов и архитектур DRL свидетельствует о том, что математический аппарат метода обучения оказывает прямое, определяющее влияние на формируемую тактику поведения агента в условиях ограниченной пропускной способности и разреженных наград:

1 Алгоритм PPO продемонстрировал высочайшую стабильность обучения и, в связке со стратегией Curriculum Learning, позволил выработать тактику точного прицеливания и дальнего боя. Однако его On-policy природа привела к критической нехватке данных, сделав процесс масштабирования нецелесообразным по времени.

2 Переход к Off-policy методам выявил уязвимость базовых алгоритмов семейства SAC к нулевым наградам (взрыв энтропийного коэффициента и крах PopArt нормализации). Лишь архитектурные модификации в SDSAC (усреднение Q-значений,

отказ от PopArt и использование n-step) позволили стабилизировать сеть, благодаря чему агенты выучили тактику защитного позиционирования.

3 Алгоритм R2D2 подтвердил критическую важность рекуррентной памяти для сред с частичной наблюдаемостью, позволив агентам использовать укрытия и контролировать дистанцию, хотя и ценой цикличных микро-движений.

4 Гибридная архитектура Hybrid SAC, усиленная механизмами стабилизации SDSAC, решила проблему размерности пространства действий и устранила недостаток ступенчатого прицеливания, характерный для полностью дискретных алгоритмов.

Таким образом, для создания высокоуровневых автономных агентов в комплексных 3D-симуляторах недостаточно применения базовых RL-алгоритмов. Требуется их глубокая адаптация под физику среды. В дальнейшей работе планируется внедрение алгоритмов класса CTDE (например, QMIX) для формирования более сложных стратегий командного взаимодействия.

Список литературы

[1] Sutton R. S., Barto A.G. Reinforcement Learning: An Introduction. 2nd ed. Cambridge, MA: The MIT press; 2018.

[2] Terry J.K., Grammel N., Son S., et al. Revisiting Parameter Sharing in Multi-Agent Deep Reinforcement Learning. arXiv. 2020. DOI: 10.48550/arXiv.2005.13625.

[3] Espeholt L., Soyer H., Munos R., et al. IMPALA: Scalable Distributed Deep-RL with Importance Weighted Actor-Learner Architectures. Proceedings of the 35th International Conference on Machine Learning (PMLR). 2018;80:1407-1416.

[4] Ng A.Y., Harada D., Russell S.J. Policy Invariance Under Reward Transformations: Theory and Application to Reward Shaping. Proceedings of the Sixteenth International Conference on Machine Learning (ICML). 1999:278-287.

[5] Amodei D., Olah C., Steinhardt J., et al. Concrete Problems in AI Safety. arXiv. 2016. DOI: 10.48550/arXiv.1606.06565.

[6] Narvekar S., Peng B., Leonetti M., et al. Curriculum Learning for Reinforcement Learning Domains: A Framework and Survey. Journal of Machine Learning Research. 2020;21(181):1-50.

[7] Schulman J., Wolski F., Dhariwal P., et al. Proximal Policy Optimization Algorithms. arXiv. 2017. DOI: 10.48550/arXiv.1707.06347.

[8] Andrychowicz M., Raichuk A., Stańczyk P., et al. What Matters for On-Policy Deep Actor-Critic Methods? A Large-Scale Study. International Conference on Learning Representations (ICLR). 2021. DOI: 10.48550/arXiv.2006.05990.

[9] Christodoulou P. Soft Actor-Critic for Discrete Action Settings. arXiv. 2019. DOI: 10.48550/arXiv.1910.07207.

[10] Hessel M., Soyer H., Espeholt L., et al. Multi-Task Deep Reinforcement Learning with PopArt. Proceedings of the AAAI Conference on Artificial Intelligence. 2019;33(01):3796-3803. DOI: 10.1609/AAAI.V33I01.33013796.

[11] Haarnoja T., Zhou A., Hartikainen K., et al. Soft Actor-Critic Algorithms and Applications. arXiv. 2018. DOI: 10.48550/arXiv.1812.05905.

[12] Zhou H., Wei T., Lin Z., et al. Revisiting Discrete Soft Actor-Critic. arXiv. 2022. DOI: 10.48550/arXiv.2209.10081.

[13] Kapturowski S., Ostrovski G., Quan J., et al. Recurrent Experience Replay in Distributed Reinforcement Learning. International Conference on Learning Representations (ICLR) (Poster). 2019.

[14] Chen C., Deng X. A Hybrid SAC Algorithm: Advantage-Guided Soft Actor-Critic. MSCE '25: Proceedings of the 2025 International Conference on Management Science and Computer Engineering. 2025:459-462. DOI: 10.1145/3760023.3760100.

Авторский вклад

Гулис Антон Александрович – выбор задачи исследования, разработка программной среды взаимодействия с игровым клиентом, реализация архитектуры распределенной системы и нейросетевых моделей, проведение экспериментов, анализ полученных результатов.

Калугина Марина Алексеевна – постановка проблемы, научное руководство исследованием методов глубокого обучения с подкреплением для управления автономными агентами, консультации по математическому аппарату алгоритмов.

INVESTIGATION OF THE IMPACT OF REINFORCEMENT LEARNING METHODS ON AUTONOMOUS AGENT BEHAVIOR STRATEGIES

A.A. Gulis
Student of BSUIR

M.A. Kalugina
Associate Professor of Informatics Department of the BSU

Abstract. The article presents the results of a study on how various deep reinforcement learning (DRL) architectures and algorithms affect the development of behavioral strategies in autonomous agents. The current client of the multiplayer game Tanks Blitz is used as the test environment. We consider multi-agent cooperative–competitive interaction in a «2 vs 2» setting using the self-play paradigm, where an agent is trained against its own historical versions. Training in this environment is complicated by partial observability and bandwidth limitations. A distributed data-collection architecture is proposed, and the PPO, Discrete SAC (DSAC), Stable DSAC (SDSAC), R2D2, and a hybrid Hybrid SAC model are analyzed. We identify the phenomena of reward hacking, temperature coefficient explosion, and entropy collapse, and propose specific methods to mitigate them.

Keywords: deep reinforcement learning, self-play, PPO, SAC, DSAC, SDSAC, Hybrid SAC, reward shaping.