УДК 004.8
DOI https://doi.org/10.32689/maup.it.2024.2.1

В'ячеслав БОЧОК
аспірант кафедри інженерії програмного забезпечення в енергетиці,
Національний технічний університет України «Київський політехнічний інститут імені Ігоря Сікорського», vybochok@gmail.com
***ORCID:** 0009-0000-3929-2758*

Наталія ФЕДОРОВА
доктор технічних наук, доцент,
професор кафедри інженерії програмного забезпечення в енергетиці,
Національний технічний університет України «Київський політехнічний інститут імені Ігоря Сікорського», natasha_f@ukr.net
***ORCID:** 0000-0002-4548-4198*

ЦЕНТРАЛІЗОВАНЕ НАВЧАННЯ ДЛЯ DEEP Q-LEARNING МОДЕЛЕЙ

***Анотація**. Стаття присвячена використанню централізованого навчання та обміну знаннями між Deep Q-leaning агентами. Багатоагентні системи доволі стійкі до відмов та здатні до самоорганізації, проте досягнення цього може вимагати багато ресурсів. Агент самостійно досліджує середовище, поступово адаптуючись до різних ситуацій. Для систем, де простір станів є неперервним, а отже, має безліч варіантів, а результат переходу в майбутньому невідомий, для агента складно обирати досліджувати простір дій і станів, обирати вигіднішу стратегію та не застрягати у псевдовиграшних стратегіях (локальних мінімумах). **Метою** є підвищення стабільності процесу навчання. На прикладі підходу MADDPG та фреймворку KnowSR було запропоновано таку **методологію**: використати декілька агентів, що обмінюються досвідом та знаннями між моделями, утворюючи спільний буфер. **Науковою новизною** є використання централізованого навчання для підвищення стабільності дій Deep Q-learning агентів з механізмом обміну вже засвоєного знання.*

***Висновки**. Було проведено експерименти, що показали, що такий підхід значно підвищив стабільність навчання, зменшивши дисперсію між епізодами, а також збільшив швидкість навчання агентів. Кращий результат проявляється, коли показники успішнішого агента мають більший вплив при поширенні знань. З таким підходом агент, що знаходить кращу стратегію, «підтягує» інших агентів. На додаток до навчання на спільному досвіді також важливим є і навчання на власному, що дає можливість кожному агенту пробувати унікальні підходи та по-своєму досліджувати середовище, що час від часу може виводити його в лідери, та виводити інших з локального мінімуму площини оптимізації. Негативною стороною є те, що процес обміну знаннями також «стримує» агентів від різких змін стратегій, через що спостерігається, що агент, що навчається на власному досвіді, може різко виходити на значно більшу сумарну винагороду, хоч і нестабільно. Це продемонстровано в статті, на прикладі подвійного навчання за епізод агента на власному досвіді, коли агент демонструє як кращий, так і гірший результат.*

***Ключові слова:** deep Q-learning, reinforcement learning, knowledge distillation, обмін знаннями, централізоване навчання.*

Viacheslav BOCHOK, Nataliia FEDOROVA. CENTRALIZED LEARNING FOR THE DEEP Q-LEARNING MODELS

***Abstract.** The article is devoted to centralized learning and knowledge sharing between Deep Q-leaning agents. Multi-agent systems are fault-tolerant and capable of self-organization, but achieving this can require a lot of resources. The agent independently explores the environment, gradually adapting to different situations. For systems where the state space is continuous, and therefore has many options, and the outcome of the transition in the future is unknown, it is difficult for the agent to choose to explore the space of actions and states, select a more profitable strategy and not get stuck in pseudo-winning strategies (local minima). The **goal** is to increase the stability of the learning process. On the example of the MADDPG approach and the KnowSR framework, the following **methodology** was proposed: to use several agents that exchange experience and knowledge between models, forming a common buffer. The **scientific novelty** is the use of centralized learning to increase the stability of actions of Deep Q learning agents with a mechanism for sharing already learned knowledge. **Conclusions**. Experiments were conducted that showed that this approach significantly increased the stability of learning, reducing the variance between episodes, and also increased the learning rate of agents. The better result is manifested when the performance of the more successful agent has a greater influence on the diffusion of knowledge. With this approach, an agent that finds a better strategy «pulls up» other agents. In addition to learning from shared experience, learning from one's own is also important, which allows each agent to try unique approaches and explore the environment in its own way, which can sometimes lead it to become the leader, and lead others out of the local minimum of the optimization plane. On the negative side, the process of knowledge sharing also «restrains» agents from sudden changes in strategies, due to which it is observed that an agent learning from its own experience can dramatically reach a much higher total reward, albeit unstable. This is demonstrated in the paper, using the example of double learning per episode on an agent from its own experience, when the agent shows both better and worse results.*

***Key words:** deep Q-learning, reinforcement learning, knowledge distillation, exchange of knowledge, centralized training.*

Вступ. Багатоагентні системи характеризуються стабільністю та стійкістю до відмов [5], хоча можуть поступатися ефективності кожного окремого агента. Вони дають змогу відійти від традиційних

методів математичного моделювання та інженерних практик, які розглядають складну систему як централізовану та неподільну. Натомість багатоагентні системи розглядають систему як набір окремих інтелектуальних компонентів, які взаємодіють один з одним. Це дозволяє розв'язувати рівняння або складні, «незрозумілі» проблеми з непрозорою логікою тощо.

Архітектура багатоагентних систем нагадує реальні системи, такі як фінансові ринки, транспортні системи, соціальні структури тощо, що мотивує їх використання для розв'язання подібних завдань [7]. Такі агенти можуть бути як програмними, так і фізичними. Багатоагентна система (MAS – Multi-Agent System) – це система, яка складається з більш ніж одного інтелектуального агента та середовища, в якому вони діють, наприклад, обмінюються знаннями та співпрацюють. Ці системи не мають чітких центрів, жодна з її частин не описує завдання в цілому. Однак, зібрані разом, частини мають властивість самоорганізації та розв'язання кінцевої проблеми всієї системи [8].

Узагальнений термін «агент» може вказувати на реальну або віртуальну, автономну, розумну сутність, оснащену своїми власними цілями. Цілі та механізми їх визначення визначаються інженером під час проєктування. Іноді агент може самостійно розв'язувати завдання або взаємодіяти з іншими [2].

Агенти, здатні до навчання, є вкрай корисними для багатьох інженерних задач. Вони володіють всіма вищезгаданими характеристиками, але можуть використовуватися і в єдиному екземплярі для розв'язання деяких задач.

Процес навчання для таких агентів є досить складним і ресурсозатратним у порівнянні з традиційними задачами оптимізації. Ця стаття фокусується саме на агентах, політика яких базується на моделях машинного навчання. Хоч самі моделі, можуть бути такими самими, як і для навчання з вчителем (зведеними до задач класифікації чи регресії) [1], але процес навчання відрізняється. Різниця, викликана природою багатоагентної системи. Зазвичай вони розглядаються як марковський процес прийняття рішень (MDP), де його модель можна описати так:

1) Набір станів (або неперервний простір станів)
2) Набір дій (або неперервний простів для дії/дій)
3) Набір нагород (або функція залежності нагороди від переходу)
4) Функція переходу між станами залежно від дії

Можлива винагорода в поточному стані не залежить від дій у минулому. Кращим рішенням системи є набір таких дій залежно від стану, який максимізує загальну зібрану винагороду в епізоді. Згідно до принципу оптимальності Беллмана, найефективніша дія повинна ґрунтуватися не лише на поточній ситуації, але й на можливих майбутніх винагородах від усіх можливих наступних дій, які поки невідомі. Саме тому, методи навчання з вчителем неможливі, адже нема набору готових правильних відповідей [1].

Для навчання агентів зазвичай використовують підходи навчання з підкріпленням (англ. reinforcement learning). Враховуючи, що наперед невідомі всі переходи, або ж сумарна можлива майбутнья нагорода для кожного стану, то агент "досліджує" середовище, базуючись на власних прийнятих рішеннях, з кожним кроком уточнюючи власне розуміння ситуації. При цьому можна легко потрапити в ситуацію, коли агент обиратиме ті самі шляхи, не отримуючи нових знань, або ж переключившись на інший шлях, різко поміняє значення параметрів, "забувши" вже відомі стани. Це і робить навчання довгим і нестабільним.

Методи reinforcement learning можуть використовуються і для навчання одного агенту для розв'язання задач [3], якщо правильні відповіді заздалегіть невідомі, чи потребується самостійний пошук оптимальнох стратегії. Виникає закономірне питання, чи можна використати декілька агентів, що обмінюватимуться досвідом і знанням, що змогло б пришвидшити чи стабілізувати процес навчання.

Аналіз останніх досліджень і публікацій. Для нестаціонарних багатоагентних середовищ, де дії одних агентів можуть впливати на стан інших, існує метод MADDPG. Під час навчання кожен агент використовує додаткову інформацію від інших агентів (наприклад, їхні дії та спостереження) для стабілізації навчання. Такий централізований підхід дозволяє агентам ефективніше навчатися, враховуючи спільний простір станів і дій. Незважаючи на централізоване навчання, кожен агент виконує свою політику незалежно, використовуючи лише свої локальні спостереження. Це робить алгоритм придатним для реальних сценаріїв, де агенти не можуть отримати доступ до приватної інформації інших агентів під час виконання завдань. Головна ціль – навчити кілька агентів взаємодіяти в одному середовищі, де дії можуть бути як кооперативними, так і конкурентними.

Ідея навчання не тільки на власному досвіді, а й на досвіді інших агентів відносно мало вивчена. Наприклад, у статті була запропонована модифікація MADDPG.

Автори [9] модифікували метод алгоритм MADDPG. Їх підхід полягає в тому, щоб збирати досвід агента-актора, щоб пізніше поділитися ним у формі «порад» іншим агентам, щоб вони могли вчитися на цьому на додаток до власного досвіду. Завдяки такому підходу агенти швидше досягли кращих

результатів. Для досягнення такого єфекти вони використали механізм дистиляції знань (Knowlage Distillation) [10], що чудово підходить для агентів, модель яких виконує завдання класифікації. Під час дослідження вони виявили, що чим більше агентів ділиться своїм досвідом, тим кращих результатів вони досягають.

Постановка завдання. Для середовищ, що є стаціонарними, та де не задано агентів, що маєть взаємодіяти (кооперувати чи конкурувати), можна використати і Deep Q-learning (DQN). Це чудовий вибір, щоб навчити одного агента оптимальній політиці в середовищі з дискретними діями.

Логічним є припущення, що, замість одного DQN агенту, можна використати декілька, що будуть досліджувати світ незалежно, але з централізованим навчанням, обмінюючись досвідом і/або знаннями, що теоретично може підвищити стабільність навчання.

Методологія проведення експериментів. Для експериментів використовувалося середовище CartPole-v1 з бібліотеки OpenAI Gym. В якості моделі для DQN агенту було взято нейромережу зі слоями відповідно (4-24-12-2) нейрони.

Досвід агентів збирався в буфер розміром в 10 тисяч записів, що зберігається між епізодами. При навчанні на власному досвіді, дані з цього буферу береться випадково [6]. Основною причиною випадкової вибірки даних минулого є розрив кореляцій у даних. Дані між послідовними перехожами можуть бути сильно пов'язаними, що призведе до завчання стратегій, що не факт, що є оптимальними. Дані, які використовуються для навчання поточної політики, також генеруються поточною політикою, якщо не використовується буфер відтворення досвіду. Усереднення навчання шляхом випадкової вибірки даних від багатьох попередніх політик за допомогою буфера відтворення досвіду може допомогти запобігти значним осциляціям. Під політикою мається на увазі нейромережа, між етапами навчання.

Функція активації Relu для всіх слоїв, але останній лінійна. Для навчання на власному досвіді використовувався batch_size = 32, 1 епоха. MSE використовується як функція втрат.

Для подолання дилеми дослідження та експлуатації (з англ. exploration-exploitation dilemma) [4], агент обирає випадкову дію у першому епізоді у 100% випадків, в геометричній прогресії збільшуючи вплив політики на вибір дії. Множник прогресії дорівнює 0.995. Мінімальна можливість випадкової дії становить 1%.

На деяких графіках пунктиром зображений контрольний агент, що навчався тільки на власному досвіді (класичний підхід для DQN моделі). Механізм навчання через причини, згадані вище, доволі нестабільний, тому мета графіків показати не точні цифри, а тенденції, що зберігаються при повторному запуску з випадковою ініціалізацією. Контрольний агент на момент 500-го епізоду зазвичай мав значення від 50 до 300.

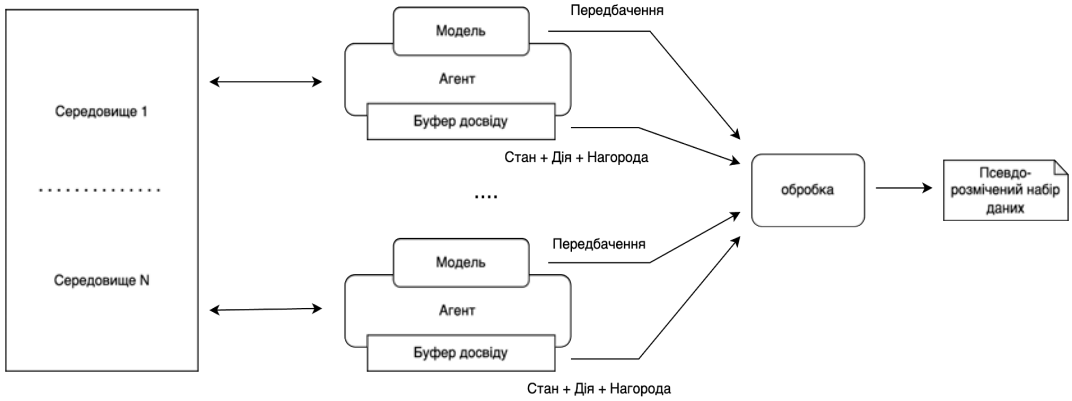

Рис. 1. Схема механізму додаткового навчання на спільному досвіді

Аналіз результатів. В ході експериментів було використано 2 механізми навчання:

1) На власному буфері досвіду (вираховуючи помилку між попередніми передбаченнями та передбаченням, після уточнення нагороди за крок);

2) На спільному досвіді, з утворенням псевдо-розміченого набору даних (рис. 1).

Для деяких експериментів перший і другий спосіб об'єднувалися, що давало найкращий та найстабільниший результат.

Під навчанням на спільному досвіді мається на увазі, що після епізоду (до навчання на власному досвіді, якщо таке планується), агенти діляться ситуаціями, в яких вони були (рис. 1: стан + дія +

нагорода) в спільний буфер. Далі, всі агенти оцінюють всі ситуації з цього буферу власною моделлю. Далі, передбачення агентів для моделей узагальнюються (або береться середнє, або зважена сума тощо), і утворюється псевдо розмічений набір даних, що використовується для навчання всіх агентів. Далі, для зменшення розмірності буферу датасет зменшується семплуванням.

Для експериментів на рисунку 2 і 3 не використовувалося навчання агентів на власному досвіді. Для узгодження передбачень (рис. 2) агентів використовувалося середнє значення. Як видно, це привело до повного узгодження показників та деградації.

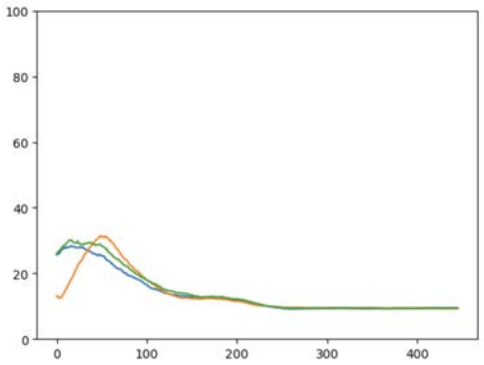

Рис. 2. Графік Moving Average(50) від зібраної сумарної нагороди за епізод з навчанням тільки на спільному досвіді (усереднення передбачень)

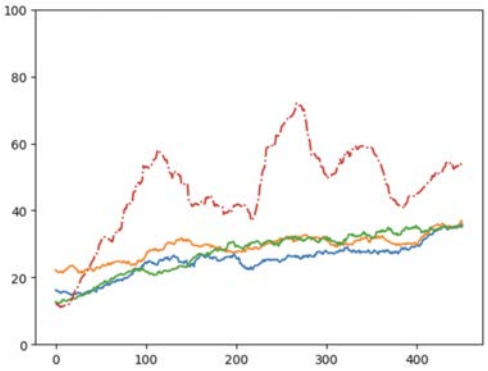

Рис. 3. Графік Moving Average(50) від зібраної сумарної нагороди за епізод з навчанням на спільному досвіді (зважена сума передбачень)

Для експерименту (рис. 3) було пораховано зважену суму передбаченнь. Очевидним результатом для рисунку 3 є висока стабільність показників під час навчання, але низька швидкість навчання (у порівнянні з рис. 4 та 5 та з контрольним агентом).

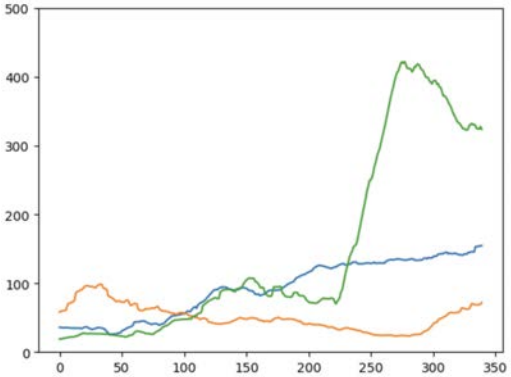

Рис. 4. Графік Moving Average(50) від зібраної сумарної нагороди за епізод з навчанням на власному та спільному досвіді (усереднення передбачень)

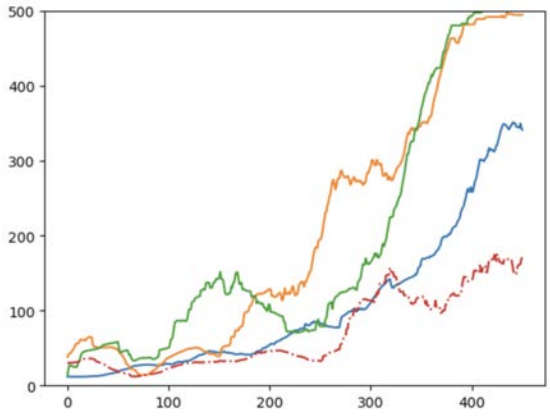

Рис. 5. Графік Moving Average(50) від зібраної сумарної нагороди за епізод з навчанням на власному та спільному досвіді (зважена сума передбачень)

Для експериментів на рисунках 4 і 5 було використано такий самий підхід, для спільного навчання, що і для рисунку 2 і 3, але додатково застосовувалося навчання на власному досвіді. Очевидно, що використання спільного буферу досвіду підвищило стабільність у порівнянні з контрольним агентом, але успіх одних агентів не передавався іншим (рис. 4), як це видно на рисунку 5. Очевидно, що коли один агент знаходить кращу стратегію, він покращує результат інших агентів. На рисунку 6 видно, що різниця між показниками агентів є (спостерігається один відстаючий агент), але всі вони кращі за контрольний.

Для порівняння також наведено експеримент (рис. 6), де агенти навчалися в 2 рази більше тільки на власному досвіді, без поширення знань іншим. Очевидна висока нестабільність, що призводить до падінь та швидких стрибків ефективності.

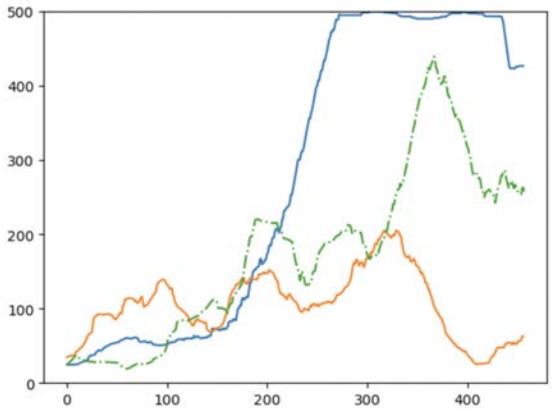

Рис. 6. Графік Moving Average(50) від зібраної сумарної нагороди за епізод з подвійним навчанням на власному досвіді

Висновок. Результати експериментів показують, що використання спільного досвіду для навчання та поширення знань може бути дієвим інструментом у поєднанні з класичним навчанням агенту на власному досвіді, що призводить до суттєвого підвищення стабільності та швидкості навчання Deep Q агентів. Для узгодження передбачень різних агентів для ситуації краще використовувати зважену суму, де вага визначається успішністю агенту за попередній епізод. Такий підхід дозволяє агенту, що знайшов переможну стратегію поширити її і на інших агентів, але без повного узгодження, що дозволяє агентам шукати нові рішення без потрапляння в спільний локальний мінімум (що видно, коли агенти з найкращими показниками змінюють один одного).

Список використаних джерел:

1. Eysenbach B., Kumar A. Reinforcement learning is supervised learning on optimized data. The BAIR Blog. 2020. February 1, 2024, Retrieved from https://bair.berkeley.edu/blog/2020/10/13/supervised-rl/

2. Gao Z., Xu K., Ding B., Wang H., Li Y., Jia H. KnowSR: Knowledge Sharing among Homogeneous Agents in Multi-agent Reinforcement Learning. 2021. (arXiv preprint arXiv:2105.11611).

3. Hinton, Geoffrey; Vinyals, Oriol; Dean, Jeff (2015). «Distilling the knowledge in a neural network». arXiv:1503.02531
4. Leitão, Paulo; Karnouskos, Stamatis (March 26, 2015). Industrial agents: emerging applications of software agents in industry. Leitão, Paulo, Karnouskos, Stamatis. Amsterdam, Netherlands. ISBN 978-0128003411. OCLC 905853947.
5. M. Brambilla, E. Ferrante, M. Birattari and M. Dorigo, «Swarm robotics: A review from the swarm engineering perspective», Swarm Intell., vol. 7, no. 1, pp. 1-41, 2013.
6. M. Dorigo, G. Theraulaz and V. Trianni, «Reflections on the future of swarm robotics», Sci. Robot., vol. 5, no. 49, 2020.
7. Mnih V. et al. Playing atari with deep reinforcement learning //arXiv preprint arXiv:1312.5602. 2013.
8. Richard S. Sutton, Andrew G. Barto. Reinforcement Learning: An Introduction (2nd edition). 2020.
9. Stefano V. Albrecht, Filippos Christianos, Lukas Schäfer. Multi-Agent Reinforcement Learning: Foundations and Modern Approaches. MIT Press, 2024. https://www.marl-book.com/
10. Wooldridge, Michael. An Introduction to MultiAgent Systems. John Wiley & Sons. 2002. p. 366. ISBN 978-0-471-49691-5.