THEORY OF INFORMATION TECHNOLOGIES AND SYSTEMS CONSTRUCTION

ТЕОРІЯ ПОБУДОВИ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ ТА СИСТЕМ

https://doi.org/10.15407/intechsys.2025.04.003
УДК 004.87 + 004.032.26

О.А. УРСАТЬЄВ, канд. техн. наук., старш. наук. співроб., пров. наук. співроб.,
Інститут інформаційних технологій та систем НАН України,
просп. Акад. Глушкова, 40, м. Київ, 03187, Україна
https://org/0009-0009-8323-0525
aleksei@irtc.org.ua

О.Є. ВОЛКОВ, канд. техн. наук., старш. дослідник, директор,
Інститут інформаційних технологій та систем НАН України,
просп. Акад. Глушкова, 40, м. Київ, 03187, Україна
https://orcid.org/0000-0002-5418-6723
alexvolk@ukr.net

ПОСИЛЕНЕ НАВЧАННЯ НЕЙРОМЕРЕЖІ В УЯВІ В СИСТЕМАХ КЕРУВАННЯ БЕЗПІЛОТНИМИ РУХОМИМИ ОБ'ЄКТАМИ

Проаналізовано зарубіжний досвід розроблення та застосування засобів штучного інтелекту, а саме глибокого посиленого навчання за моделлю для розв'язання проблем поведінки рухомих об'єктів у невідомих частково спостережуваних середовищах. Досліджено задачу керування рухомими об'єктами в одно- та багатоагентних системах із застосуванням ментальної моделі світу. Такі системи діють за аналогією роботи мозку людини. Для розв'язання задачі керування рухомими об'єктами застосовують великі рекурентні нейронні мережі – моделі, які здатні навчатися за даними виміряними у часі та просторі. Для вибору оптимальної стратегії дій агентів й точного відтворення середовища, вхідні дані мають бути високої розмірності. На основі проведеного аналізу запропоновано застосування відомого підходу на основі глибинного посиленого навчання для розв'язання задачі керування рухомими об'єктами. Мета керування за цим підходом досягається шляхом побудови моделі уявлення світу замість проведення реальних дорогих випробувань.

Ключові слова: *безпілотні рухомі об'єкти, глибоке посилене навчання, ментальна модель світу, нейромережі, навчання агентів, багатоагентне середовище, рекурентна модель простору станів.*

Цитування: Урсатьєв О.А., Волков О.Є. Посилене навчання нейромережі в уяві в системах керування безпілотними рухомими об'єктами. *Information Technologies and Systems*. 4 (4). 2025. 3–27. https://doi.org/10.15407/intchsys.2025.04.003

Вступ

Спираючись на огляд робіт [1–4], складне завдання створення мультиагентних систем, що включають рухливі об'єкти, доцільно вирішувати за допомоги засобів штучного інтелекту (*AI*), який генерується глибокими нейронними мережами. Сучасні технології намагаються імітувати процеси навчання та оброблення інформації в людському мозку шляхом створення штучних нейронних мереж, які здатні виявляти в даних складні взаємозв'язки. Так, робота [3] присвячена великим рекурентним (циклічним) нейронним мережам (*RNN*)[1], які з нуля навчилися керувати змодельованими автомобілями (*simulated cars*) за допомоги багатомірного відео-входу. Однак людський мозок багато в чому більш потужний. Зокрема, люди вивчають прогнозну модель спочатку невідомого їм середовища і якимсь чином використовують її для абстрактного планування та роздумів. Керуючись алгоритмічною теорією інформації, було відтворено алгоритм штучного інтелекту *AI* (*RNNAI*), призначеного для цих же цілей, щоб зрозуміти послідовність операцій і дій під час навчання *RNN* за моделлю. Таким *RNNAI* алгоритмом можна навчати багато послідовних завдань, деякі з яких задає користувач, інші генерує сам *RNNAI* в допитливій ігровій формі, і тим самим покращує свою модель світу. На відміну від попередніх розробок моделей для навчання за підтримкою, створених у 1990 році, *RNNAI* навчає свою модель для абстрактних роздумів, планування та прийняття рішень, по суті, «навчаючись думати».

Метою статті є аналіз сучасного зарубіжного досвіду розроблення і застосування аналітичних платформ керування рухомими об'єктами в одно- і багатоагентних системах. Таке керування відбувається засобами штучного інтелекту, згенерованого глибокими нейронними мережами, навчаючи модель за допомоги посиленого навчання у невідомих, частково спостережуваних середовищах. Як модель середовища застосувалася ментальна модель світу, підхід, який добре себе зарекомендував внаслідок навчання нейромереж подібно тому, як це відбувається у мозку людини. Мета керування за цим підходом досягається шляхом побудови моделі уявлення світу, замість проведення реальних дорогих випробувань.

Штучні нейронні мережі

Штучні нейронні мережі — це ядро систем глибокого навчання, їхня базова технологія. Терміни *глибоке навчання* та *нейронні мережі* взаємозамінні, тому що всі системи глибокого навчання складаються з багаторівневих нейронних мереж — мають кілька прихованих шарів,

[1] *AWS*. Що таке *RNN* (рекурентна нейронна мережа)? — *https://aws.amazon.com/en/what-is/recurrent-neural-network/*

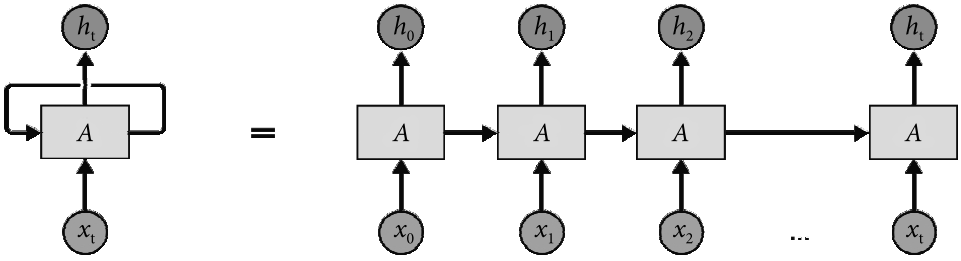

Рис. 1. Фрагмент та розгорнута рекурентна нейронна мережа

які роблять їх глибокими. Причина, через яку більша кількість прихованих шарів збільшує можливості нейронної мережі, ґрунтується на принципі, що дані може бути подано на різних рівнях абстракції. Нижні рівні абстракції використовуються ближче до вхідного шару (x_i), а вищі — до вихідного шару (h_i). Відповідно, глибокі нейронні мережі можуть отримувати ознаки даних на дедалі вищих рівнях абстракції. Спочатку для нейронів, які мають рекурентний зворотній зв'язок, він зображувався як петля (рис. 1)[2]. Це зображення відтворено іншим, у якому рекурентний зв'язок розгортається у часі. У ньому рекурентний зв'язок показано як спрямовану стрілку, що вказує від нейрона (*A*) на попередньому кроці у часі до того самого нейрона на наступному кроці.

Це циклічні мережі, що дають змогу інформації зберігатися. Фрагмент нейронної мережі (на рис. 1 ліворуч) приймає вхідні дані (x_t) та видає значення (h_t). Цикл дає змогу передавати інформацію з одного кроку мережі на інший. *RNN* можна розглядати як копії тої самої мережі, де кожна копія надсилає повідомлення наступній.

Нейронні мережі прямого поширення (*feedforward NN* (*FNN*)) є ациклічними і обробляють дані в одному напрямку від вхідного до вихідного вузла [5]. Поверхневі (неглибокі) і глибокі нейромережі, відрізняються шляхами присвоєння ваг або кредитів, які є ланцюжками нейронів (*A*), що навчаються причинно-наслідковим зв'язкам між діями і ефектами. Існує два основних типи систем глибокого навчання з різною архітектурою — рекурентні нейронні мережі (*RNN*) та згорткові нейронні мережі (*CNN*)[3].

Проблема пошуку тривалих залежностей. Однією з привабливих сторін *RNN* є ідея, що вони можуть пов'язувати попередньо отриману інформацію з поточним завданням, наприклад, аналіз попередніх відеокадрів може допомогти в розумінні поточного кадру. Іноді нам потрібно лише проаналізувати нещодавню інформацію, щоб виконати поточне завдання. У випадках, коли розрив між релевантною інформацією та контекстом, у якому вона потрібна, невеликий, *RNN*

[2] *Blog Olah Christopher. Understanding LSTM Networks. August, 2015. – http://colah.github.io/posts/2015-08-Understanding-LSTMs/*

[3] *What is deep learning in AI? – https://aws.amazon.com/what-is/deep-learning/?nc1=h_ls*

можуть навчитися використовувати попередню інформацію. В іншому разі — ні, традиційні нейронні мережі не можуть цього зробити.

Мережі з довгою короткочасною пам'яттю [4] — це особливий вид *RNN*, здатний вивчати довгострокові залежності. Вони спеціально розроблені, щоб уникати проблеми тривалих залежностей. Запам'ятовування інформації протягом тривалих періодів часу — це їхня поведінка за умовчанням, а не тому, що її не вистачає для навчання. Всі рекурентні нейронні мережі мають форму ланцюжка модулів нейронів що повторюються. У стандартних рекурентних нейронних мережах цей модуль, що повторюється, має дуже просту структуру, наприклад, один шар тангенса кута діелектричних втрат (*tanh*).

Навчання нейромережі за уявною моделлю

Наразі в складних системах із застосуванням штучного інтелекту для розв'язання завдань застосовується посилене навчання (навчання з підкріпленням, *Reinforcement Learning, RL*) [6]. Ми зосередимось на підході посиленого навчання, який засновано на моделі (*World Models introduces a Model-based approach to Reinforcement Learning*) [1, 2]. В [2] автори об'єднали кілька ключових концепцій із серії статей 1990 — 2015 років за моделями світу та контролерами на основі *RNN-based* [3, 7–10]. У роботі [2], де пропонується розглядати модель світу з більш сучасними інструментами ймовірнісного моделювання, подано спрощений підхід для перевірки деяких із цих ключових концепцій у сучасних середовищах [5] *RL* [11]. Подальші експерименти показали, що цей підхід може бути використаний для розв'язання складної задачі навігації гоночного автомобіля за пікселями, яку раніше не змогли подолати з використанням традиційних методів. Більшість підходів *RL* на основі наявних моделей [5], вивчають модель середовища *RL*, але все ще навчаються на реальному середовищі. В [5] запропоновано дослідити повну заміну реального середовища *RL* на згенероване, і навчати контролер агента тільки всередині середовища, за його власною внутрішньою моделлю світу, а потім переносити цю політику в реальне середовище.

---

[4] Довга короткочасна пам'ять (*long short-term memory, LSTM*) – це модифікація архітектури рекурентних нейронних мереж, що дає змогу навчання тривалим залежностям, запропонована 1997 р. *Sepp Hochreiter і Jürgen Schmidhuber. RNN і LSTM* є спеціальними архітектурами нейронних мереж, здатними обробляти послідовні дані, у яких важливим є хронологічний порядок. *LSTM,* значно покращені версії *RNN,* дають змогу інтерпретувати довші послідовності даних [***Long short-term memory*** *[PDF] Hochreiter, S. and Schmidhuber, J., 1997. The MIT Press. Neural Computation.* 9(8):1735-1780 *DOI:10.1162/neco.1997.9.8.1735*]

[5] *OpenAI Gym* – набір інструментів для досліджень у галузі посиленого навчання. Включає в себе колекцію еталонних завдань, що постійно зростають, які мають спільний інтерфейс, а також веб-сайт, на якому люди можуть ділитися своїми результатами і порівнювати продуктивність алгоритмів – *Brockman G., Cheung V., Pettersson L., OpenAI Gym* 2016 *https://doi.org/10.48550/arXiv.1606.01540*

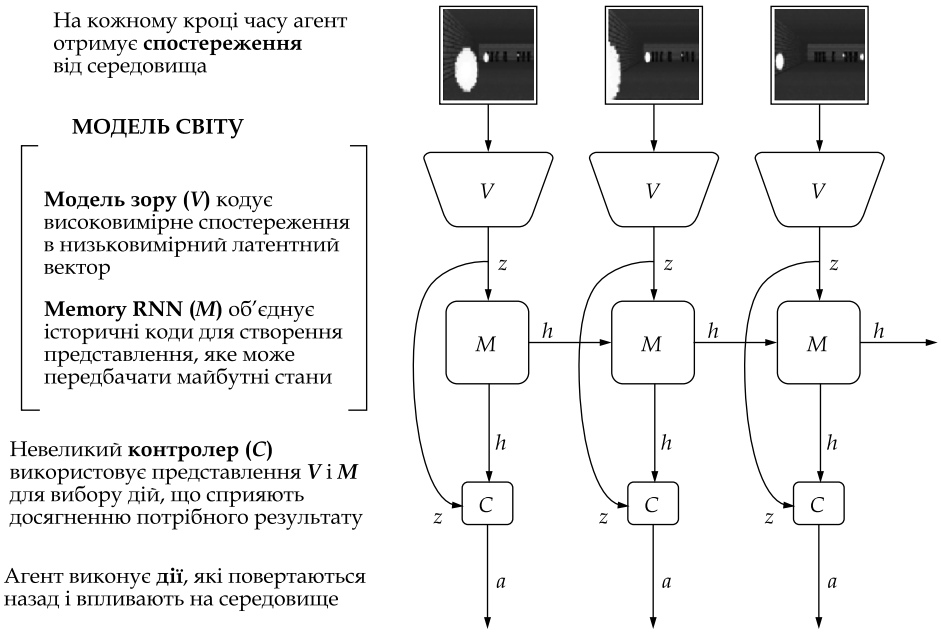

Рис. 2. Агент складається з трьох компонентів, які тісно взаємодіють: бачення (*V*), пам'ять (*M*) і контролер (*C*)

В [2] розглянуто посилене навчання великої нейронної мережі[6] при вирішенні завдань поставленої задачі *RL*. При цьому агент включає як велику модель світу так і невелику модель контролера (рис. 2). Спочатку здійснюється самонавчання великої нейронної мережі моделі світу агента, потім меншої моделі контролера — для вміння виконувати завдання з використанням цієї моделі світу. Контролер дає змогу алгоритму навчання зосередитися на задачі присвоєння кредитів у невеликому просторі пошуку, не жертвуючи при цьому продуктивністю та наочністю моделі світу.

Модель агента. Автори [2] надали просту модель, згенеровану власною когнітивною системою. У цій моделі[7] в агента (див. рис. 2) є візуально-сенсорний компонент *V*, що стискає зображення, яке спостерігається, у невеликий репрезентативний код. Він також має компонент пам'яті *M*, що робить прогнози про майбутні коди на основі раніше накопиченої (історичної) інформації, отриманої за минулих прихованих дій. У результаті агент має компонент прийняття рішень — контролер *C*, який визначає, які дії треба зробити, ґрунту-

[6] Типові *RL* без моделі (*Model-free*) мають від 10^3 до 10^6 параметрів моделі. Розглядалися моделі навчання з параметрами порядку 10^7, що все ще досить мало в порівнянні з сучасними моделями глибокого навчання з параметрами від 10^8 до 10^9. У принципі, процедура, описана в [2], може використовувати переваги цих великих мереж.

[7] Алгоритм, параметри якого були налаштовані за допомоги машинного навчання, називаються моделлю.

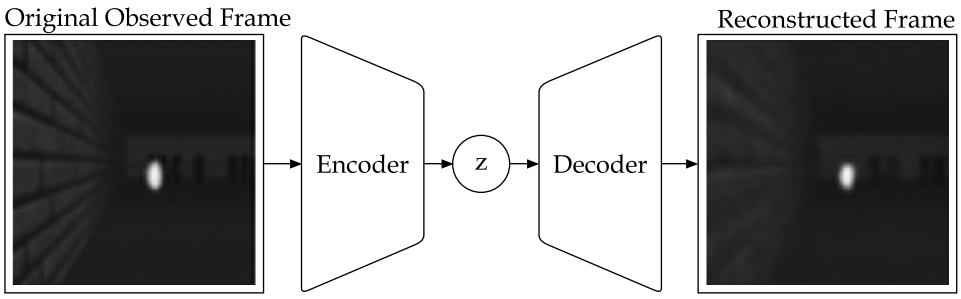

Рис. 3. Блок-схема варіаційного автокодера ***VAE*** (***V***) [12]

ючись тільки на уявленнях, створених його компонентами бачення *V* і наявної пам'яті *M*.

Сама ідея взаємодії компонентів полягає у вивченні моделі прогнозованого світу *M*, яку контролер *C* може використовувати для ефективнішого досягнення своїх цілей, наприклад, за допомоги дешевих *уявних* випробувань на основі *M*, на відміну від дорогих реальних випробувань. *M* намагається навчитися передбачати вхідні дані *C* (включно з сигналами винагороди) на основі попередніх вхідних даних та дій. *M* також використовується як імітація навколишнього середовища: *M* і *C* утворюють зв'язану *RNN*, де вихідні дані *M* стають вхідними даними *C*, а її вихідні дані (дії), надалі стають вхідними даними *M*.

Модель VAE (***V***). Середовище надає агенту багатовимірні вхідні дані на кожному етапі. Ці дані зазвичай є кадром 2*D*-зображення, який є частиною відеопослідовності. Роль моделі *V* полягає у вивченні абстрактного стисненого кожного спостережуваного вхідного кадру, в яких є фігури, що переміщаються і набувають різних поз. Використано варіаційний автокодер[8] [12], де генеративна модель навчає як кодувальника, так і декодер. Кодувальник стискає вхідні зображення, які він отримує на кроці *t*, в низькорозмірний прихований вектор (*Latent Vector*) z_t. Завдання декодера — відновити вихідне зображення з прихованого подання (рис. 3) [12].

Model M. Рекурентна нейромережа-симулятор **(*MDN-RNN*)**. Роль моделі *V* (див. рис. 2) полягає в тому, щоб стискати те, що бачить агент на кожному кроці. Це стиснення проводиться постійно. Для цього завданням моделі *M* стає передбаченням майбутнього. Модель *M* служить прогностичною моделлю майбутніх векторів *z*, які, як

[8] Варіаційні автокодувальники призначено для стискання інформації входу до обмеженого багатовимірного латентного розподілу, щоб відбудовувати її якомога точніше. Це стохастичний варіаційний алгоритм виведення і навчання у спрямованих імовірнісних моделях за наявності безперервних прихованих змінних з важкорозв'язними апостеріорними розподілами на великих наборах даних. Він належіть до сімейств імовірнісних графових моделей та варіаційних баєсових методів — Вікіпедія. Варіаційний автокодувальник — *https://uk.wikipedia.org/wiki/Варіаційний_автокодувальник*

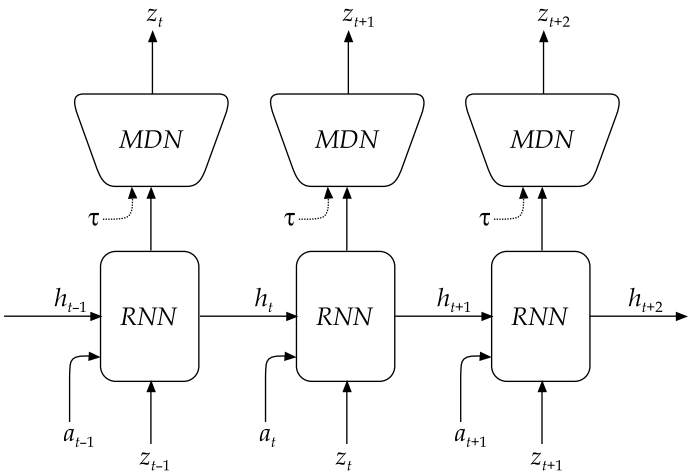

Рис. 4. Структура моделі *M* — рекурентної нейромережі-симулятора *MDN-RNN*

очікується, створить *V*. Оскільки багато складних середовищ є стохастичними за своєю природою, *RNN* навчається виводити функцію щільності ймовірності $p(z)$ замість детермінованого прогнозу *z*. Тобто, *M*-модель (див. рис. 2 та рис. 4) навчена передбачати сховане кодування наступного кадру з урахуванням попередніх схованих кодувань та дій. Для моделі *M* використано рекурентну нейронну мережу *LSTM* у поєднанні з мережею «щільності суміші»[9] *MDN* використано як вихідний шар. На виході *MDN* (рис. 4) — параметри суміші, яка використовується для передбачення наступного прихованого вектора *z*. Реалізована *M* як рекурентна нейронна мережа з довгою короткочасною пам'яттю (*long short-term memory, LSTM*) у поєднані з *Mixture Density Network, MDN*. Температурний параметр τ (див. рис.4) служить для налаштування контролю невизначеності моделі[10] під час навчання контролера *C* [2].

Алгоритм навчання в уяві

Модель контролера *C* (рис. 5) відповідає за визначення послідовності дій, які необхідно зробити для максимізації очікуваної сукупної ви-

[9] Термін «щільність суміші» стосується того факту, що складний розподіл імовірностей може бути створений поєднанням («змішуванням») кількох простих розподілів імовірностей. Прості розподіли зазвичай є Гауссовими розподілами.

[10] ***E2-Create Education*** — дослідницький проєкт, який отримав фінансування від дослідницької та інноваційної програми Європейського союзу *Horizon* 2020 в рамках грантової угоди Марії Склодовської-Кюрі № 840465. Подано документи, що стосуються тем і програмних засобів, які було розглянуто та розроблено в рамках цього проєкту. Він також охоплює декілька бібліотек програмного забезпечення з відкритим кодом і моделей машинного навчання. Розділ «*Pose Sequence Generation (RNN+MDN). Summary*» — *https://wp.coventry.domains/e2edu/pose-sequence-generation-rnnmdn/*

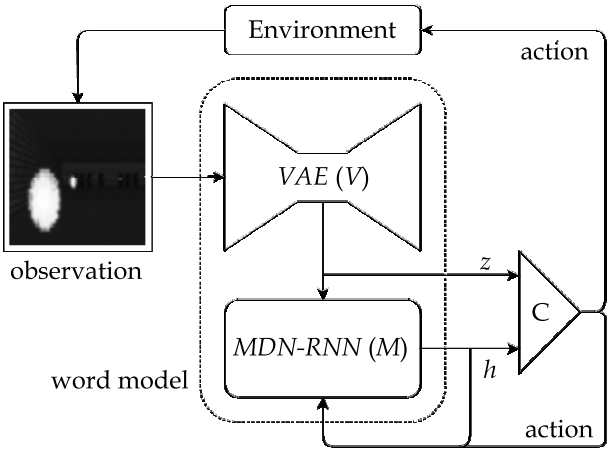

Рис. 5. *Блок-схема моделі агента*

нагороди агента під час розгортання середовища. На кожному кроці в часі контролер приймає як вхідні дані закодований поточний кадр і поточний стан *MDN-RNN*, де міститься інформація про всі попередні кадри та дії. Контролер *C* зроблено максимально простим, і навчати його потрібно окремо від *V* і *M*, тому основна частина агента пов'язана з моделлю світу (*V* і *M*). Цей мінімальний дизайн має важливі практичні переваги. Досягнення в галузі глибокого навчання забезпечили інструменти ефективного навчання великих, складних моделей за умови, що можливо визначити функцію втрат, яка диференціюється. Моделі *V* і *M* розроблені для ефективного навчання за допомоги алгоритму зворотного розповсюдження з використанням сучасних графічних процесорів *GPU*, тому бажано, щоб основна частина складових моделі та її параметрів входила до *V* і *M*.

Контролер *C* — проста одношарова лінійна модель, яка перетворює z_t і h_t безпосередньо на дію a_t на кожному кроці (див. рис. 5) [2]. Кількість параметрів *C* лінійної моделі мінімальна порівняно з *V* та *M*. Контролер навчається, взаємодіючи із середовищем за допомоги *CMA-ES*. Таке рішення дало змогу дослідити не традиційні способи навчання *C*, наприклад, використовуючи стратегії еволюції (*ES*) адаптації коваріаційної матриці (*CMA-ES*)[11] [13] як алгоритм оптимізації. Зменшена розмірність *Z* дає змогу класичному еволюційному алгоритму з нею впоратися.

[11] *CMA-ES*: Стратегія розвитку *CMA* (*ES*), де *CMA* означає адаптацію матриці коваріації. *CMA-ES* — це стохастичний або рандомізований метод оптимізації нелінійних неопуклих функцій з дійсними параметрами (безперервна область). Це алгоритм для складних завдань оптимізації чорної скриньки в безперервній області. Еволюційна стратегія (*EC*) є стохастичною, похідною від методів чисельної оптимізації нелінійних або неопуклих проблем безперервної оптимізації. Належить до класу еволюційних алгоритмів і еволюційних обчислень — Вікіпедія — *https://uk.wikipedia.org/wiki/CMA-ES*

Суть роботи алгоритму зводиться до такого: *VAE*(*V*) автокодер вивчає модель світу і передає своє середнє значення зі зменшеною розмірністю *Z* в рекурентну нейромережу *MDN-RNN* (*M*) для відстеження змін. *M* видає теж стисле *Z*, але вже для наступного кроку в симуляції, що дає змогу нейромережі *MDN-RNN* робити прогноз на кілька кроків уперед, передаючи свій прогноз *Z* собі на вхід, і так кілька разів поспіль. Відтак цей прогноз, що видається рекурентною нейромережею-симулятором, можна застосувати для навчання основної нейромережі для розв'язання задачі. Тут же основне навчання відбувається в *уяві*, механізм його утворений (див. рис. 2) за циклом між *MDN-RNN* і *C*. Функцію прийняття рішення покладено на контролер *C*. Він, щоб взаємодіяти не тільки в рамках своєї уяви, але і з безпосереднім симулятором, також передає вектор дії в навколишнє середовище для поповнення бази знань автокодера *VAE* (*V*).

Резюмуючи, контролер *C* приймає як вхідні дані приховане кодування поточного кадру, і прихований стан *MDN-RNN* з урахуванням минулих прихованих кодувань та дій, і видає нову дію. Він навчений максимізувати накопичену винагороду з використанням стратегії еволюції адаптації коваріаційної матриці, універсального алгоритму оптимізації чорної скриньки.

Автори [2] звертають увагу на те, що прогноз z_{t+1} не подається на контролер *C* безпосередньо, а тільки як прихований стан h_t і z_t. Це тому, що h_t має всю інформацію, необхідну для генерації параметрів суміші розподілу Гауса, якщо ми хочемо підготувати вибірку z_{t+1} для прогнозу.

Результати роботи [2] застосовано у матеріалах проєкту[12] та відтворені у блозі[13]. У записі блогу *Reproducing* «*World Models*» наведено результати поглибленої перевірки роботи [2] за посиленим навчанням на основі моделі, яка демонструє високу продуктивність у складному середовищі *CarRacing-v0*[14]. Зокрема, ці результати легко відтворити. Ймовірно, це означає, що метод для цієї проблеми не лише досягає високої продуктивності, а й дуже стабільний. Це важливе зауваження щодо методу посиленого глибокого навчання. Також надано матеріали з реалізації на *PyTorch*[15] та проведено додаткові експе-

[12] ***E2-Create Education*** — дослідницький проєкт № 840465. 8 July 2022 — *https://wp.coventry.domains/e2edu/2022/07/08/hello-world/*

[13] ***Bblog Tallec, C., Blier, L., Kalainathan, D. Reproducing "World Models"*** — *https://ctallec.github.io/world-models/. In this blog post, we are delving into World Models (Ha et al., 2018) a recent model based reinforcement learning paper that achieves surprisingly good performance on the challenging CarRacing-v0 environment. Along with a short summary of the paper, we provide a PyTorch implementation , as well as additional experiments on the importance of the recurrent network in the training process.*

[14] *Car_Racing_v0* належить до сімейства *Box2D* популярних завдань *RL* — *https://github.com/AGiannoutsos/car_racer_gym*

[15] Високопродуктивна, масштабована та готова до використання на підприємстві платформа *PyTorch* на *AWS* — *https://aws.amazon.com/en/pytorch/*

рименти щодо важливості рекурентної мережі у процесі навчання. Зокрема у середовищі *CarRacing-v*0, схоже, рекурентна мережа служить лише рекурентним резервуаром, забезпечуючи доступ до важливої інформації про динаміку середовища. Навіть якщо рекурентна модель не здатна передбачити наступний стан середовища, її рекурентний стан все одно містить інформацію першого порядку, таку як швидкість автомобіля, яка може бути відсутня в окремих кадрах, отже, і у прихованих кодах. Тому стратегії, навчені без *MD-RNN*, не можуть використовувати таку інформацію.

Посилене навчання комп'ютерним іграм із застосуванням моделі світу

Ще один приклад середовищ, добре знайомих більшості з нас є комп'ютерні ігри. При їх розгляді виникає думка [14], чому гравці можуть навчитися грати, скажімо, в ігри *Atari*[16] за лічені хвилини, однак деякі з найкращих безмодельних алгоритмів посиленого навчання *RL* вимагають десятків чи сотень мільйонів кроків у часі – еквівалент кількох тижнів навчання в реальному часі, тобто значно більше, ніж людині необхідно вивчення тих самих ігор. Чому люди можуть навчитися грі набагато швидше? Частково відповідь може полягати в тому, що вони можуть вивчити, як працює гра, і передбачати, які дії призведуть до бажаних результатів. Тобто люди мають інтуїтивне розуміння фізичних процесів, відтворених у грі, і можуть передбачити наслідки своїх дій.

Ідея посиленого глибокого навчання нейромережі в уяві з використанням моделі світу та концептуальні рішення з реалізації цього підходу, викладені в [2], і застосовано в роботі [14]. Це дало змогу використати головну перевагу навчання на основі моделей (*Model-Based*) — істотно підвищити ефективність у застосуванні досвіду або підняти ефективність вибірки під час навчання. Уявлення про довкілля надає агенту можливість передбачити своє майбутнє і має фундаментальну привабливість для посиленого навчання. Досліджено, як навчені моделі на основі відео можуть уможливити навчання в *ALE* середовищі *Atari*, а також аналогічним чином дозволити агентам навчатися грі *Atari* на основі спостережень за зображеннями та з меншою кількістю кроків, ніж методи без моделей. Створено алгоритм посиленого навчання на основі моделі під назвою *Simulated Policy*

[16] Ігри *Atari* набули популярності як еталон для посиленого навчання із запровадженням Аркадного навчального середовища (*ALE*) *Bellemare et al*. (2015). Навчання інтелектуальних агентів відбувається в середовищі *Atari*, яке представляє емуляцію ігрової консолі *Atari*2600 за допомоги платформи *OpenAI Gym*. Поєднання посиленого навчання та глибоких моделей потім дозволило алгоритмам *RL* навчитися грати в ігри *Atari* безпосередньо з зображень ігрового екрану

Learning (*SimPLe*), який працює безпосередньо на сирих піксельних спостереженнях та вивчає ефективні політики для ігор у середовищі *ALE* [14]. Навчання формалізовано у Марківських процесах прийняття рішень (*MDP*) [4]. У пошуках ефективної моделі світу експериментували з різними архітектурами як новими, так і модифікованими наявними версіями. Цей пошук призвів до створення нової стохастичної моделі прогнозування об'єктів на відео — прогностичної моделі, яка базується на дискретних латентних змінних, й досягла чудових результатів порівняно з іншими раніше запропонованими моделями [14].

Основний цикл алгоритму *SimPLe* (див. рис. 1 в [14]):

1) *Agent Evaluation* (оцінка агента) — агент починає взаємодіяти з реальним середовищем, дотримуючись останньої політики (ініціалізованої випадковим чином);

2) *World Model* — зібрані спостереження (*Observations*) використовуватимуться для навчання (оновлення) поточної моделі світу;

3) *Agent Training* — агент оновлює політику, діючи за моделлю світу.

Нова політика оцінюватиметься для вимірювання продуктивності агента, а також для збору додаткових даних (повернення до 1). Навчання моделі світу є самоконтрольованим для станів, які виникають, і контрольованим для винагороди [14].

Архітектура нейронної мережі зі стохастичною моделлю з дискретними латентними змінними подана згортковою нейронною мережею *CNN*, щоб агент міг «бачити» і «розуміти» ігрові зображення, частина моделі складається зі згорткових кодера і декодера. Друга частина — це мережа виводу *CNN*, яка апроксимує апостеріорну ймовірність, задану наступним кадром. Під час навчання стохастична модель з прихованими дискретними змінними дискретизує приховані значення з апроксимованої апостеріорної ймовірності. Для цього служить допоміжна рекурентна мережа з пам'яттю *LSTM*, що дає можливість навчання тривалим залежностям. Детальний огляд архітектури, що відображає алгоритм *SimPLe,* подано у [14].

Автори наголошують, що модель є загальною, а не специфічною для *Atari*, і вважають, що вона може впоратися з іншими завданнями візуального передбачення у таких областях, як робототехніка та автономне водіння. Прогностична модель має близько 74 млн. параметрів [14].

Робота [14] просуває передовий досвід *RL* з *Model-Based*, подаючи систему, яка першою успішно впоралася з різними складними іграми в тесті *ALE*. Емпірична оцінка довела, що *SimPLe* значно ефективніший за вибірками, ніж високо-налаштована версія найсучаснішого (*state-of-the-art*) алгоритму *Rainbow*[17]*RL* [15], майже в усіх іграх. Зокре-

[17] У статті [15] розглядаються шість розширень алгоритму *Deep Q-Networks* (*DQN*) та емпірично вивчається їх комбінація. Поєднання *Q*-навчання зі згортковими

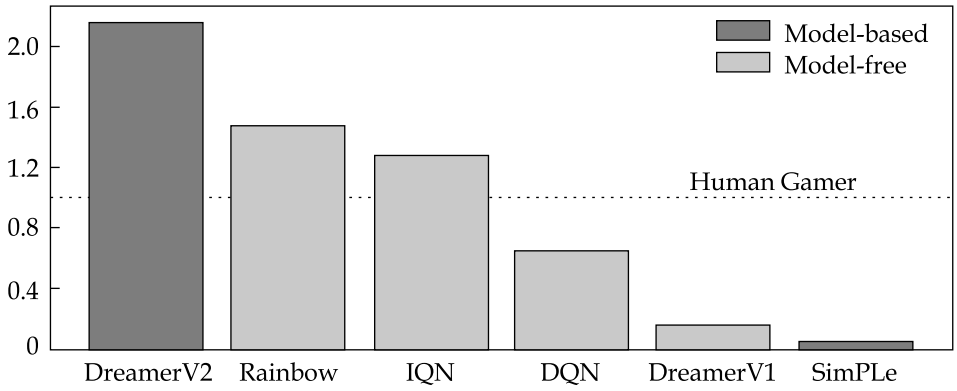

Рис. 6. Продуктивність на тестах *Atari*

ма, в режимі не великої кількості даних у 100 тис. вибірок більш ніж у половині ігор метод *SimPLe* досягає такого результату, для якого *Rainbow* вимагає як мінімум удвічі більше вибірок. Зазначається, що навчання моделей світу призводить до кращих результатів, отриманих за набагато тривалішого навчання порівняно з коротшим навчанням. Через обмеження ресурсів інші абляції (*усунення, виключення*) було зроблено з коротким навчанням моделі [14].

Алгоритм *Dreamer V*2

Істотне поліпшення результатів навчання грі досягнуто у роботах [16, 17], де сучасні *Model-Based approaches* до *RL* зі застосуванням моделей дискретного світу дали змогу створити *Google DeepMind*[18] у співпраці з Університетом Торонто та апробовані на іграх *Atari* безпосередньо з зображень ігрового екрану *DreamerV*2, який перевершує кращі алгоритми без моделей з тим самим обчислювальним та вибірковим бюджетом (рис. 6).

нейронними мережами та відтворення досвіду дали змогу йому навчитися на необроблених пікселях, як грати в ігри на рівні людини. Експериментально показано, що ця комбінація забезпечує найсучаснішу продуктивність на тесті *Atari* 2600 як з точки зору ефективності даних, так і з точки зору кінцевої продуктивності.

[18] *DeepMind* об'єднує дві провідні світові лабораторії *AI Google Brain* і *DeepMind*. Розпочала свою діяльність у 2010 році з міждисциплінарного підходу до створення загальних систем *AI*. Дослідницька лабораторія *Google* об'єднала нові ідеї та досягнення у галузі *ML*, нейробіології, інженерії, математики, моделювання та обчислювальної інфраструктури, а також нові способи організації наукових починань.

Раніше, піонер у галузі глибокого навчання з підкріпленням, використовуючи ігри для тестування своїх систем, одним із її проривів стала програма під назвою *DQN*, яка навчилася грати у 49 різних ігор *Atari* з нуля, просто спостерігаючи за сирими пікселями на екрані та отримуючи команди набрати максимальну кількість очок. У 2015 році *DeepMind* представила *AlphaGo*, першу комп'ютерну програму, яка перемогла чемпіона світу *з Go… https://deepmind.google/*

Нормалізований медіанний рахунок гравця на бенчмарку (тестах) *Atari* з 55 ігор з фіксованими діями на 200 млн кроків (кадрів). *DreamerV2* — перший агент, який навчається виключно в рамках моделі світу, щоб досягти продуктивності *Atari* на рівні людини, демонструючи високу точність своєї вивченої моделі світу. При тому ж обчислювальному бюджеті та часі роботи *DreamerV2* перевершує підсумкову продуктивність найкращих агентів без моделі: *Rainbow* та *IQN*. *SimPLe*, за словами його авторів, оцінювався на легшій підмножині з 36 ігор і навчався для меншої кількості кроків, а додаткове навчання не призвело до подальшого підвищення його продуктивності [14].

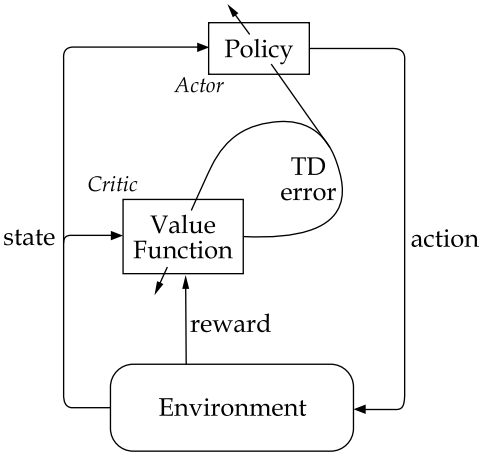

Рис. 7. Архітектура актор-критик

DreamerV2 (друге покоління агента *Dreamer*), інтелектуальний агент *RL*, заснований на моделі світу, який навчається поведінці в прихованому (латентному) просторі моделі, навченої на пікселях. Глибоке посилене навчання дає змогу інтелектуальним агентам з часом покращувати свої рішення. *Model-based approaches* дали змогу вивчати точні моделі із вхідних зображень та використовувати їх для планування подальших дій агента. Моделі світу можуть навчатися на основі меншої кількості взаємодій та уможливити повторне використання знання у кількох завданнях. Вони спрямовані на вивчення шаблонів [18] середовищ, які є загальнішими за будь-яку конкретну задачу, щоб згодом вирішувати завдання ефективніше.

Абстрактна модель світу. *DreamerV2* вивчає модель світу та використовує її для навчання поведінці актора-критика (рис. 7) виключно на основі передбачених траєкторій. Щоб планувати подальші дії агента в невідомих середовищах, агенту необхідно вивчати модель світу на основі досвіду, зокрема зрозуміти динаміку взаємодій із середовищем. Оскільки окремі спостереження за зображеннями зазвичай не розкривають повний стан середовища, воно розглядалося як Марківський процес прийняття рішень (*POMDP*), спостережуваного частково. Для планування процесу прийняття рішень необхідно оцінити тисячі послідовностей дій на кожному кроці агента у часі. Тому запропоновано застосовувати *рекурентну модель простору станів* (*Recurrent State-Space Model*, *RSSM*) [19], використану ними в ранніх розробках [20, 21] для забезпечення прогнозів майбутніх дій у латентному просторі аналогічно до описаних моделей. Цю модель можна розглядати як нелінійний фільтр Калмана чи послідовний *VAE* [19–21].

Агент на основі моделі (*Model-based agen*t) вивчає динаміку середовища за зображеннями та вибирає дії за допомоги швидкого онлайн-планування у прихованому просторі. Щоб досягти високої продуктивності, модель динаміки має точно прогнозувати майбутні нагороди для кількох кроків у часі. Автори робіт [16–17, 19–21] виявили, що як стохастичні, так і детерміновані моделі переходу мають вирішальне значення для успішного планування. Тому використовувалася модель прихованої динаміки з детермінованими та стохастичними компонентами переходу.

Методи актора-критика є природним розширенням ідеї методів градієнтного діапазону для навчання за часовою різницею (*Temporal-Difference Learning*, *TD*). Структура політики відома як актор — використовується для вибору дій, а функція оцінного значення (функція стану-значення) відома як критик, — для оцінки дії актора. Актор, відповідає за вибір дії агента та навчається методами апроксимації функції політики. Після кожного вибору дії критик оцінює новий стан, щоб визначити, чи стало все краще чи гірше, ніж очікувалося. Ця оцінка є помилкою *TD*. Навчання завжди відбувається за політикою: критик повинен дізнатися чи треба критикувати будь-яку політику, якої в даний час дотримується актор. Метод *TD* є комбінацією ідей Монте-Карло та ідей динамічного програмування (*DP*). Навчання може відбуватися безпосередньо на сирому досвіді без моделі динаміки середовища. Методи актора-критика, ймовірно, залишаться актуальними через дві істотні переваги: вимагають мінімальних обчислень для вибору дій; можуть вивчати явно стохастичну політику, тобто. оптимальні ймовірності вибору різних процесів. Ця здатність виявляється корисною у конкурентних і не Марківських випадках [22].

Модель світу навчається обчислювати компактні подання своїх зображень, які виявляють корисні концепції, такі як поточні положення об'єктів у моделі світу, та вивчає, як ці концепції змінюються у відповідь на різні дії. Це дає змогу агенту генерувати абстракції своїх зображень, які ігнорують несуттєві деталі, та забезпечувати прогнози з масовим паралелізмом на одному графічному процесорі (*GPU*). *DreamerV*2 ґрунтується на моделі *RSSM*. Під час навчання кодер перетворює кожне зображення на стохастичне подання, а потім переводить їх у латентний стан. З них витягується лише та інформація, яка саме потрібна для прогнозування. Це робить поведінку агента стійкою до невидимих зображень. З кожного стану декодер реконструює відповідне зображення, щоб отримати загальні представлення. Для забезпечення планування дій агента без генерації зображень предиктор[19] вчиться передбачати стохастичні представлення без доступу до зображень, з яких їх було обчислено.

[19] Предиктор (прогностичний фактор) — один чи декілька чинників, які впливають на предбачувану подію [18]. Виділення або відбір суттєвих ознак і

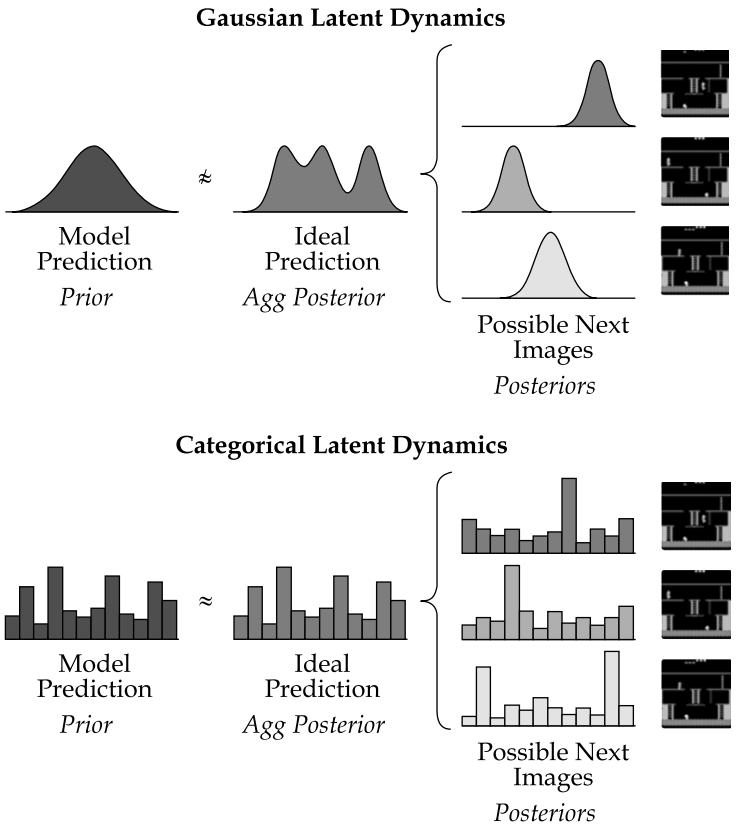

***Рис.* 8.** Подання зображень за допомоги категоріальних змінних замість Гауссових

*DreamerV*2 подано двома новими методами, реалізованими в *RSSM*, які призводять до істотно точнішої моделі світу для навчання успішній політиці. Перший полягає у поданні кожного зображення кількома категоріальними[20] змінними замість Гауссових змінних (рис. 8), застосовуються *PlaNet*, *DreamerV*1 та інші моделі світу, наприклад, наведені у [2]. Це призводить до сприйняття моделі світу в термінах дискретних концепцій і дає змогу більш точно прогнозувати майбутні стани.

Декілька можливих категоріальних зображень рис. 8 можуть бути точно передбачені категоріальним предиктором, на відміну від Гауссівського предиктора, який недостатньо гнучкий.

Кодер (рис. 9) перетворює кожне зображення в 32 розподіли за 32 класами, кожне значення яких визначається автоматично в процесі

укрупнення їх у групи, інакше скорочення числа предикторів — це процедура відкидання незначних змінних з очищеної вибірки перед машинним навчанням та інтелектуальним аналізом даних.

[20] Під категоріальними змінними [18] мають на увазі змінні, які не мають чисельного подання, їм властиво два унікальні значення (бінарні ознаки) або більше. — *Categorical variable — Wikipedia (en) https://en.wikipedia.org/wiki/Categorical_variable*

навчання моделі світу. Вектори *One-hot*[21], створені з цих розподілів, об'єднуються в масив розрідженого представлення та надходять у модель *RSSM*. Подання зображень за допомоги категоріальних змінних дає змогу предиктору точно вивчити розподіл векторів *One-hot* прогнозованих зображень.

Іншим нововведенням, що використовується в *DreamerV*2, є балансування розподілів відстані *KL* (Кульбака-Лейблера). Багато попередніх моделей світу використовують *ELBO*[22], яке заохочує точні реконструкції, зберігаючи при цьому стохастичні апостеріорні уявлення близькими до їх прогнозів (апріорним), щоб упорядкувати обсяг інформації, видобутої з кожного зображення, та полегшити узагальнення. Стохастичні уявлення та їх прогнози можна зробити більш схожими, наблизивши одне до іншого. Однак наближення уявлень до їх прогнозів може бути проблематичним, коли предиктор ще не точний. Балансування розподілів *KL* дає змогу прогнозам швидше досягати уявних результатів, що призводить до більш точних прогнозів. Це є ключем успішного планування.

Вивчення моделі світу. Моделі світу узагальнюють досвід агента в прогностичну модель, яку можна використовувати замість середовища, вивчаючи поведінку агента. Коли вхідні дані є багатовимірними зображеннями, доцільно для прогнозування досліджувати компактні подання станів зображень в латентному просторі [2]. Ці моделі називаються моделями латентної динаміки. Прогнозування полегшує тривалі прогнози та дає змогу ефективно здійснювати тисячі послідовностей компактних станів без необхідності генерувати зображення. Модель світу навчається на основі набору даних минулого досвіду агента, який містить послідовності зображень, дій, винагород і коефіцієнтів дисконтування, і постійно доповнюється.

---

[21] *One-hot*. У *ML* кодування *One-hot* — це метод, що часто використовується для роботи з категоріальними даними. Оскільки багатьом моделям *ML* потрібні числові вхідні змінні, категоріальні змінні необхідно перетворити на частини попередньої обробки — *https://en.wikipedia.org/wiki/One-hot*

[22] *ELBO* (*Evidence lower bound*) нижня межа доказів у варіаційних баєсовських методах є корисною нижньою межею логарифмічної правдоподібності деяких даних, оскільки вона забезпечує гарантію найгіршого випадку для логарифмічної правдоподібності деякого розподілу (наприклад, $p(X)$, що моделює набір даних. Фактична логарифмічна правдоподібність може бути вищою (вказуючи на ще кращу відповідність розподілу), оскільки *ELBO* включає член розбіжності Кульбака-Лейблера (розбіжність *KL*), який зменшує *ELBO* через те, що внутрішня частина моделі неточна, незважаючи на хорошу відповідність моделі в цілому. Таким чином, покращення оцінки *ELBO* вказує або на покращення правдоподібності моделі $p(X)$, або на відповідність компонента, внутрішнього для моделі, або те й інше. Оцінка *ELBO* створює хорошу функцію втрат, наприклад, для навчання глибокої нейронної мережі з метою покращення як моделі загалом, так і внутрішнього компонента — *https://en.wikipedia.org/wiki/Evidence_lower_bound*

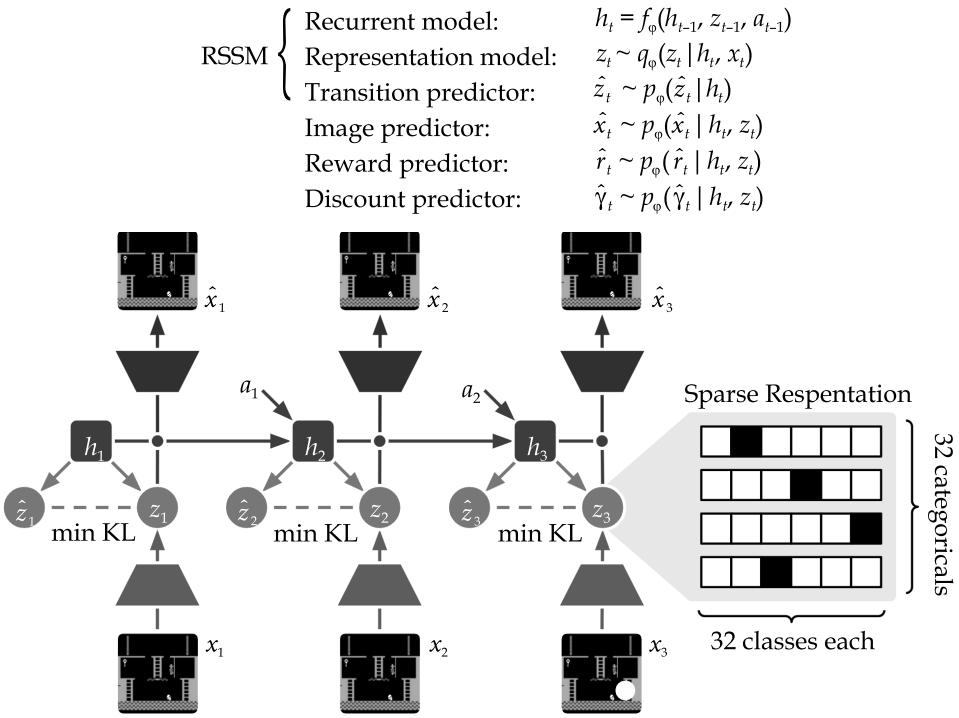

Рис. 9. Навчання моделі світу у *DreamerV*2

Компоненти моделі. Модель світу (рис. 9) складається з кодера та декодера зображень, моделі *RSSM* для вивчення динаміки та предикторів для зображення, скалярних винагород та фактора дисконтування. Компоненти моделі [16]:

Навчальна послідовність зображень x_t кодується за допомоги згорткової нейронної мережі *CNN*. Модель світу підтримує повторювані стани (h_1–h_3), з яких породжуються дії (a_1–a_2) і містять інформацію про зображення (x_1–x_3) через стохастичні уявлення (z_1–z_3). Предиктор переходу (*transition predictor*, $\hat{z}_t$) передбачає уявлення як ($\hat{z}_1$ – $\hat{z}_3$) без доступу до зображень, з яких вони були згенеровані.

RSSM, використовуючи послідовність детермінованих рекурентних станів h_t, на кожному кроці обчислює два види стохастичних розподілів: апостеріорний стан z_t, який містить інформацію про поточне зображення x_t, і апріорне $\hat{z}_t$, яке намагається прогнозувати апостеріорне (без доступу до поточного зображення). Конкатенація (об'єднання) детермінованих та стохастичних станів формує компактний стан моделі — стохастичний стан *DreamerV*2 подається вектором із кількох категоріальних змінних. З апостеріорного стану моделі реконструюється поточне зображення x_t та прогнозується винагорода r_t і фактор дисконтування γ_t. Воно ж використовується для уявних дій агента.

Інформація про втрати *KL* міститься як апріорна інформація, а апостеріорна інформація надходить від зображення. Регуляризація

підвищує стійкість до нових вхідних даних. Вона також заохочує повторне використання наявної інформації попередніх кроків для прогнозування винагород та реконструкції зображень, таким чином вивчаючи тривалі залежності.

Усі компоненти реалізовано як нейронні мережі. Предиктор переходу передбачає наступний стан моделі тільки виходячи з її поточного стану та дії, але без використання наступного зображення. Це дає змогу вивчити поведінку, передбачаючи послідовності станів моделі без необхідності спостерігати або генерувати зображення. Предиктор дисконтування (*Discount predictor*) дає змогу оцінити ймовірність закінчення епізоду щодо поведінки з прогнозів моделі.

Нейронні мережі[23]. Модель *Representation model* реалізована як згорткова нейронна мережа (*Convolutional Neural Network*, *CNN*), за якою слідує багатошаровий персептрон (*Multi-Layer Perceptron*, *MLP*), який отримує вкладення зображення та детермінований рекурентний стан. *RSSM* використовує блок *Gated Recurrent Unit*, *GRU* для обчислення детермінованих рекурентних станів. Стан моделі є конкатенацією детермінованого стану *GRU* і вибірки стохастичного стану. Предиктор зображення є транспонованою *CNN*, а предиктори переходу, винагороди та знижки — *MLP*. Масштаб зображень зменшено у відтінках сірого 84 × 84 до 64 × 64 пікселів, щоб можна було застосувати архітектуру згортки *DreamerV1*.

Мультіагентне глибоке посилене навчання, засноване на моделі

MAMBA DRL. Спираючись на критичний аналіз робіт з *MARL* [4] та досвід застосування *World Models* [2, 14, 16], що дає змогу істотно підвищити ефективність вибірки при навчанні, запроваджено парадигму централізованого навчання агентів у кооперативному середовищі з децентралізованим виконанням (*CTDE*) при використанні *World Models introduces a model-based approach to RL* [23, 24]. Парадигма *CTDE* (*Centralized Training with Decentralized Execution*) — це домінувальниий підхід як для кооперативного, так і для змішаного середовища, завдяки своїй здатності ефективно навчати децентралізованій політиці, є комбінацією незалежного і централізованого *MARL* [4]. У багатоагентному посиленому навчанні агенти, що взаємодіють в кооперативних середовищах, мають одну й ту саму мету та не виключають співпраці між собою. Тоді як у змішаних середовищах повна автономія агентів може бути бажаним результатом, кооперація дає змогу агентам обмінюватися інформацією, хоча, безумовно, ста-

[23] Реалізація *Dreamer V*2 на *Pytorch* у цьому репозиторії міняє по черзі навчання моделі світу, навчання політики та збирання досвіду на *https://github.com/danijar/dreamerv2*

вить під загрозу автономію агентів, але істотно підвищує продуктивність у досягненні поставленого завдання. Багато реальних сценаріїв, таких як автономне управління дронами, передбачають співпрацю агентів для досягнення своїх цілей, зокрема вони можуть ділитися інформацією для кращої координації. Підходи, які використовують цю техніку, зазвичай називають методами комунікації [4].

Автори стверджують [23, 24], що для комунікації між агентами достатньо підтримки моделі світу для кожного агента під час фази виконання, тоді як уявні розгортання можуть використовуватися для навчання, усуваючи необхідність взаємодії з середовищем. Ці властивості дають приклад ефективного алгоритму, що може масштабуватися тільки за кількістю агентів.

Якщо розглядати багатоагентну проблему як одноагентну, це призведе до централізованого підходу, де керування всіма агентами відбувається одночасно на основі глобальної інформації, отримаємо *прокляття розмірності* внаслідок експоненційного зростання простору дій та станів, що накопичується [25]. Спільне адаптування агентів породжує нестаціонарність середовища з погляду одного агента. Вивчити модель світу, яка може підтримуватись децентралізованим чином під час виконання, можна, використовуючи лише спілкування між агентами [23].

Основна складність багатоагентного *RL* полягає в узагальненні на безперервні дії та простір станів, а також масштабування на множину агентів, які приймають рішення. В цілому поточні проблеми *MAMBA DRL* розглянуто в [1, 4].

Аналізуючи можливі шляхи вирішення цього завдання та не порушуючи цілісності концепції *CTDE* було сформульовано ***desideratum*** — «бажані чи необхідні речі» [23]. Розглянемо детальніше основні шляхи.

• **Ефективність вибірки.** У разі одного агента надано емпіричні докази і теоретичні обґрунтування того, що підходи *Model-Based RL* досягають майже оптимальної ефективності вибірки порівняно з *Model-Free*. Ця теорія поширена і на багатоагентне середовище. Однак емпірична продуктивність наразі відсутня через складність моделі, що збільшується, та експоненційне зростання просторів дій та станів.

• **Масштабованість.** Однією з ключових проблем динаміки навчання багатоагентних середовищ є прокляття розмірності. Для його усунення необхідно, щоб модель світу масштабувалася для великої кількості агентів у середовищі, що вимагало б від моделі керування результатним експоненційним зростанням просторів станів та дій. Це б дало змогу опрацьовувати велику кількість переходів як під час навчання, так і під час фаз виведення. Можливе рішення: локальність середовища.

• **Локальність.** Численні дослідження [4] показали, що у великих середовищах можна значно скоротити простір станів, щоб ви-

користовувати лише релевантну інформацію для прийняття рішень агентами. *World Model* повинна мати властивість локальності: спілкування слід обмежити лише найближчими до агента сусідами під час фази виконання.

• **Усунення нестаціонарності.** Буфер відтворення досвіду є популярним підходом, який покращує ефективність вибірки на відміну від методів, що ґрунтуються на політиці (*on-Policy*). Однак у багатоагентних середовищах накопичений досвід швидко стає неактуальним через властиву їм нестаціонарність. Застосування алгоритмів, заснованих на політиці, та моделі світу, яка застосовується для розгортання уявних траєкторій, що починаються зі станів у буфері для поточних політик, апроксимує динаміку середовища та пом'якшує проблеми з неактуальним досвідом.

• **Децентралізоване виконання.** Агенти визначають дії відповідно до своєї індивідуальної політики. Для цього необхідно, щоб вони могли ухвалювати свої рішення незалежно на основі своїх локальних спостережень та повідомлень від інших агентів. Комунікації між агентами достатньо для підтримки моделі світу для кожного агента під час фази виконання, тоді як уявні розгортання можуть використовуватися для навчання, усуваючи необхідність взаємодії з навколишнім середовищем. Парадигма *CTDE* досягає високої продуктивності в багатоагентних доменах, зберігаючи при цьому децентралізацію на етапі виконання [4].

З огляду на математичну формалізацію завдань у мультиагентних системах за посиленого навчання [1] як основу *MAMBA DRL* обрано *state-of-the-art DRL*-алгоритм *DreamerV2* з *World Models* для дискретних середовищ, який модифіковано (рис. 10—11). У мультиагентному середовищі на вхід системи подається n зображень від кожного агента (рис. 10). В модифікований алгоритм *DreamerV*2 — введений *Communication Block* [23, 24]. У ньому використано архітектуру *Transformer* [26] для кодування пар стан-дія (z_t, a_t). Ці кодування потім використовують агенти для оновлення моделі світу та прогнозування дій. Щоб розрізняти агентів у однорідній обстановці використано позиційне кодування [26]. Латентні стани z дії агентів a з попереднього кроку надходять у *Communication Block*, який видає вектор e для кожного агента, оновлюючи модель світу. Для цього є рекурентна нейронна мережа, яка переводить прихований стан h на поточний крок, а вже з цього стану та спостереження агента *0* (реконструюється поточне зображення, тобто картинка), передбачається поточний латентний стан z у кожного агента, з якого можна вирахувати все, що потрібно агенту [23, 24]. На рис. 10 показано, якою інформацією агенти діляться один з одним безпосередньо у середовищі. Все тими ж латентними станами z і діями a, які також передаються в *Communication Block*, за допомоги якого кожен агент оновлює свою власну модель світу.

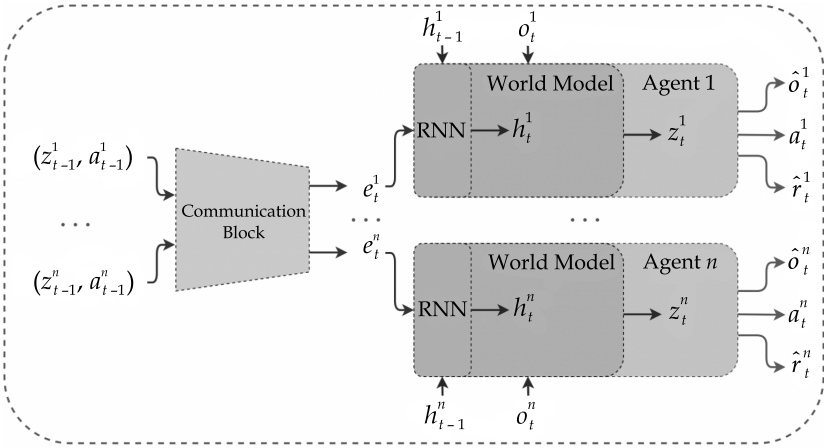

Рис. 10. Фаза навчання

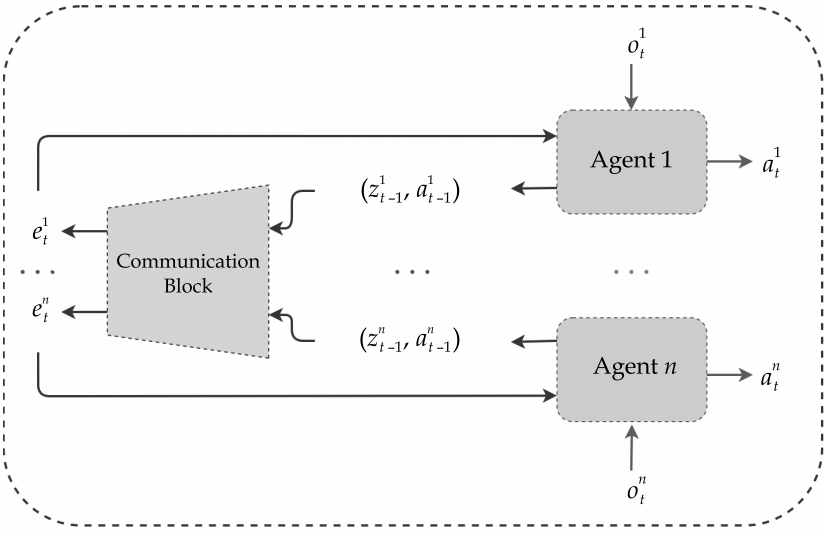

Рис. 11. Фаза виконання

Щодо *Communication Block* з архітектурою *Transformer* [26]. В конфігурації кодер-декодер на складних рекурентних або згорткових нейронних мережах засновано домінуючи моделі[24] послідовної трансдукції[25] (*Sequence to Sequence, Seq2Seq*). Самі ефективні моделі також

[24] Моделі «послідовність-послідовність» *Seq2Seq* — це фундаментальна архітектура глибокого навчання, призначена для вирішення завдань, в яких вхідні та вихідні послідовності можуть відрізнятися за довжиною.

[25] Трансдукція послідовностей — це процес машинного навчання, в якому набір даних перетворюється з однієї форми на іншу. Застосовується в таких областях, як оброблення природної мови, розпізнавання мовлення, оброблення зображень та комп'ютерний зір. Алгоритми перетворення послідовностей базуються на рекурентних та згорткових нейронних мережах, використовує пам'ять *LSTMs*.

пов'язують кодер і декодер через механізм зосередження уваги[26]. В [26] запропоновано використовувати нову просту мережеву архітектуру, *Transformer*, засновану виключно на механізмах зосередження уваги, повністю обходячись без рекурентності та згорток [27, 28]. Експерименти з двома завданнями машинного перекладу показали, що ці моделі краще паралелізуються та вимагають значно менше часу на навчання. *Transformer* добре узагальнюється і на інші завдання, наприклад, відомо його застосування для аналізу англійської електоральної аудиторії як із великими, і з обмеженими навчальними даними.

Автори [23, 24] стверджують, що це перший досвід, який розширює моделі світу за середовищем з довільним числом агентів і розглядають їх як приклад комунікації. Комунікація може бути достатньою для підтримки моделі світу, що дає змогу використовувати централізоване навчання, яке ґрунтується на політиці (*on-Policy*), без взаємодії з середовищем. Для оновлення моделі світу під час фази виконання достатньо обмінюватися дискретними повідомленнями з сусідніми агентами.

Застосування підходу *Model-Based RL* скорочує необхідну кількість вибірок на порядок порівняно з підходами *Model-Free RL* і перевершує сучасні методи в режимі низького обсягу даних у складних середовищах. Емпірично підтверджено, що *MAMBA DRL* досягає хорошої продуктивності, одночасно скорочуючи кількість взаємодій з навколишнім середовищем на порядки порівняно із сучасними підходами *Model-Free* у складних областях *SMAC* та *Flatland*.

Висновки

Посилене навчання за моделлю має суттєві переваги порівняно з безмодельними алгоритмами, оскільки може скоротити необхідну кількість вибірок на порядки. Це веде до підвищення ефективності навчання. Використання ментальної моделі світу дає змогу ефективно досягати мети поставленого завдання шляхом побудови моделі уявлення світу, на кшталт аналога навчання мисленню подібно до мозку людини, замість проведення реальних дорогих випробувань. Штучний інтелект, який генерується глибокими нейронними мережами, здійснює ту ж саму функцію, а великі нейронні мережі вивчають просторові та часові представлення даних для досягнення оптималь-

[26] У базовій моделі кожне введення має бути закодовано у вектор стану фіксованого розміру, оскільки це єдине, що передається декодеру. Щоб дати декодеру більш прямий доступ до даних введення, в [*Bahdanau et al*., 2014] було розроблено механізм зосередження уваги. Достатньо зазначити, що він дає змогу декодеру заглядати у дані введення на кожному кроці декодування, тим самим даючи можливість моделі фокусуватися на різних частинах вхідної послідовності під час генерування кожної частини вихідного коду.

ної стратегії дій агентів й точного опису середовища, тому для роботи потребують даних високої розмірності. Тож посилене навчання має справу з традиційною проблемою *великих даних*.

В роботі проаналізовано зарубіжний досвід розроблення та застосування засобів штучного інтелекту, а саме глибокого посиленого навчання за моделлю для розв'язання проблем поведінки рухомих об'єктів; досліджено задачу керування рухомими об'єктами із застосуванням ментальної моделі світу. Показано, що для розв'язання цієї задачі застосовують великі рекурентні нейронні мережі. На основі проведено аналізу автором запропоновано застосування відомого підходу на основі глибинного посиленого навчання для розв'язання задачі керування рухомими об'єктами, де побудова реальної моделі заміняється ментальної моделі світу, що дозволяє моделювати поведінку рухомих об'єктів не проводячи реальних дорогих випробувань. Це дає змогу більш точно прогнозувати майбутні стани об'єктів.

Проаналізовані в статті роботи пропонують безкоштовну реалізацію алгоритмів на фреймворку машинного навчання *PyTorch*. Результати їх роботи легко відтворити. Ймовірно, це означає, що метод для цієї проблеми не лише досягає високої продуктивності, а й дуже стабільний. У реалізаціях чергуються навчання моделі світу, навчання політики та накопичення досвіду. Загалом ці алгоритми можна адаптувати до вирішення своїх завдань.

ЛІТЕРАТУРА/REFERENCES

1. Oursatyev O.A., Volkov O.Ye. Підходи до створення мультиагентних систем і глибокого посиленого навчання дронів. Information Technologies and Systems, 2025, Vol. 2 (2), 30–55.
2. Ha D., Schmidhuber J. World Models. Can agents learn inside of their own dreams? NIPS 2018, March 27 2018, Oral Presentation. https://doi.org/10.5281/zenodo.1207631
3. Schmidhuber J. On Learning to Think: Algorithmic Information Theory for Novel Combinations of Reinforcement Learning Controllers and Recurrent Neural World Models, 2015, 36 p. https://doi.org/10.48550/arXiv.1511.09249
4. Gronauer S., Diepold K. Multi-agent deep reinforcement learning: a survey. Artificial Intelligence Review, 2021, 1–49. URL: https://link.springer.com/article/10.1007/s10462-021-09996-w
5. Schmidhuber J. Deep Learning in Neural Networks: An Overview. Neural Networks, 2015, Vol. 61, 85–117. https://doi.org/10.1016/j.neunet.2014.09.003
6. Sutton R. S., Barto A. G. Reinforcement Learning: An Introduction. A Bradford Book, The MIT Press Cambridge, Massachusetts, London, England, 2015, 1—337. URL: https://web.stanford.edu/class/psych209/Readings/SuttonBartoIPRLBook2ndEd.pdf
7. Schmidhuber J. Making the World Differentiable: On Using Self-Supervised Fully Recurrent Neural Networks for Dynamic Reinforcement Learning and Planning in Non-Stationary Environments. IDSIA, 1990. URL: https://people.idsia.ch/~juergen/FKI-126-90_(revised)bw_ocr.pdf
8. Schmidhuber J. An on-line algorithm for dynamic reinforcement learning and planning in reactive environments. IJCNN International Joint Conference on Neural Networks, 1990, Vol. 2, 253–258. https://doi.org/10.1109/IJCNN.1990.137723

9. Schmidhuber J. Reinforcement Learning in Markovian and Non-Markovian Environments. IDSIA, 1991. URL: https://sferics.idsia.ch/pub/juergen/nipsnonmarkov.pdf
10. Schmidhuber, J., A Possibility for Implementing Curiosity and Boredom in Model-building Neural Controllers. The First International Conference on Simulation of Adaptive Behavior on From Animals to Animats, 1990. 222–227. MIT Press/Bradford Books, 1991. https://doi.org/10.7551/mitpress/3115.003.0030
11. Arulkumaran K. et al. Deep reinforcement learning: A brief survey, 2017. https://doi.org/10.1109/MSP.2017.2743240
12. Kingma D. P. and Welling M. Auto-Encoding Variational Bayes. Cornell University, 2013. URL: https://pure.uva.nl/ws/files/2511146/162970_1312.6114v10.pd.pdf
13. Hansen (TAO). The CMA Evolution Strategy: A Tutorial. 2016, 1–39. URL: https://arxiv.org/abs/1604.00772v2
14. Kaiser L. *et al*., Model-Based Reinforcement Learning for Atari. ICLR 2020, 1–28. URL: https://arxiv.org/abs/1903.00374
15. Hessel M. et al. Rainbow: Combining Improvements in Deep Reinforcement Learning. AAAI 2018. https://doi.org/10.1609/aaai.v32i1.11796
16. Hafner D. et al. Mastering Atari with Discrete World Models. ICLR 2021, 1–26. URL: https://arxiv.org/abs/2010.02193
17. Mastering Atari with Discrete World Models. February 18, 2021, Posted by Hafner D., Google Research. URL: https://research.google/blog/mastering-atari-with-discrete-world-models/
18. Oursatyev O. Data Research in Industrial Data Mining Projects in the Big Data Generation Era. Control Systems and Computers, Issue 3, 33–54. [In Ukrainian: Урсатьєв О.А., Дослідження даних у промислових data-mining-проєктах в епоху генерації великих даних]. URL: https://doi.org/10.15407/csc.2023.03.033
19. Hafner D. et al., Learning Latent Dynamics for Planning from Pixels, 2018. URL: https://arxiv.org/abs/1811.04551
20. Introducing PlaNet: A Deep Planning Network for Reinforcement Learning, Febr. 2019, Posted by Danijar Hafner. URL: https://research.google/blog/introducing-planet-a-deep-planning-network-for-reinforcement-learning/
21. Introducing Dreamer: Scalable Reinforcement Learning Using World Models. March, 2020. Posted by Danijar Hafner. URL: https://research.google/blog/introducing-dreamer-scalable-reinforcement-learning-using-world-models/
22. Sutton R.S., Barto A.G. Reinforcement Learning: An Introduction. MIT Press, Cambridge, Massachusetts, London, England, 2018, 526 p. URL: https://www.google.com/url?sa=t&source=web&rct=j&opi=89978449&url=https://www.andrew.cmu.edu/course/10-703/textbook/BartoSutton.pdf
23. Egorov V., Shpilman A. Scalable Multi-Agent Model-Based Reinforcement Learning. The 21st International Conference on Autonomous Agents and Multiagent Systems (AAMAS), 381–390. https://dl.acm.org/doi/abs/10.5555/3535850.3535894
24. Egorov V., Shpilman A. Scalable Multi-Agent Model-Based Reinforcement Learning. ArXiv, 2022. URL: https://arxiv.org/abs/2205.15023v1
25. Sunehag P. et al. Value-Decomposition Networks For Cooperative Multi-Agent Learning. ArXiv, 2017. URL: https://arxiv.org/abs/1706.0529625.
26. Vaswani A. et al.Attention Is All You Need. ArXiv, 2017. URL: https://arxiv.org/abs/1706.03762
27. Bahdanau D., Cho K., Bengio Y. Neural Machine Translation by Jointly Learning to Align and Translate. ArXiv. URL: https://arxiv.org/abs/1409.0473
28. Cho K. et al. Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation. https://doi.org/10.3115/v1/D14-1179

Отримано/Received 20.08.2025

O.A. Oursatyev, PhD (Engineering), Senior Researcher, Leading Researcher,
Institute of Information Technologies and Systems of the NAS of Ukraine,
40, Hlushkova Akad. ave., Kyiv, 03187, Ukraine
https://orcid.org/0009-0009-8323-0525
aleksei@irtc.org.ua

O.Ye. Volkov, PhD (Engineering), Senior Researcher, Director,
Institute of Information Technologies and Systems of the NAS of Ukraine,
40, Hlushkova Akad. ave., Kyiv, 03187, Ukraine
https://orcid.org/0000-0002-5418-6723
alexvolk@ukr.net

REINFORCED LEARNING OF NEURAL NETWORKS IN IMAGINATION IN CONTROL SYSTEMS OF UNMANNED MOVING OBJECTS

Introduction. Artificial intelligence based on recurrent (cyclic) neural networks (RNNAI) performs the function of learning to think like the human brain. Large recurrent neural networks can learn spatial and temporal representations of data. The model-based reinforcement deep learning method requires high-dimensional input data to achieve an optimal agent action strategy and an accurate representation of the environment.

However, reinforcement learning faces the traditional problem of large data: algorithms rarely work with high-dimensional data. First, traditional RL algorithms have difficulty learning the weights of a large model in the task of assigning grades or credits, especially at the end of a sequence of algorithmic steps. The weight distribution problem solves the problem of determining which steps should be considered rewarding or punishing for the final result. Second, the performance of physical simulators, whose task is to anticipate changes in the environment, is low. The selection of optimal actions is carried out through separate task scheduling. This requires simulating many random actions and selecting the best one. This is classic Model-Based RL. However, with large dimensions and long chains, the number of possible actions becomes too large to enumerate. Therefore, the author explored an approach that uses a mental model of the world as the environment model. Neural networks in this approach are trained similarly to how they are trained in the human brain. This approach achieves the control goal by constructing a model of the world instead of conducting costly real-world testing.

Purpose of this article is to analyze current international experience in developing and implementing analytical platforms for controlling moving objects in single- and multi-agent systems. This control is achieved using artificial intelligence generated by deep neural networks, training the model through reinforcement learning in unknown, partially observable environments.

Methods. An approach to controlling moving objects in single- and multi-agent systems using neural networks and a mental model of the world is considered.

Results. An approach to controlling moving objects using neural networks and a mental model of the world is investigated. This article analyzes international experience in the development and application of artificial intelligence tools, specifically deep reinforcement learning, to solve problems of moving object behavior in unknown, partially observable environments.

Conclusions. Based on this analysis, the author proposes the application of a well-known approach based on deep reinforcement learning to the problem of controlling moving objects. This approach achieves the control goal by constructing a model representation of the world instead of conducting costly real-world testing.

Keywords*: unmanned moving objects, deep reinforcement learning, World Models, mental model of the world, neural networks, agent learning, multi-agent environment, recurrent state space model.*