МАГІСТЕРСЬКА РОБОТА
МР.ІПм – 22.00.00.000 ПЗ

Група ІПм-22-5

Балабан Степан

2024

Івано-Франківський національний технічний університет нафти і газу

Інститут інформаційних технологій

Кафедра інженерії програмного забезпечення

Балабан Степан Михайлович

(прізвище, ім'я, по батькові)

УДК 004.942

(індекс)

МАГІСТЕРСЬКА РОБОТА

Моделі та методи застосування машинного навчання для контролю технологічних процесів

(назва роботи)

Інженерія програмного забезпечення

(назва освітньої програми)

121 - Інженерія програмного забезпечення

(шифр і назва спеціальності)

Балабан С.М.

(підпис, ініціали та прізвище здобувача освітнього ступеня)

Науковий керівник Дмитрик Тарас Богданович, асистент

(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

Допущено до захисту

В.о. завідувача кафедри

доц. Бандура В.В.

(посада) (підпис) (дата) (ініціали та прізвище)

Рецензент

(посада) (підпис) (дата) (ініціали та прізвище)

Робота містить результати власних досліджень. Використання ідей, результатів і текстів інших авторів мають посилання на відповідне джерело

Івано-Франківськ – 2024

Івано-Франківський національний технічний університет нафти і газу

Інститут інформаційних технологій

Кафедра інженерії програмного забезпечення

Освітній рівень магістр

Спеціальність 121 – Інженерія програмного забезпечення

ЗАТВЕРДЖУЮ:
В.о. зав. кафедрою **ІПЗ**
доц. В.В. Бандура
“ 04 ” вересня 2023 р.

ЗАВДАННЯ

НА МАГІСТЕРСЬКУ РОБОТУ СТУДЕНТУ

Балабану Степану Михайловичу

(прізвище, ім’я, по-батькові)

1. Тема магістерської роботи **“**Моделі та методи застосування машинного навчання для контролю технологічних процесів”
керівник проекту (роботи) Дмитрик Тарас Богданович, асистент
затверджені наказом закладу вищої освіти від “ 18 ” грудня 2023 р. № 738/7

2. Строк подання студентом проекту (роботи) 25 січня 2024 р.

3. Вихідні дані до проекту (роботи) Теоретичні концепції та формальні моделі побудови та функціонування інформаційних та програмних технологій певного класу

4. Зміст розрахунково - пояснювальної записки(перелік питань, які потрібно розробити)

1. Теоретичні основи застосування машинного навчання в промисловості
2. Побудова методології застосування Q-НАВЧАННЯ
3. Аналіз отриманих результатів

5. Перелік графічного матеріалу (з точним забезпеченням обов’язкових креслень)

1. Діаграма, на якій показано взаємодію між агентом МНта його середовищем (рис. 1.1)
2. Простий приклад зворотного поширення RNN з функціями активації (рис 1.2)
3. Архітектура модуля машинного навчання під час руху до метасимулятора (рис.2.2)
4. Продуктивність типового запуску з використанням шляхів (рис. 3.1)
5. Прогон із використанням сегментів і вибору дій на основі SARSA (рис. 3.5)

6. Консультанти з роботи із зазначенням розділів роботи, що їх стосуються

Розділ	Консультант	Підпис, дата
Нормоконтроль	доц., к.т.н. Вовк Р.Б.	
Перевірка на плагіат	доц., к.т.н. Вовк Р.Б.	

7. Дата видачі завдання **4 вересня 2023 р.**

Керівник _____
(підпис)

Завдання прийняв до виконання_____
(підпис)

КАЛЕНДАРНИЙ ПЛАН

№ п/п	Назви етапів магістерської роботи	Строк виконання етапів роботи	Примітка
1	Підбір і вивчення літератури	01.10.2023	виконано
2	Теоретичні основи застосування машинного навчання в промисловості	10.11.2023	виконано
3	Побудова методології застосування Q-навчання	01.12.2023	виконано
4	Аналіз отриманих результатів застосування методів машинного навчання	15.12.2023	виконано
5	Затвердження пояснювальної записки роботи завідувачем кафедри	25.01.2024	виконано

Студент - магістр _____
(підпис)

Керівник _____
(підпис)

АНОТАЦІЯ

Магістерська робота: 76 с., 25 рис., 43 джерела

Тема: Моделі та методи застосування машинного навчання для контролю технологічних процесів

Об'єкт дослідження: методи машинного навчання

Мета роботи: Визначення ефективних моделей та методів застосування машинного навчання для контролю та оптимізації технологічних процесів у промислових системах з використанням концепції цифрового виробництва.

Предмет дослідження: моделі та методи застосування машинного навчання для контролю технологічних процесів

Результати дослідження:

Розроблені та впроваджені ефективні моделі та методи машинного навчання для автоматизації та оптимізації контролю технологічних процесів в промислових системах з використанням підходу цифрового виробництва – концепції технологічної підготовки виробництва в єдиній віртуальному середовищі з допомогою інструментів планування, перевірки і моделювання виробничих процесів.

Висновок:

В ході дослідження було виявлено, що застосування моделей та методів машинного навчання для контролю технологічних процесів у промислових системах є актуальним та перспективним напрямком досліджень. Аналіз технологічних процесів виявив їхню складність та динаміку, що робить актуальним використання адаптивних підходів.

СИМУЛЯТОР, ШТУЧНИЙ ІНТЕЛЕКТ, РЕКРЕНТНІ НЕЙРОННІ МЕРЕЖІ, ПРОГРАМНЕ ЗАБЕЗПЕЧЕННЯ, АЛГОРИТМ, ФРЕЙМВОРК, АГЕНТ, ТЕХНОЛОГІЧНИЙ ПРОЦЕС, МАШИННЕ НАВЧАННЯ

ANNOTATION

Master's thesis: 76 pages, 25 figures, 43 sources

Topic: Models and methods of applying machine learning to control technological processes

Research object: machine learning methods

Purpose of work: Determination of effective models and methods of application of machine learning for control and optimization of technological processes in industrial systems using the concept of digital production.

Subject of research: models and methods of application of machine learning for control of technological processes

Research results:

Developed and implemented effective models and methods of machine learning for automation and optimization of control of technological processes in industrial systems using the approach of digital production - the concept of technological preparation of production in a single virtual environment with the help of tools for planning, verification and modeling of production processes.

Conclusion:

During the research, it was found that the application of models and methods of machine learning to control technological processes in industrial systems is a relevant and promising direction of research. The analysis of technological processes revealed their complexity and dynamics, which makes the use of adaptive approaches relevant.

SIMULATOR, ARTIFICIAL INTELLIGENCE, RECURRENT NEURAL NETWORKS, SOFTWARE, ALGORITHM, FRAMEWORK, AGENT, TECHNOLOGICAL PROCESS, MACHINE LEARNING

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ, СКОРОЧЕНЬ І ТЕРМІНІВ

ML - машинне навчання

ШІ - штучний інтелект

RNN - рекурентні нейронні мережі

TCS - системи керування дорожнім рухом

DNN - глибокі нейронні мережі

DRF - розподілений випадковий ліс

GBM - машина посилення градієнта

GLM- узагальнена лінійна модель

KNN - K Nearest Neighbor

SVR - Support Vector Regression

ANN- Artificial Neural Network

ВСТУП

Актуальність роботи

Оптимізація виробництві в умовах цифрового виробництва, є актуальним завданням на сьогоднішній день. Сучасні підходи засновані на використанні людських ресурсів. Дані підходи є стандартними, це дуже часто веде до помилкових результатів.

Застосування моделей та методів машинного навчання для контролю технологічних процесів є вкрай актуальною в сучасному промисловому середовищі. Існуючі технології стають все більш складними та динамічними, вимагаючи вдосконалення систем контролю та управління.

Деякі ключові аспекти актуальності цієї теми включають:

- Складність технологічних процесів: Промислові виробництва розвиваються, впроваджуючи складні технологічні процеси, які важко контролювати за допомогою традиційних методів. Машинне навчання може забезпечити адаптивні та ефективні рішення для оптимізації цих процесів.

- Зростання обсягів даних: За останні роки значно зросла кількість доступних даних. Машинне навчання може використовувати ці великі обсяги даних для створення точних та прогностичних моделей для контролю технологічних процесів.

- Потреба в оптимізації: Промислові підприємства постійно шукають способи оптимізації виробничих процесів з метою підвищення продуктивності, зниження витрат та забезпечення якості виробництва. Машинне навчання може допомогти в досягненні цих цілей.

- Автоматизація та індустрія 4.0: Розвиток концепції індустрії 4.0 передбачає повну автоматизацію та взаємодію систем. Застосування машинного навчання стає ключовим елементом для досягнення цих цілей.

- Ріст конкуренції: В умовах глобального ринку важливо мати ефективні та інноваційні технології.

Застосування новітніх методів машинного навчання може стати конкурентною перевагою для підприємств. Отже, дослідження в галузі застосування моделей та методів машинного навчання для контролю технологічних процесів відзначається високою актуальністю і може сприяти вдосконаленню сучасних промислових систем.

Традиційні методи контролю технологічних процесів: Ефективність в стабільних умовах, методи, такі як PID-регулятори, можуть бути ефективними у стабільних умовах, де динаміка системи вже добре відома та прогнозується. Висока вартість налаштування, налаштування традиційних систем може бути трудомістким та вимагати значних зусиль та експертної експертизи. Обмежена адаптивність, традиційні підходи мають обмежену адаптивність до змін у середовищі, що може призвести до менш ефективного контролю у динамічних умовах.

Моделі машинного навчання можуть ефективно адаптуватися до змін у технологічних процесах, що робить їх ефективними в умовах змінюючогося середовища. Машинне навчання дозволяє створювати точні прогностичні моделі, що поліпшує передбачуваність та управління процесами. Машинне навчання може оптимізувати стратегії контролю в реальному часі, що призводить до покращення продуктивності та зменшення витрат. Для створення моделей машинного навчання не завжди потрібно глибоке експертне знання про технічні деталі процесів. Необхідність в якісних даних: Як у традиційних, так і в сучасних методах, які використовують машинне навчання, важливо мати якісні та достовірні дані для ефективного контролю технологічних процесів. Необхідно систематично перевіряти та валідувати розроблені моделі для забезпечення їхньої точності та ефективності. Порівнюючи обидва підходи, можна визначити, що машинне навчання має переваги у динамічних та змінних умовах, а традиційні методи можуть бути ефективними у стабільних умовах. Ефективне поєднання обох підходів може дати комплексніші та ефективніші стратегії контролю технологічних процесів.

Мета і задачі дослідження

Метою магістерської роботи є визначення ефективних моделей та методів застосування машинного навчання для контролю технологічних процесів у промислових системах.

Досягнення мети включало розв’язання таких **задач**:

- Аналіз технологічних процесів,
- Розробка моделей машинного навчання,
- Оптимізація контролю процесом засобами машинного навчання,
- Проведення імітаційного моделювання,
- Оцінка ефективності пропонованої методології.

Об’єктом дослідження є методи машинного навчання.

Предметом дослідження є моделі та методи застосування машинного навчання для контролю технологічних процесів.

Методи дослідження

Для досягнення поставленої мети використанні такі методи:

- статистичний аналіз: використання статистичних методів для вивчення особливостей та взаємозв'язків у технологічних процесах.
- моделювання,
- методи машинного навчання,
- імітаційне моделювання.

Наукова новизна отриманих результатів

Отримані результати вносять вагомий вклад у практичне вдосконалення технологічних процесів та розвиток наукових підходів до їхнього контролю в промислових умовах. І полягають у розробці адаптивних моделей машинного навчання, оптимізації стратегій контролю за допомогою Q-навчання, впровадження методології тестування в промисловості, застосування метасимулятора для вирішення проблем симуляційних платформ. Також розглянуто концепцію оптимізації технологічних процесів на основі підходу цифрового виробництва.

Практичне значення одержаних результатів

Практичне значення отриманих результатів полягає в підвищенні ефективності виробничих процесів, покращення якості продукції, економії ресурсів, швидкому реагування на зміни, підвищенні конкурентоспроможності. Впровадження інноваційних підходів до контролю технологічних процесів забезпечить компанії конкурентні переваги на ринку.

Особистий внесок студента

Розроблені та впроваджені ефективні моделі та методи машинного навчання для автоматизації та оптимізації контролю технологічних процесів в промислових системах та цифровому виробництві.

Структура та обсяг магістерської роботи

Магістерська робота викладена на 75 сторінках друкованого тексту, який складається із вступу, трьох розділів, висновків, списку використаних джерел (43 найменування). Робота містить 25 рисунків.

РОЗДІЛ 1.
АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ ЗАСТОСУВАННЯ МАШИННОГО НАВЧАННЯ В ПРОМИСЛОВОСТІ

1.1 Цілі та задачі дослідження

Метою цієї роботи є розробка модуля прийняття рішень, який може автоматично навчитися створювати правила руху для координації автономних машин для видобутку в заданій зоні видобутку. Модуль повинен мати можливість розпізнавати, коли ось-ось виникне взаємоблокування, і запобігати його виникненню. Цей модуль буде оцінено на основі точності його рішень і часу, необхідного для того, щоб стати достатньо точним. Частота помилок 1 взаємоблокування на 24 години є цільовою продуктивністю модуля. Критерії для оцінки будуть визначені в ключових - показниках продуктивності (KPI), які оцінюють продуктивність рішення, пов’язану з частотою тупиків трафіку, які системні оператори повинні вирішити вручну. Навчання час буде досліджено, щоб визначити, чи можна навчити рішення до прийнятної точності в розумні часові рамки.

Оцінка та вибір відповідного алгоритму ML для цієї конкретної області застосування буде базуватися на попередній роботі, виконаній у цій галузі. Після вибору одного або кількох відповідних алгоритмів буде розроблено прототип модуля прийняття рішень, який визначає розташування автономної машини та активно вивчатиме, як уникнути тупикових ситуацій з іншими машинами. Оцінка можливих економічних факторів буде обговорена в аналізі результатів.

Обсяг проекту та розмежування

Через часові рамки цього проекту буде впроваджено та оцінено лише один алгоритм ML. Використовувані платформи та параметри даних будуть доступними за допомогою поточної платформи моделювання AF.

1.2 Штучний інтелект і машинне навчання

У книзі «Глибоке навчання» Гудфеллоу та ін. [1], штучний інтелект загалом - визначається як комп'ютерна програма зі здатністю сприймати цільове середовище, змодельоване чи ні. На основі даних про об'єкти та їх стани або інші фактори в цьому середовищі ШІ виконуватиме (заздалегідь визначені) дії з найвищою ймовірністю досягнення визначеної мети.

Машинне навчання також визначається як програма, яка використовується для виконання певних завдань, але це робиться з використанням наборів даних, часто спрощених наборів даних і розпізнавання шаблонів для формулювання алгоритму замість використання явного програмування або інструкцій після виконання умов [1]. Причиною використання машинного навчання є те, що моделювання ефективного алгоритму може бути неможливим для виконання вручну.

У тематичному дослідженні щодо впровадження підходу ML для короткострокового прогнозування транспортних потоків між штатами 64 у Міссурі [2] було реалізовано чотири різні моделі ML. Моделями були глибокі нейронні мережі (DNN), розподілений випадковий ліс (DRF), машина посилення градієнта (GBM) і узагальнена лінійна модель (GLM). Точність розроблених моделей оцінювали за допомогою трьох індексів похибки, коефіцієнта детермінації (R2), середньої абсолютної похибки (MAE) та середньоквадратичної похибки (RMSE). П'ятимісячні дані, які складалися з швидкості, потоку, зайнятості та часу доби, були використані для навчання та перевірки моделей. Час обчислення для всіх моделей становив менше однієї секунди. Час навчання моделі DNN був у середньому в 33 рази довший, ніж в інших моделях – 34 хвилини. DRF трохи перевершив інші моделі. Під час перевірки прогнозування потоку транспорту протягом 24 годин у 5-, 10- та 15-хвилинних вікнах призвело до MAE 7,522%, 7,131% та 7,197% відповідно.

Показник RMSE дав результати 11,123%, 10,455% і 10,738%. В [3] провели дослідження підходу ML для прогнозування транспортного потоку,

щоб зменшити затори на урбанізованих магістральних дорогах. Ці дороги описуються як постійно завантажені, призначені для поїздок і перевезень вантажів, мають високу пропускну спроможність і здатні обробляти великі обсяги транспорту, але мають обмежені точки в'їзду та виїзду. Було використано три моделі ML: K Nearest Neighbor (KNN), Support Vector Regression (SVR) і Artificial Neural Network (ANN). Моделі були застосовані до реальних наборів даних. Дослідження показало, що модель ШНМ дає найточніші прогнози з RMSE 16,95%. З усіма трьома моделями можна було б зробити кращі прогнози, якби замість загальної кількості машин використовували індивідуальну загальну кількість класів машин . RMSE ШНМ знизився в цьому випадку до RMSE 16,56%. Більш глибока архітектура була рекомендована як основа для майбутніх досліджень, оскільки вона може створити точнішу функцію прогнозування.

У дослідженні щодо короткострокового прогнозування транспортного потоку була використана довгострокова нейронна мережа з повторюваною короткочасною пам'яттю (LSTM RNN), оскільки характеристики довгострокового навчання LSTM RNN роблять її бажаною для цього випадку використання. У цьому дослідженні бл. Було використано 18000 точок вибірки даних реального світу в часовому ряді, що охоплює більше двох місяців. Ці дані були зібрані з п'яти станцій і включали інформацію про заповненість, швидкість і потік транспорту. Певна станція використовувалася як об'єкт для прогнозування потоку транспорту з використанням комбінацій цих точок даних та інформації з сусідніх станцій. Вони дійшли висновку, що сам по собі транспортний потік може призвести до прийнятної точності прогнозування, а комбінації точок даних і включення даних сусідніх станцій призвели до кращої точності, де найкращий MAE із запропонованих моделей становив 9,26%.

Дослідження, проведене В [5], мало на меті спрогнозувати як короткостроковий, так і довгостроковий транспортний потік у Сеулі,

використовуючи набори статистичних і синтетичних даних у реальному часі, щоб зменшити затори. Використані набори даних збиралися протягом двох років. Дані в реальному часі збиралися кожні п’ять хвилин з автомобілів за допомогою навігаційної програми, а статистичні дані з GPS-даних автобусів/таксі кожні 30 хвилин. Дані складалися з таких речей, як погода на той час, середня швидкість, інформація про аварії та інформація про повороти. Це призвело до наборів даних із 210 526 і 17 544 точок даних відповідно. Синтетичні дані мали такий самий обсяг, як і статистичні дані. Результати показали, що застосування LSTM RNN у цій програмі призвело до прогнозованого потоку транспорту, подібного до фактичного спостережуваного потоку транспорту зі статистичними даними. Прогнози щодо іонів з використанням синтетичних даних відхилялися у вечірні години пік, а прогнози з використанням даних у реальному часі відхилялися в години поза піком.

У дослідженні [6] автори перевіряють ефективність використання функції апроксимації архітектури CMAC у поєднанні з нейронною мережею для навчання з підкріпленням. Було зроблено висновок, що нейронна мережа, яка реалізує CMAC, у більшості випадків мала швидший час навчання та показала, що вона здатна узагальнити навчання. Один початковий стан призводив до приблизно 86-кратного зменшення кількості випробувань, необхідних для досягнення успішних результатів навчання, тоді як інший стан призводив до 14-кратного збільшення кількості випробувань, необхідних для досягнення результату.

У дослідженні [7] показали, що за допомогою алгоритму посилення під назвою Q-learning можна скоротити час зупинки, спричинений заторами на зустрічних з’їздах до різних автострад у Торонто, на 26,7% порівняно з немає контролю [7]. Порівняно з одним із найбільш часто використовуваних алгоритмів керування рампою, ALINEA, вони виміряли зменшення на 15,71% часу зупинки та зменшення часу подорожі на 4,69% у найкращому випадку

ALINEA. Це шляхом перенаправлення дорожнього руху за допомогою змінних знаків у критичних місцях, щоб інформувати водіїв про повторювані чи одноразові затори. Агент у цьому експерименті отримував винагороду, якщо було виявлено зменшення затримки, і покарання, якщо було виявлено збільшення. Через велику кількість станів у їхній моделі симуляцію було запущено 15 000 разів, де кожен запуск складався з 1,5 годин змодельованого руху в годину пік з інцидентом, який блокував рух на 15 хвилин, доданим у першій половині прогону. Для зменшення простору станів дослідники використовували алгоритм CMAC. Він працює шляхом відображення безперервних станів, які використовувалися, у плитках, що перекриваються, для узагальнення відвіданих станів.

У [8] показали у своєму дослідженні контролю міського руху, що за допомогою алгоритму Q-навчання середній час очікування значно зменшився на 64 перехрестях у центрі Дубліна. Крім того, за допомогою підходу до спільного навчання з підкріпленням час зменшився ще більше. Співпраця означає, що різні агенти, які контролюють одне перехрестя, спілкуються, щоб контролювати потік транспорту, а не кожен агент має інформацію лише про своє власне перехрестя. Моделювання базувалося на 4032 машинах, які рівномірно вставлялися в систему протягом приблизно 133 хвилин. RL без використання спільного підходу був на 15,9-45% кращим, ніж доступні на той час рішення, а розширення його до кооперативного -покращило час очікування на 22% порівняно з автономним методом RL.

Іншим можливим рішенням, запропонованим в [9] для оптимізації сигналів світлофора, є використання генетичного алгоритму. Це ще один тип алгоритму навчання з підкріпленням, який базує своє навчання на створенні нових поколінь агентів навчання, які базуються на попередньому запуску симуляції. Вони виявили, що генетичний алгоритм працює краще при середньому та високому обсязі трафіку, ніж як звичайні, так і випадкові інтервали сигналу, але трохи гірше, ніж регулярні інтервали для розріджених

обсягів.

1.3 Дослідження алгоритмів машинного навчання

В [10] навчання з підкріпленням визначається як тип машинного навчання, мета якого полягає в тому, щоб дізнатися, що робити в певних ситуаціях, іншими словами, зіставлення ситуацій з діями, які максимізують числове значення. Програма, яка намагається розв'язати задану проблему, називається учнем або агентом. Агент навчається, взаємодіючи зі своїм середовищем, і отримує винагороду або покарання залежно від результату цієї дії.

Для впровадження учня або агента потрібна політика, яка визначає, як учень повинен діяти в різний час [11]. Політика лежить в основі процесу навчання підкріплення, оскільки вона визначає поведінку агента. Це можна описати як карту дій щодо заданого стану середовища. Політикою може бути проста таблиця пошуку або більш складна, наприклад функція пошуку. Агенту також потрібен сигнал винагороди, щоб знати результат виконаної дії. Винагорода використовується для визначення мети процесу навчання. Щоб отримати винагороду, агент повинен взаємодіяти зі своїм оточенням. Коли він робить це, як показано на рисунку 1.1, він отримує нагороду та наступний стан у відповідь.

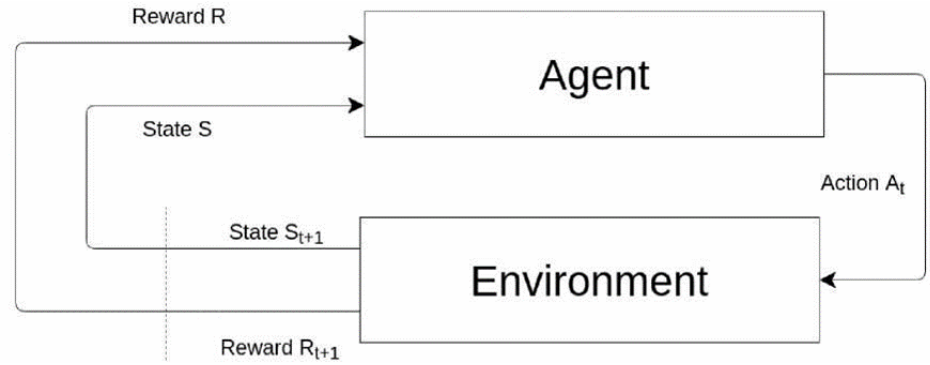

Рис. 1.1. Діаграма, на якій показано взаємодію між агентом машинного навчання та його середовищем

Потім вони використовуються для оновлення політики, яку використовує агент з часом. Агент не може сам викликати ці винагороди, він повинен або виконати дію, яка винагороджується негайно, або привести до стану, який може. призвести до додаткових винагород. Сигнал винагороди – це те, що агент використовує, щоб знати, що добре робити в короткостроковій перспективі, але в довгостроковій перспективі, щоб досягти бажаної мети, він використовує функцію цінності [11]. Функція значення намагається оцінити значення поточного стану середовища. Значення стану — це кількість винагород, які агент може очікувати отримати в майбутньому від цього стану. Це те, що дозволяє агенту RL робити вибір, який не дає негайної винагороди, але веде до стану з потенціалом надання більшої винагороди.

Оскільки мета агента полягає в тому, щоб максимізувати винагороди, які він отримує з часом, метою процесу навчання з підкріпленням стає оптимізація політики для даного стану таким чином, щоб вона приносила максимальну кількість винагород [7]. Q-навчання є простим і широко використовуваним алгоритмом для цього [12]. Гарантовано зближення до оптимальної політики з часом [13]. Q-навчання — це так званий алгоритм «поза політикою», тобто він не повністю покладається на попередні результати, а скоріше досліджує різні варіанти, щоб знайти найкращий. Еквівалент Q-навчання «on-policy» називається SARSA. Різниця полягає в тому, що SARSA завжди вибирає найкращу дію на основі політики. У той час як Q-навчання зійдеться до оптимальної політики за умови достатньої кількості ітерацій, SARSA знайде приблизно оптимальну політику.

Пари «Дія-стан», позначені як Q(s, a), є оціночним значенням певної дії для даного стану та описують суму очікуваних майбутніх виграшів, які дія дасть у цьому стані. Q-навчання працює шляхом вибору дії ае^ з огляду на поточний стан se£ на основі політики, де A і S є всіма можливими діями та станами відповідно [12]. Нагорода та наступний стан від цієї дії повертаються агенту, а значення Q(s, a) оновлюється за допомогою:

$$Q(St, At) <= Q(St, At) + a[Rt+ 1 + yMAXaQ(St+ 1 , a) - Q(St, At)] \quad (1.1)$$

Вивчене значення — це негайна винагорода плюс дисконтована оцінка оптимального значення наступного стану мінус поточне Q(s, a). Швидкість навчання можна регулювати з часом, щоб змусити алгоритм частіше досліджувати дії з потенційно нижчою винагородою на початку процесу навчання, щоб мати можливість знаходити потенційно -кращі дії в цьому стані

Значення швидкості навчання а коливається між o<a<i. K. De Asis та інші пояснюють у «Rein forcement Learning and Artificial Intelligence Laboratory», що низьке значення сповільнює навчання алгоритму, але наближається до оптимального рішення, тоді як високе значення матиме кращу початкову продуктивність, але не наближається до оптимального розв'язку [15]. Коефіцієнт дисконтування у визначає, скільки коштує майбутня винагорода порівняно з негайною. Іншими словами, скільки вона важить вартість майбутньої держави [16].

Для регулювання швидкості навчання можна використовувати правило розміру кроку в рівнянні (1.3), описане А. Госаві в книзі «Оптимізація на основі моделювання» [17]. Де an+1 — це швидкість навчання для n-го відвідування стану, T1 — вихідне значення для а, а T2 — це значення, яке прискорює або сповільнює швидкість, з якою а зменшується з часом.

$$an +1 = T 1 / [1 + n 2 / (1 + T 2)] \quad (1.2)$$

З використанням правила розміру кроку винагорода, отримана від виконаної дії, буде більшою вагою на початку процесу навчання та зменшуватиметься з часом. У міру того як агент отримує більше досвіду, швидкість навчання зменшується, щоб оновлювати значення Q(s, a) все менше і менше, оскільки воно наближається до оптимальної політики.

Щоб змусити алгоритм досліджувати менш сприятливі стани для пошуку потенційно кращих дій (попередньо зазначена проблема дослідження проти експлуатації), можна використати так званий електронний жадібний алгоритм, який вибирає дію, яка раніше принесла найкращу винагороду більшу частину часу. а інші дії з рівною ймовірністю [18]. На початку швидкість дослідження висока, а в міру набуття агентом досвіду швидкість знижується. Інший підхід полягає у виборі дії на основі градуйованої функції їх оцінного значення, так званий soft-max вибір дії [18]. Зазвичай це робиться за допомогою розподілу Гіббса/Больцмана. Дія a вибирається за допомогою формули, яка обчислює ймовірність вибору дії p(a), яка визначається як:

$$p(a) = (e\ Q[,] / r\ )/(Z\ a\ e\ Q[s,a]/r\ ) \quad (1.3)$$

Де t керує випадковим вибором дії. Коли t є високим, алгоритм частіше досліджуватиме менш оптимальні дії, а зі зменшенням t агент частіше вибиратиме оптимальну на даний момент дію та сходиться до жадібної функції. Порівняно з e-greedy підходом, який однаково зважує різні неоптимальні дії, soft-max зважує їх відповідно до їх значення Q(s, a).

Штучна нейронна мережа є одним з основних інструментів в машинному навчанні. Нейронні мережі складаються з вхідного та вихідного шарів, а також прихованого шару, що складається з блоків, що використовують перетворення вхідних даних. Вони вирізняються відмінними інструментами для пошуку шаблонів, що надто важкі або чисельні для людини, аби обробити ці дані. Штучна нейронна мережа являє собою систему з’єднаних простих процесів (штучних нейронів), що взаємодіють між собою. Схему штучної нейронної мережі представлено на рисунку 2.1.

Існує декілька типів штучних нейронних мереж, кожен з яких використовується в певному випадку. Вони мають різні рівні складності. Основним типом штучної нейронної мережі є штучна нейронна мережа

прямого розповсюдження, де інформація від входу до виходу переміщується тільки в одному напрямку.

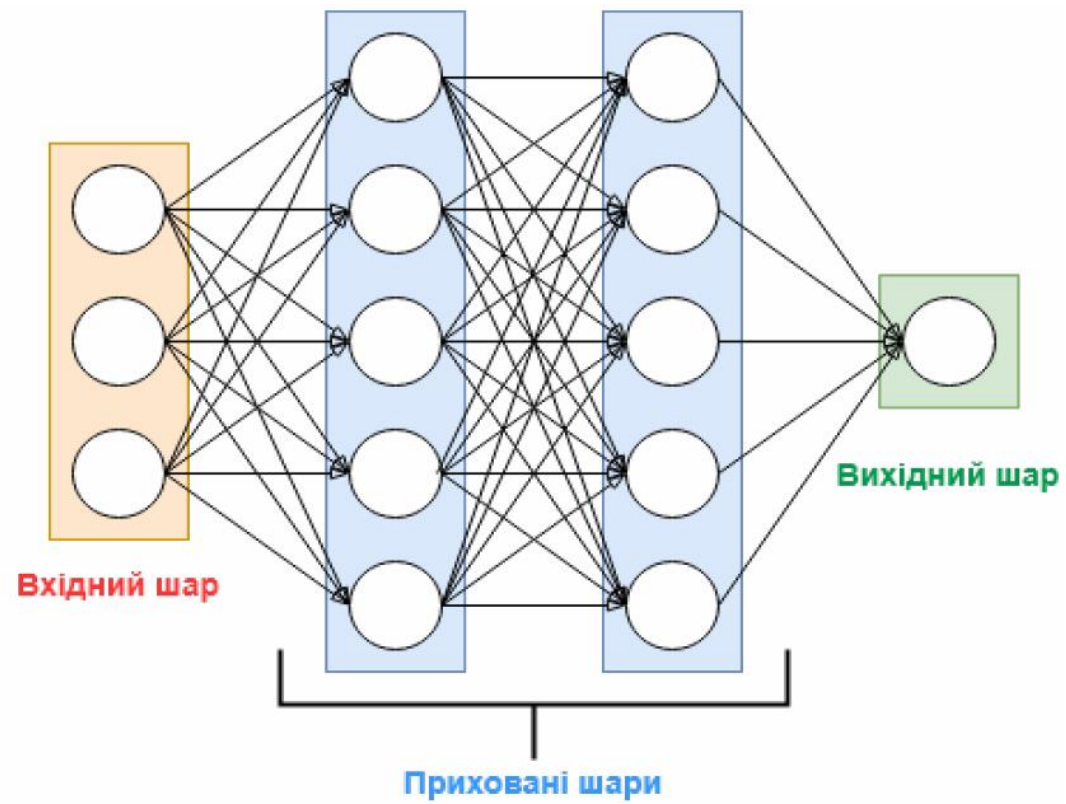

Рис. 1.2. Схема штучної нейронної мережі

Найбільш використовуваним типом мережі є рекурентна штучна нейронна мережа. В ній данні можуть передаватися в декількох напрямках. Дані штучні нейронні мережі мають добрі здібності до навчання, тому широко застосовуються для вирішення задач типу розпізнавання мови або вивчення рукописного вводу.

Вибір штучної нейронної мережі залежить від даних, на яких вона буде навчатися, поставленої задачі та багатьох інших факторів. В деяких випадках виникає необхідність використання декількох підходів до вирішення задачі, наприклад, розпізнавання голосу.

Штучні нейронні мережі, в цілому, використовуються для визначення закономірностей у даних. Задачі, які вирішує штучна нейронна мережа,

можуть включати класифікацію, кластеризацію, прогнозування, апроксимацію, аналіз даних, оптимізацію, стиснення даних та асоціативну пам’ять. Класифікація наборів даних застосовується до попередньо визначених класів, кластеризація ж виконується за різними невизначеними категоріям, а прогнозування передбачає вгадування майбутніх подій на основі попередніх.

На технічному рівні однією з найбільших проблем штучних нейронних мереж є кількість часу, що необхідна для навчання штучної нейронної мережі, оскільки при розв’язанні більш складних задач вона може потребувати значного об’єму обчислювальних потужностей. Проте найбільшою проблемою є те, що штучні нейронні мережі представляють собою «чорні скрині», в які користувач вводить свої дані та отримує відповідь. Штучні нейронні мережі дають змогу точно налаштувати відповіді, проте вони не дають доступу до точного процесу прийняття рішень [23].

Моделювання за допомогою штучних нейронних мереж полягає в налаштуванні вхідного, вихідного, прихованих шарів, кількості нейронів та їх властивостей, що при проходженні певної кількість епох (кількість разів, що дані проходять крізь модель, для навчання) забезпечують необхідну точність.

За структурою зв’язків штучні нейронні мережі можуть бути згруповані в два основних класи:

– нейронні мережі прямого розповсюдження;

– рекурентні нейронні мережі.

В першому класі частіше використовуються багатошарові нейронні мережі, де штучні нейрони розташовані шарами, зв'язок між якими односпрямований і вихідні дані одного шару є вхідними для наступного. Такі мережі відносяться до статичних моделей, оскільки не мають ані зворотного зв’язку. ані динамічних елементів, а вихід не залежить від попередніх станів мережі [24]. Класифікація штучних нейронних мереж наведена на рис. 1.3.

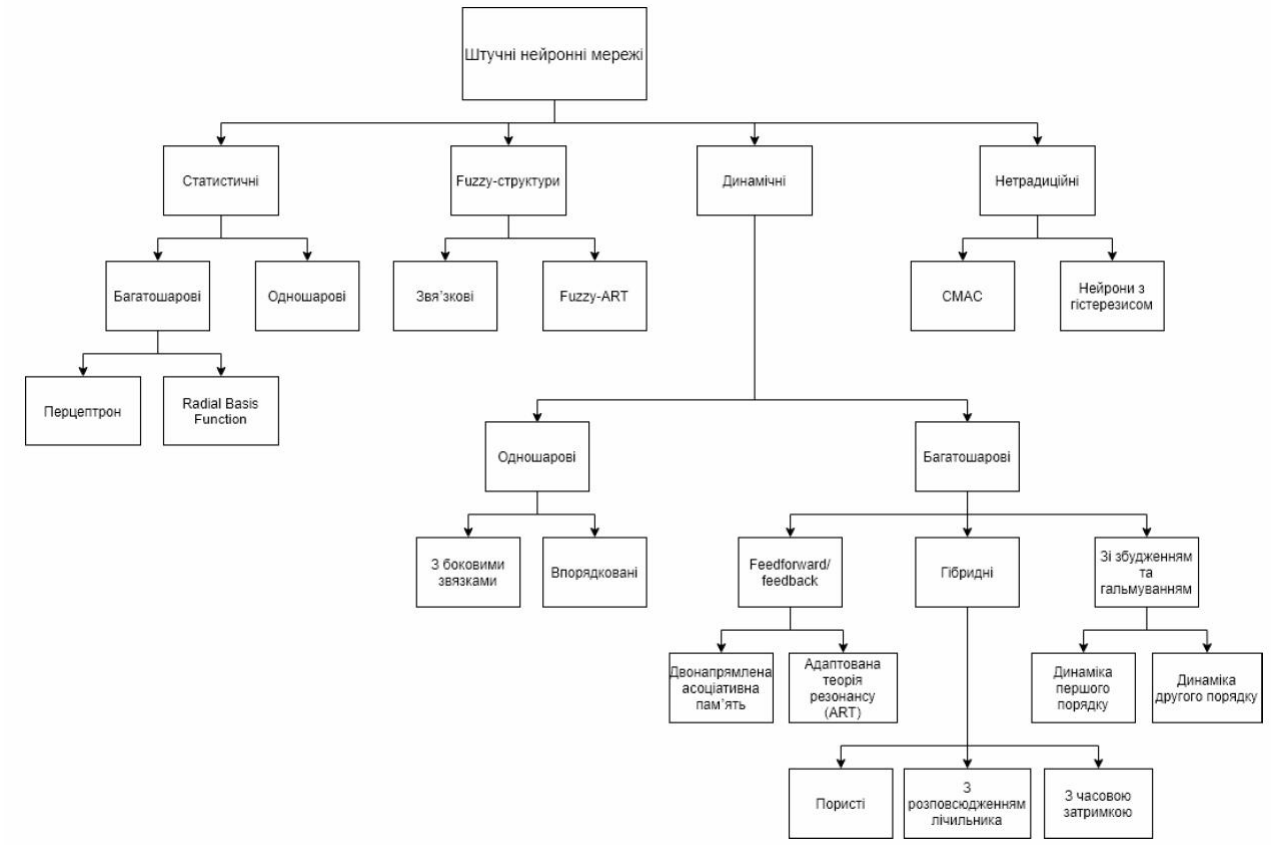

Рис. 1.3. Класифікація штучних нейронних мереж

Навчання в нейронній мережі пов'язане з тим, як люди навчаються у своєму житті, тобто також базується на біологічних процесах. Штучні нейронні мережі можуть навчатися "із вчителем" для опису процесів, які перетворюють вхідні дані на вихідні, тобто для знаходження передатної функції. Навчання "з вчителем" передбачає, що для кожного вхідного вектору існує цільовий вектор, що являє собою необхідний вхід. Разом вони називаються "навчальною парою". Зазвичай, ШНМ навчається на деякій кількості таких начальних пар. Схему навчання "з вчителем" представлено на рисунку 1.4.

Проте навчання "без вчителя" є більш правдоподібною моделлю навчання в біологічній системі. Даний метод навчання не потребує цільового вектору для виходів, а, отже, не передбачає порівняння з ідеальними значеннями. Навчальна множина складається лише з вхідних векторів. Навчальний алгоритм налаштовує ваги так, аби було отримано вихідні

вектори, тобто представлення досить близьких вхідних векторів давало однакові виходи. Процес навчання виділяє статистичні властивості навчальної множини та групує подібні вектори в класи.

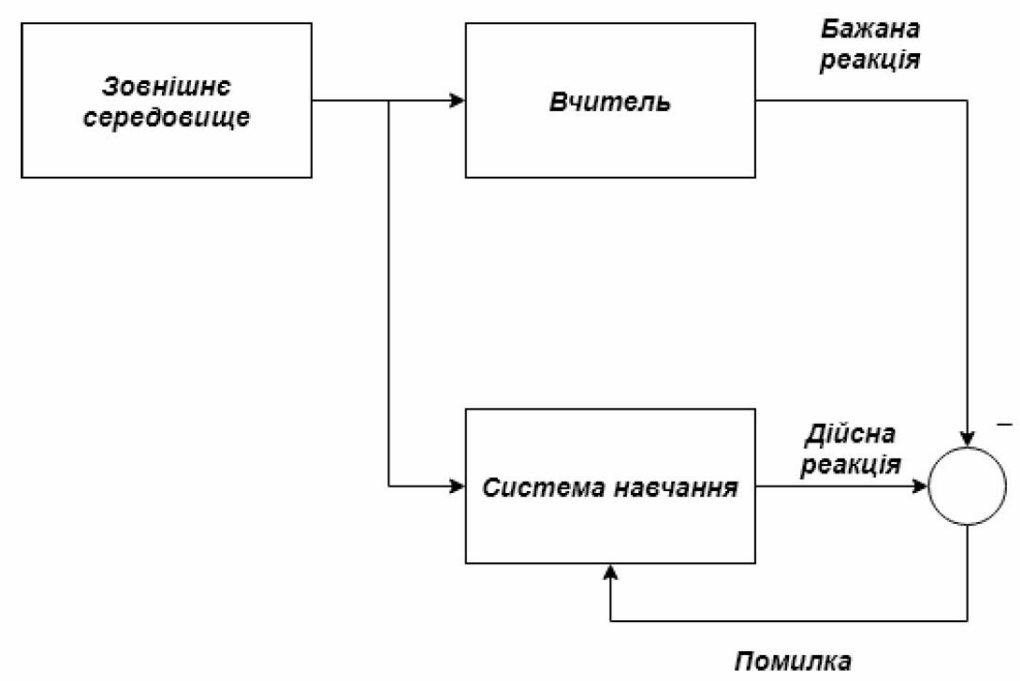

Рис. 1.4. Схема навчання “з вчителем”

Подання на вхід вектору з даного класу дасть певний вихідний вектор, але до навчання неможливо передбачити, який саме вихід буде виконуватися даним класом вхідних векторів. Отже, виходи подібної мережі повинні трансформуватися в певну зрозумілу форму, що обумовлена процесом навчання. Проте це не є серйозною проблемою. Зазвичай, це не складно – визначити зв'язок між вхідними та вихідними даними, що встановила мережа (рис. 1.5).

На основі фактичних та прогнозованих значень вираховується функція втрат (loss function), як різниця між ними. Потім функція втрат надсилається назад у систему.

Нашою метою є мінімізація функції втрат. Отже, доки існує різниця між фактичними та прогнозованими значеннями, є необхідність коригувати ваги,

тобто повертати на вхід функцію втрат – створювати зворотній зв'язок. Цей процес необхідно повторювати, доки дана різниця не стане мінімально можливою. Дана процедура має назву зворотне розповсюдження та застосовується, поки значення loss function не стане мінімальним.

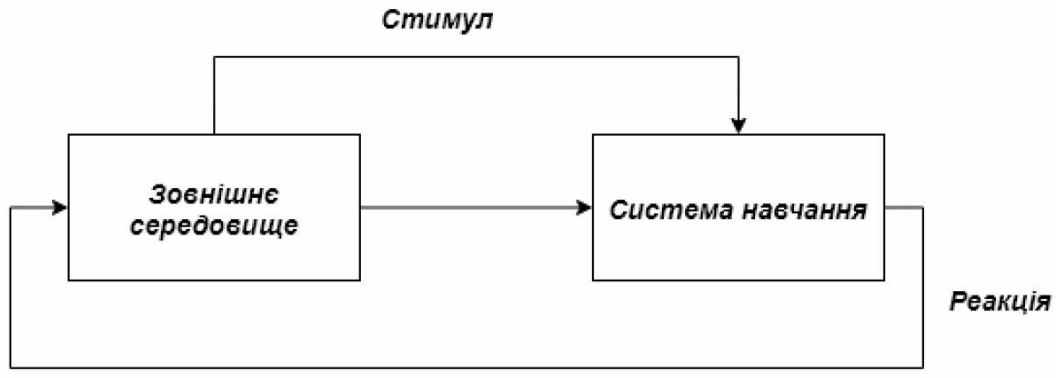

Рис. 1.5. Схема навчання “без вчителя”

Нейронна мережа, що керується даною процедурою, носить назву рекурентна нейронна мережа. Якщо в нейронній мережі дана процедура не використовується, то дану мережу називають нейронною мережею прямого розповсюдження.

Тренування штучної нейронної мережі із застосуванням SGD виконується наступним чином:

1. Довільно задаються ваги в межах від 0 до 1, проте не рівні 0.

2. Вводиться перший рядок даних у вхідний шар, кожний параметр в один вузол.

3. Пряме розповсюдження (Forward-Propagation) виконується зліва направо нейрони активуються таким чином, аби вплив активації кожного нейрону було обмежено вагами.

4. Порівняння фактичного результату із прогнозованим вимірюється loss function.

5. Зворотне розповсюдження (Back-Propagation) виконується справа наліво, помилка розповсюджується в зворотному напрямку. Оновлюються

ваги, що найбільше виплинули на отримання похибки. В такому випадку швидкість навчання відповідає за кількість ваг, які необхідно оновити.

6. Повторюються кроки 1-5 і оновлюються ваги після кожного спостереження.

Проходження пунктів 1-6 називають епохою (epoch).

1.4 Рекурентні нейронні мережі

Повторювана нейронна мережа — це тип моделі нейронної мережі, де мережа вузлів з'єднана сама з собою в цикл у часовому порядку, що дозволяє інформації зберігатися протягом тривалого часу. Це робиться для спільного використання параметрів (наприклад, зміщення та ваги) і виведення в цій моделі для отримання більш точного прогнозу.

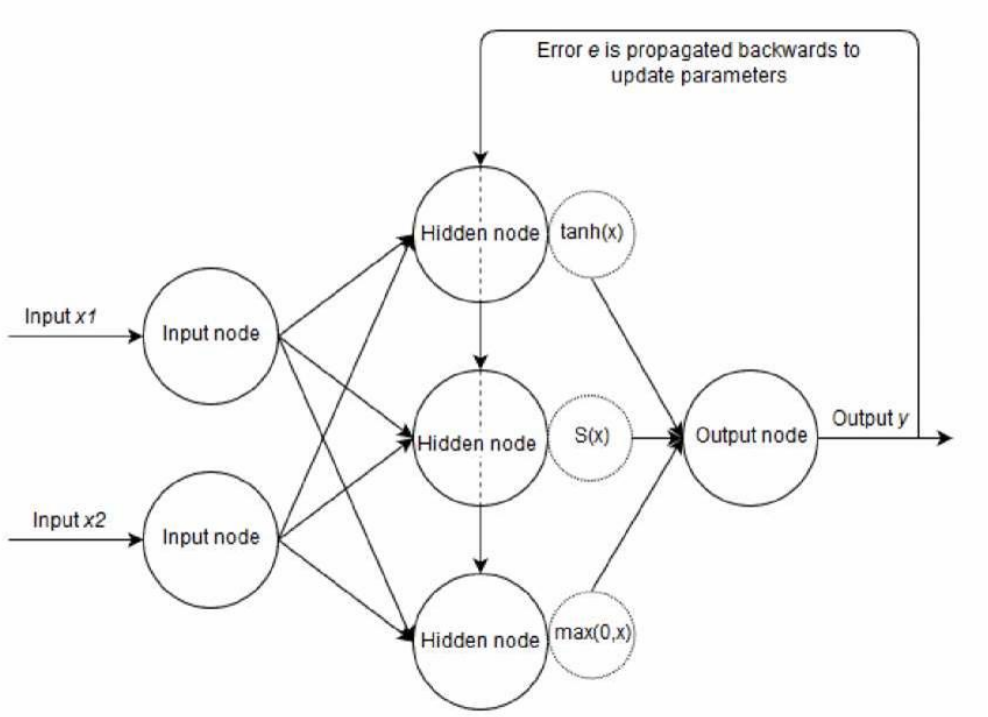

Рис. 1.6 Простий приклад зворотного поширення RNN з функціями активації на прихованих вузлах

Спільне використання параметрів дозволяє використовувати вхідні послідовності (або вектори) нефіксованої довжини [19]. В дослідженні

«Розпізнаванні мовлення з глибокими рекурентними нейронними мережами» [20], що це корисно, наприклад, під час розпізнавання послідовних даних, у яких -важливий контекст, наприклад текст або шаблони мовлення.На рисунку 1.6 показаний приклад простої рекурентної нейронної мережі, де вхідні дані від двох вхідних вузлів поширюються вперед через прихований шар вузлів, де вихідна помилка з попередніх часових кроків поширюється назад у RNN. Вихід із прихованих вузлів шлюзується функціями активації.

На рисунку 1.6 синаптичні ваги були опущені для простоти. Функція активації визначає діапазон значень, які вузол може вивести на основі вхідних даних, наприклад, сигмоїда (s(x), 0 до 1), гіперболічний тангенс (tanh(x), від -1 до 1) або випрямляч (max(0,x)). Це робиться, оскільки вхідними даними для одного вузла може бути сума будь-якої кількості значень. Значення зміщення, додане до цієї суми перед застосуванням функції активації, може вказати, наскільки високою або низькою має бути сума значень, перш ніж відбудеться активація. Вага зв’язку між вузлами визначає, наскільки певний вихідний сигнал від одного вузла впливає на інший вузол. Вихідний вузол повертає передбачення зі значенням від 0 до 1 для представлення ймовірності.

Цінність передбачення або результату — це оцінена ймовірність того, що передбачення є правильним на основі попередньої інформації та передбачень. Потім значення можна порівняти з реальним результатом або з набору даних, або в результаті обчислення результату за допомогою набору правил, що призводить до помилки. У рекурентній нейронній мережі помилки поширюються в мережі назад для оновлення параметрів, як показано на рисунку 1.6. Цей тип архітектури дозволяє RNN базувати нові результати на попередніх результатах за допомогою зворотного поширення.

Зворотне поширення (також називається «зворотним поширенням у часі» або BPTT) — це не те, як нейронна мережа навчається, а те, як вона обчислює градієнт для оновлення параметрів [21]. Фактичне навчання виконується в поєднанні з іншим алгоритмом, таким як стохастичний

градієнтний спуск або будь-який інший алгоритм на основі градієнта загального призначення [21, 22]. Градієнт може використовуватися нейронною мережею не тільки для оновлення параметрів під час навчання, а й для аналізу вивченої моделі або як частина процесу навчання [21]. Проблема під назвою «проблема зникнення/зростання градієнта » присутня в традиційних RNN, що використовують зворотне поширення.

Проблеми градієнта, що зникає/вибухає, виникають через градієнти параметрів, які обчислюються з попередньою довгостроковою інформацією про залежності мультиплікативно з кожним наступним шаром вузлів. У проблемі зникнення це може перешкодити мережі змінювати ваги зв'язків між вузлами достатньою мірою, тобто градієнт зникає. У статті «Про труднощі навчання рекурентних нейронних мереж» [23], що це, у свою чергу, може призвести до уповільнення або зупинки процесу навчання, оскільки великі проміжки між вузлами роблять мережу нездатною з'єднати відповідну інформацію.

В онлайн-курсі «Шпаргалка про рекурентні нейронні мережі» [24], що у випадку вибухового градієнта максимальна зміна градієнта становить часто обмежується алгоритмом «відсікання градієнта», щоб запобігти надто сильному зміненню градієнта за один часовий крок [25]. Велика зміна градієнта може дозволити мережі з'єднати інформацію, яка зазвичай не вважається асоційованою, що робить навчання нестабільним [23, 26].

Використання архітектури «шлюзованої RNN», такої як «довгокороткострокова пам'ять» (LSTM), є ефективним у практичних застосуваннях збору інформації з тривалою тимчасовою залежністю з послідовності даних та її контексту [27]. LSTM вирішує раніше хитро згадану проблему зникнення градієнта [24] за допомогою введення вузлів «забуття» та «пам'яті», а також мультиплікативних блоків стробування, що призводить до того, що LSTM RNN зберігає важливі, залежні від часу дані в циклі та з часом видаляє їх. картки неважливих даних.

1.5 Застосування програмного забезпечення для моделювання технологічних процесів гірничого видобутку

Програмне забезпечення для моделювання, надане AF для використання в цьому дослідженні, керує логічними правилами моделювання. Він забезпечує дотримання цих правил за допомогою інформації щодо плану самої шахти, із зазначеними точками завантаження та скидання, які автономні -машини можуть використовувати як цілі. На рисунку 1.7 показана частина макета та те, як він розділений на сегменти, які визначають напрямок, як можна побачити у підказці у верхньому лівому куті для технологічних процесів гірничого видобутку.

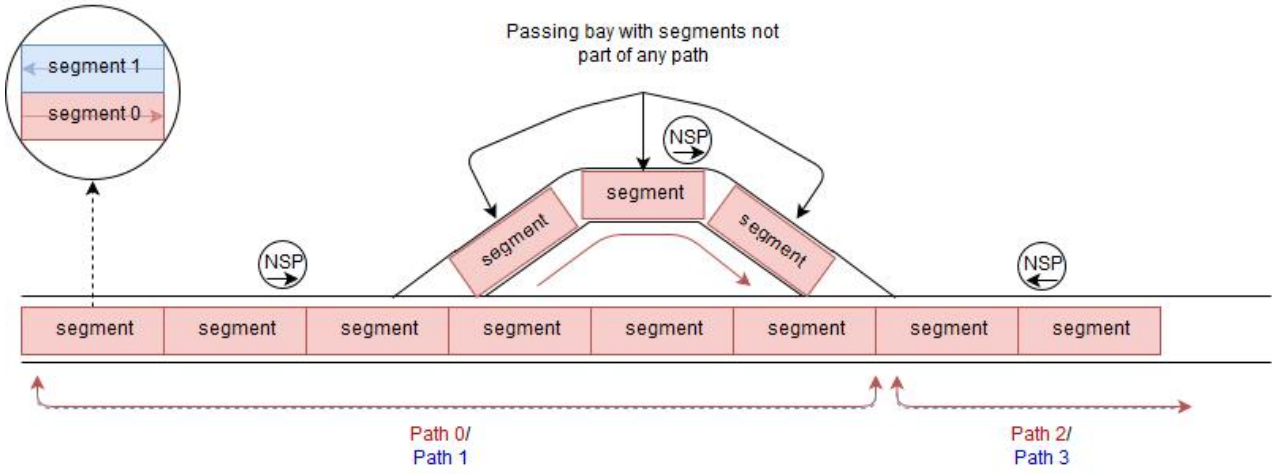

Рис. 1.7 Приклад ілюстрації частини макета з пропускною зоною, трьома точками прийняття рішення, розташованими на трьох різних сегментах, і чотирма шляхами

Кожна частина дороги має один відрізок, який вказує в одному напрямку, а інший — в іншому. Ці сегменти згруповані в так звані шляхи. Сегменти також є частиною менших груп, які називаються «зонами руху» (TZ), які використовуються для виявлення взаємоблокувань. Шляхи можуть містити кілька TZ. Існують винятки з цих узагальнень, наприклад, у проходах, де сегменти не є частинами траєкторії та є сегменти лише в одному напрямку,

як показано червоною стрілкою, що вказує напрямок руху в проході на рисунку 1.7. Машина може займати та перетинати сегмент, лише якщо машина має правильну орієнтацію, а інша машина не може займати той самий сегмент. У точках прийняття рішення, які називаються «точками наступної станції» (NSP), машина запитує систему, чи варто їй продовжувати рух. Ці точки, як правило, є розв’язками та місцями зборів, де шахти трохи ширші, щоб вмістити зупинену машину. У точках макета, де є прохідні відсіки, наказ машині зупинитися натомість спрямовуватиме її до прохідної відсіку, дозволяючи машині пройти. Коли дві машини зустрічаються там, де вони не можуть розминутися, або коли всі машини у виробничій зоні стоять протягом 3 хвилин, виявляється тупикова блокування та повідомляється. У визначених сегментах макета є позначені точки завантаження та скидання. Вони використовуються як точки місії, між якими машини постійно подорожують.

Висновки до розділу

В даному розділі проведено аналіз предметної області застосування машинного навчання в промисловості. Подано цілі та задачі дослідження, розглянуто поняття штучний інтелект і машинне навчання, виконано дослідження машинного навчання з підкріпленням. Проведено огляд застосування програмного забезпечення для моделювання технологічних процесів гірничого видобутку.

РОЗДІЛ 2
ПОБУДОВА МЕТОДОЛОГІЇ ЗАСТОСУВАННЯ МАШИННОГО НАВЧАННЯ В ТЕХНОЛОГІЧНИХ ПРОЦЕСАХ

2.1 Переваги застосування моделей машинного навчання

Оскільки алгоритми RL не вимагають великих класифікованих наборів даних, як це роблять ШНМ, вони можуть почати процес навчання без попереднього знання моделі [10]. Це означає, що модуль RL можна використовувати під час розгортання нових мін без необхідності попереднього отримання навчальних даних. Однак кількість часу, який потрібен алгоритмам RL для наближення -до оптимальної політики, перевищує час, потрібний ШНМ для аналізу набору даних. Інша проблема, пов’язана з використанням RL, полягає у створенні моделі, яка має ряд станів, які агент може досліджувати за розумний проміжок часу, а також відповідну структуру винагороди, яка підсилює поведінку, необхідну агенту для досягнення бажаної мети. Основним аргументом на користь вибору RL є те, що попередні дослідження успішно вирішували подібні проблеми правил дорожнього руху та керування за допомогою алгоритму RL [7-9].

Кількість даних, доступна для моделей DNN, ANN, DRF, GLM, GBM, KNN і SVR у попередніх дослідженнях, недоступна в цьому проекті [2, 3]. Це вагома причина відмови від реалізації однієї з цих моделей, оскільки збір навчальних даних займе значну кількість часу, який може вийти за межі часових рамок цієї дипломної роботи. Якби такі дані були доступні, можна було б використовувати «глибоку архітектуру» ШНМ, таку як LSTM RNN, оскільки ці дослідження показали багатообіцяючі результати в їх реалізації [4] [5]. Ці методи в основному застосовуються в дослідженнях, де метою дослідження є прогнозування транспортного потоку, а не контроль самого трафіку. Попередню роботу щодо прогнозування транспортного потоку,

згадану раніше, можна повторити та застосувати до керування дорожнім рухом. Можна стверджувати, що якщо потік трафіку сегментів можна точно передбачити на короткий термін, то легкий логічний модуль, який використовує ці прогнози, також може вирішити проблеми з контролем трафіку. Цей логічний модуль запитував би модуль прогнозування, чи передбачає він рух транспорту між двома точками прийняття рішення, і може надавати машинам команди щодо того, чи повинна машина продовжувати рух, якщо передбачено рух транспорту з іншого напрямку .

2.2 Аргументи для алгоритму Q-навчання та представлення основних параметрів

Типовий вигляд сценарію RL: агент здійснює дії в серидовищі, що інтерпретується в винагороду та представлення стану, що передаються назад агентові.

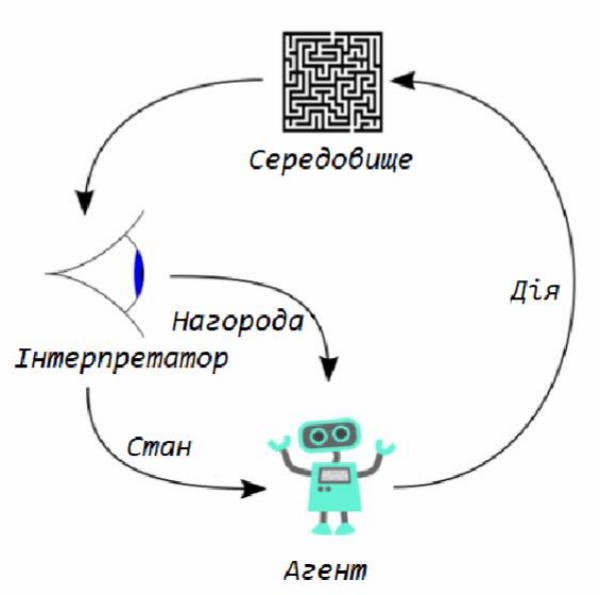

Рис. 2.1. Вигляд сценарію RL

Мета усього навчання для підкріплення - максимізувати суму майбутніх винагород. Існує 2 основних типи алгоритмів RL. Вони засновані на моделях (model-based) та без моделей (model-free).

Для реалізації модуля машинного навчання було обрано алгоритм навчання з підкріпленням Q-learning.

Це було пов’язано з його підтвердженим успіхом у пов’язаній роботі з подібними проблемами, пов’язаними з контролем дорожнього руху, як показано в роботі [8]. Дані, які надає система, і той факт, що під час дисертації відсутні навчальні дані, посилюють аргумент на користь використання навчання з підкріпленням. Вимога щодо можливості налаштування системи для навчання на новому макеті без будь-якої додаткової конфігурації вручну - також є аргументом на користь Q-learning, оскільки вона не потребує жодних попередніх знань про середовище та дії, які необхідно виконати. правильно чи неправильно в а певна ситуація; воно вчиться з часом методом проб і помилок. Реалізація Q-навчання представлена в рівнянні (1.3).

Вибрані стратегії вибору дій

Для регулювання дослідження агента було обрано e-greedy підхід завдяки раніше доведеному успіху [7]. Значення е було встановлено на початковому рівні 0,2 і зменшувалося з часом. Це число було вибрано для того, щоб агент міг досліджувати стани за межами локального оптимального рівня, зберігаючи його відносно низьким, оскільки одна несприятлива випадкова дія може призвести до тупикової ситуації. Розпад контролювався шляхом віднімання i/n*s, де n — загальна кількість запущених епізодів. Це знизить видкість дослідження лінійно на основі поточного номера епізоду, дозволяючи агенту більше використовувати вивчену політику в міру просування навчання.

Альтернативна стратегія вибору дій відповідно до політики була використана під час навчання з сегментами, ефективно перетворюючи метод машинного навчання з Q-навчання на SARSA. Це було пов’язано зі спостереженнями про те, що e-greedy підхід, здавалося, зближувався дуже повільно через збільшення простору станів при переході від шляхів до

сегментів. Після відвідування раніше невідвіданого стану всі дії, пов’язані з ним, отримають однакові значення винагороди. Агент випадковим чином вибере найпершу дію, і середовище поверне винагороду, згенеровану цією дією. Якщо агент отримує позитивну винагороду, ця пара стан-дія матиме вищу винагороду, асоційовану з нею, і її вибір продовжуватиме. Якщо він отримує негативну винагороду, дію не буде обрано, якщо пов’язана з нею винагорода буде меншою за іншу дію. У деяких штатах будь-яка дія призведе до тупикової ситуації або негативної винагороди, це через те, що дія, виконана раніше, не була оптимальною. Це буде поширюватися у зворотному напрямку з часом, доки цей «кінцевий стан» не буде досягнутий через цю неоптимальну дію. Теоретично ця стратегія добре працює під час застосування цього дослідження через послідовний характер відвідувань штатів, де випадковий вибір дій може виявитися шкідливим.

2.3 Інструменти та фреймворки, що використовуються для реалізації методів машинного навчання

З доступних бібліотек і фреймворків для машинного навчання було обрано набір інструментів OpenAI Gym, оскільки він містить інструменти для розробки алгоритму машинного навчання , середовища та інструменти для порівняння різних алгоритмів. Це середовище відповідає певному інтерфейсу, і якщо розроблений алгоритм, агент або середовище мають недоліки або є недостатніми, нові можуть бути розроблені та використані для кращої продуктивності порівняно легко.

Дані, доступні з платформи моделювання

Доступними даними були загальна кількість сегментів у тестовій зоні та стан машин. Машини надіслали інформацію, включаючи стани окремих машин, які мають дані з унікальними ідентифікаторами (UID), інформацію

щодо їхнього положення (X- та Y-координати), які сегменти або шляхи займали машини, а отже, також напрямки їх руху, а також загальна кількість машин.

Машини також надсилали запити до NSP, що містили UID машини, що подала запит, виробничу зону та пункт призначення місії. Після того, як машина пройшла точку прийняття рішення, система надсилала б інформацію про те, що сталося в результаті проходження цієї точки прийняття рішення, лише якщо виникла тупикова блокування. Інформація в цьому результаті - містить UID машини, логічну оцінку відповіді на запит (істинне, якщо сталася взаємоблокування) і повідомлення про причину, яке завжди читається як «тупикова блокування».

2.4 Дослідження проблеми з використанням симуляційної платформи для машинного навчання

Під час навчання модуля ML з використанням платформи моделювання, згаданої у фоновому режимі, стало очевидним, що використання моделювання, яке працює щонайбільше в два рази швидше за «реальний час», було надто повільним, щоб отримати результат протягом відведеного -часу, оскільки приблизно 800 епізодів закінчилися через 12 годин. За цей час агент навчився працювати з одним сценарієм, але не пройшов достатнього навчання, щоб мати можливість працювати без взаємоблокувань протягом 24 годин. Крім того, у міру того, як агент покращується з часом, він досягне станів, що знаходяться далі, і, таким чином, загальний час роботи збільшується. З дедалі складнішими схемами майнінгу та кількістю машин час процесу навчання перевищував би час, який знадобився б для ручного налаштування, і, таким чином, не було б життєздатним рішенням. Потрібна симуляція симулятора (метасимулятора), який міг би працювати швидше, ніж поточна платформа.

Вирішальними факторами, які цей метасимулятор мав точно симулювати, були:

- Відправка запитів точно в ті ж точки
- Ідентичне зображення шахти
- Точне визначення шляху, яке не порушує правил компонування
- Ідентичне виявлення взаємоблокувань
- Рух працює так само

Можливі рішення проблем із симуляцією

Альтернативою платформі моделювання було використання інструменту моделювання, такого як SUMO2 , для створення подібної симуляції. Це можна використовувати для швидшого збору даних про час навчання, а також точність моделі. Проблема з цим підходом полягає в тому, що буде важко точно відтворити симуляцію системи керування дорожнім рухом (TCS) за допомогою фреймворку, і результати, отримані в результаті цього, можуть не точно відображати продуктивність при використанні з реальною системою, або політики, створені в процесі навчання, можуть бути взагалі несумісними з TCS. Ця проблема пов’язана з тим фактом, що використовуючи файли фактичної конфігурації макета, фреймворк повинен бути адаптований для точного моделювання макета шахти через те, що він призначений для імітації звичайного трафіку.

Іншим варіантом було створення спрощеного симулятора симуляції або метасимулятора з використанням файлів конфігурації для макета шахти симулятора, який використовується у виробництві. Це було б досить складним і трудомістким як з точки зору розробки, так і тестування, але врешті-решт це було б найточнішим представленням фактичного симулятора. Це пов'язано з тим, що логічні правила моделювання і зв'язки між, наприклад, сегментами, шляхами та іншими об'єктами інтересу в макеті досить складні. Створення програмного забезпечення, що містить необхідні функціональні можливості та

атрибути, у цьому випадку може бути вигідним у порівнянні з адаптацією системи моделювання міського руху відповідно до спеціалізованих правил для наближення гірничодобувної операції. Через ці передбачувані проблеми з використанням імітаційної структури симулятора для наближення фактичної конфігурації симулятора було обрано розробку метасимулятора з використанням файлів конфігурації.

Метасимулятор

Інформація про шахту в певній виробничій зоні аналізується в метасимуляторі з файлу, характерного для цієї виробничої зони. Цей файл містить інформацію про сегменти, шляхи та зони руху. Інформація про сегмент включає всі сегменти, присутні у виробничій зоні, до яких сегментів вони підключені, а також сегмент, який належить до якого шляху та зони руху. TZ використовуються для виявлення блокування; якщо машина хоче виділити вже зайняту сусідню зону руху, виникла тупикова блокування.

Стан метасимулятора представлено кількома машинними об’єктами з унікальними ідентифікаторами. Ці машини містять початкову та кінцеву точку місії (точка завантаження/скидання), поточну кінцеву точку (таку саму, що й кінцева точка місії, за винятком перенаправлення до проходу), поточний сегмент і поточний ідентифікатор TZ. Значення сегментів і точок використовуються для маршрутизації машин через макет шахти. Ідентифікатор TZ використовується для пошуку об’єкта зони руху, призначеного машиною.

Як показано в лістингу 1.1, для ініціалізації та зміни стану симулятора є три фази, в яких симулятор може проходити, щоб точно імітувати поведінку фактичного симулятора . Під час роботи на кожному кроці часу проходять лише фази «переміщення» та «виявлення» . Якщо виникла взаємоблокування або ініціюється метасимулятор , метасимулятор переходить до «початкової» фази

```
Initial phase
      Setup machines at initial segments with destination:
            Find a path from the current segment to destination
      go to Move phase

Move phase
      For all machines:
            If the last segment in mission path:
                  Find a path to next destination

      Validate next move:
            If can allocate next TZ and If at a decision point:
                  Ask ML module if it is allowed to pass:
                        True -> move to next segment and TZ
                        False -> move to passing bay/stay on segment
                                 (if passing bay exists)
      go to Deadlock Detection phase

Deadlock Detection phase
      Check for deadlock:
            Deadlock detected -> go to initial phase
            No deadlock -> go to move phase
```

Лістинг 2.1. Псевдокод розробленого метасимулятора

Розроблений модуль машинного навчання

У цьому підрозділі буде представлено розроблене визначення епізоду, стан, дії, середовище та структуру винагороди для модуля, а також наведено причини вибору дизайну.

Визначення епізоду

Алгоритм Q-навчання було налаштовано на виконання в епізодах, де кожен епізод триває, доки не буде виявлено тупик або поки не буде досягнуто бажаної кількості точок призначення/виконаних місій. Досягнення пункту призначення означає, що машина пройшла через весь макет до місця призначення без будь-яких машин, що створюють тупикову ситуацію. Загальний час роботи в годинах можна розрахувати за допомогою метасимулятора, оскільки кожна машина зазвичай виконує 10 із цих перемикань пунктів призначення або проходів за годину. Це означає, що за 24 години одна машина виконує 240 проходів. Модуль можна вважати достатньо підготовленим для виконання визначених у цій дисертації цілей

продуктивності, коли він досягає 240 проходів на машину без виникнення тупикових блокувань. Тому цей показник можна використовувати як KPI для визначення продуктивності модуля. Під час запуску та скидання після тупикових блокувань машини розміщувалися на випадкових сегментах на визначеному шляху та з кінцевими точками місії в правильному напрямку руху, щоб запровадити обмежений стохастизм у початкових станах.

Їх не розміщували й не задавали кінцеві точки місії абсолютно випадково через те, як сценарій тестування представлено у фактичному TCS, який знаходиться на заданих сегментах і встановлених кінцевих точках місії.

Визначення стану

Було розроблено два різних способи визначення держави. Стан навколишнього середовища було представлено як n-вимірний вектор, де n – загальна кількість шляхів або сегментів у зоні видобутку. Разом із шляхом або сегментом, який займає машина запиту, вони утворюють стан, який не залежить від ідентичності машин, залежить лише від того, де вони розташовані. Довжина кожного виміру вектора була двійковою 0 або 1 залежно від того, чи був шлях або сегмент зайнятий іншою машиною чи ні. Приклад стану на основі рисунка 1.3 виглядатиме так: ([i,i,o,o],'шлях 3'). У цьому випадку перший і другий шляхи були зайняті машинами, відмінними від машини запиту, а машина , яка надіслала запит, була на шляху «шлях 3». Використання шляхів замість сегментів призводить до значно меншого простору стану порівняно з використанням загальної кількості сегментів зони. Однак це також означало, що для того, щоб модель працювала, потрібна невелика кількість попереднього визначення зони. Використання однієї зі згаданих функцій оцінки обійшло б це, але попередньо визначені шляхи дозволили б агенту набагато легше вивчити оптимальні дії через менший простір станів. Проте використання сегментів створило простір станів arger порівняно з використанням шляхів, однак дає більш точне представлення

стану симуляції. Обидва державні представлення були протестовані та порівняні.

Акції та нагороди

Дії, як згадувалося раніше, були визначені як логічні значення «іти» (істина) і «стоп» (хибно). Винагороди були структуровані відповідно до того, скільки часу алгоритм працює без створення тупикової ситуації. З цієї причини кожного разу, коли агент обробляв запит на передачу, агент отримував невелику винагороду за команду переходу, жодної винагороди за команду зупинки, велику винагороду, якщо машина досягла точки завантаження/скидання, і великий штраф за кожні час, коли система виявила взаємоблокування.

Цю загальну структуру винагороди було обрано таким чином, щоб агент визначав пріоритетність дій, які призводять до станів, у яких машини досягли пункту призначення. Таким чином, це повинно стимулювати дії, які не тільки запобігають тупиковим ситуаціям, але й забезпечують рух машин від точки до точки, з упередженням у бік руху, а не зупинки.

2.5 Опис середовища та взаємодія агентів. Представлення архітектури модуля

Середовище, як згадувалося раніше, це те, за чим агент спостерігає та з чим взаємодіє, щоб отримати винагороду. Сам агент бачить лише стан середовища та знає, які дії раніше призвели до певного результату, і вводить дію (випадковим чином або на основі того, що раніше працювало найкраще) у середовище, щоб отримати винагороду для додавання до пройденого стану в Q-таблиці.

У самому середовищі, однак, потрібно виконати два додаткових завдання всередині цієї «крокової» функції. По-перше, виконується дія, надана

агентом середовищу, і надсилається вибрана дія на платформу моделювання для виконання та чекає проходження наступного запиту, щоб оновити стан середовища. Якщо та сама машина запитує дію, і стан такий самий, як останній стан, вона повертає останню відповідь і утримується від повернення нового стану та винагороди агенту, оскільки оновлення для цієї пари стан-дія вже виконано, і тому дискретний часовий крок ще не зроблено. По-друге, винагорода генерується залежно від того, до чого призвела дія агента або яка це була дія. Нарешті, це не завдання саме по собі, але все ж важливий крок: новий стан, отримана винагорода та логічне значення, яке вказує, чи закінчився епізод, повертаються агенту.

Архітектура модуля

Модуль машинного навчання складається із середовища з агентом для взаємодії з ним. Оскільки середовище має чекати оновлень, наданих TCS, між TCS і середовищем було розміщено координатор. Спільний об'єкт між середовищем і координатором служить способом зв'язку. Коли машина досягає точки прийняття рішення, TCS надсилає запит «координатору», який, у свою чергу, ініціює агента виконати дію. Зазвичай середовище OpenAI/Gym має поточний стан і всі правила для його оновлення, але розроблене середовище залежить від форми моделювання та її правил. Це було основною причиною використання координатора. Коли агент виконує дію, вона пересилається через координатора до TCS, щоб наказати машині або йти, або зупинитися.

Як показано на рисунку 2.2, під час запуску головна програма запускає агент і надсилає посилання на спільний об’єкт, через який середовище й координатор можуть спілкуватися.

Агент, у свою чергу, запускає середовище та починає виконувати дії. Щоразу, коли координатор отримує оновлення позиції від однієї з машин, він оновлює стан спільного об’єкта. Коли координатор отримує запит перевірити,

чи дозволено машині пройти чи ні, він додає подію в чергу спільного об’єкта, яка спонукає середовище повернути наступний стан агенту разом із відповідними винагородами.

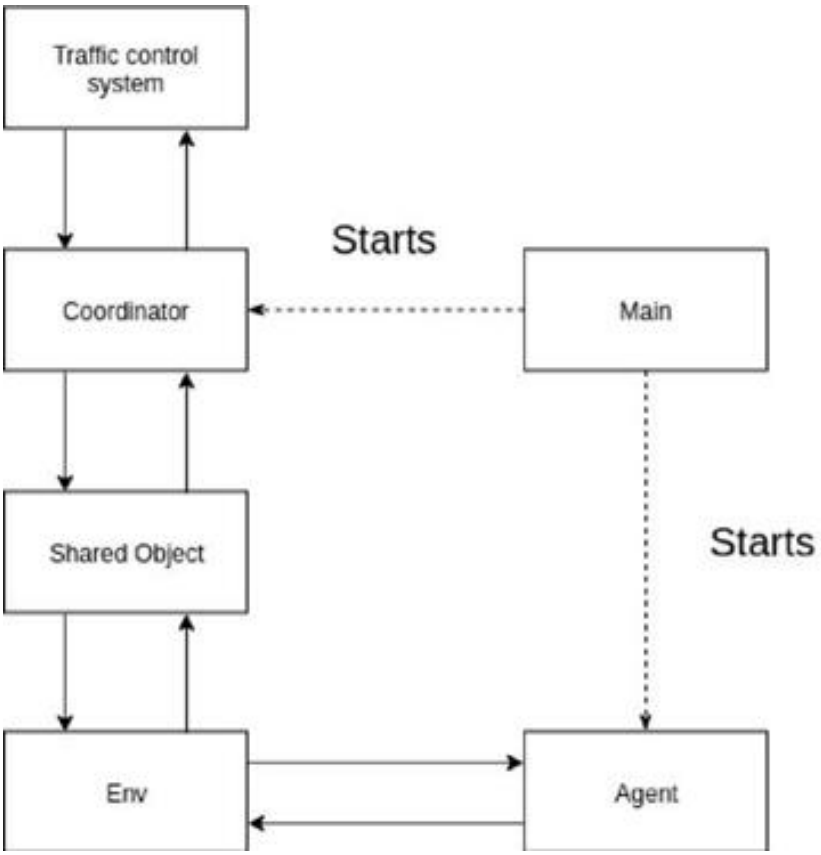

Рис. 2.2. Діаграма, на якій показано архітектуру, яка використовується для підключення середовища до зовнішньої платформи моделювання TCS

Агент виконує наступну дію на основі нового стану, який середовище записує в спільний об’єкт, і додає відповідь у чергу відповідей, щоб повідомити координатора про те, що дію було вибрано . Потім координатор повертає цю дію системі керування дорожнім рухом.

Агент чекає своєї винагороди та наступного стану до наступного разу, коли машина дійде до точки прийняття рішення, де система запитує наступну дію. Якщо виникне взаємоблокування , система повідомить про це агенту з наступним станом і винагородить повідомленням про завершення епізоду.

Потім положення машин скидаються, і починається новий епізод, коли машина досягає точки прийняття рішення.

Під час використання мета-симулятора під час навчання пакет, що містить координатор і систему контролю руху, було замінено на мета-симулятор, як показано на рисунку 2.3.

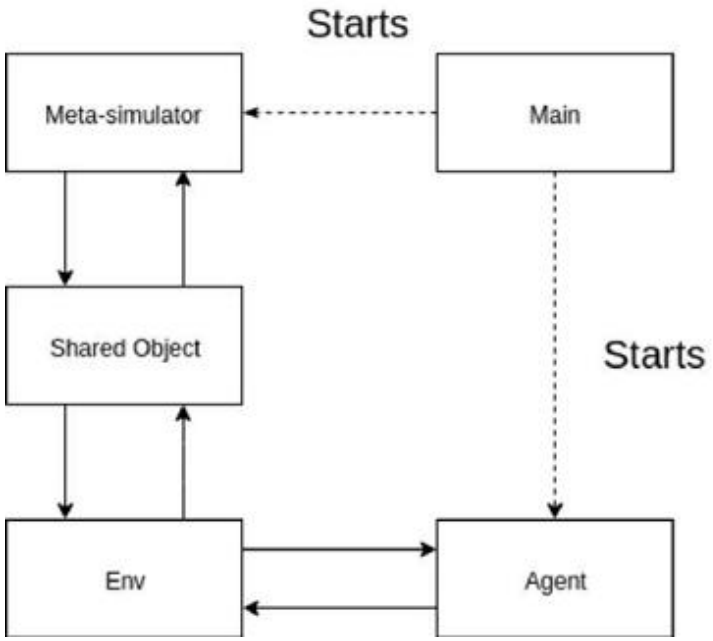

Рис. 2.3 Архітектура модуля машинного навчання під час руху до метасимулятора, зауважте, що координатор і TCS замінено розробленою метасимулятором

Основна програма, яка налаштовує симулятор і модуль ML, може запускати кілька агентів, використовуючи одну Q-таблицю разом із кількома метасимуляторами. Це буде прискорити швидкість, з якою стани зустрічаються та вивчаються. Обидва агенти використовуватимуть цю спільну Q-таблицю для читання та запису результатів. У міру того, як швидкість дослідження зменшується , подальша експлуатація в процесі навчання буде базуватися на тому, що спостерігали обидва агенти. Таким чином, обидва чоловіки повинні отримувати однакові винагороди, не враховуючи винагороди від випадкових дій. Кількість агентів, що працюють паралельно, була

обмежена доступним апаратним забезпеченням і могла бути збільшена, якщо було доступно більше ядер процесора. Для цього проекту апаратне забезпечення могло підтримувати лише два екземпляри одночасно до досягнення максимального використання ЦП, але в програмному забезпеченні немає верхньої межі щодо кількості екземплярів, які можуть працювати одночасно.

2.6 Методологія тестування пропонованого рішення для технологічного процесу

Під час скидання машини розміщуються напіввипадково у визначених частинах макета, які можна побачити на рисунку 2.4. Вони були обрані, оскільки це позиції, які призводять до логічно складних ситуацій.

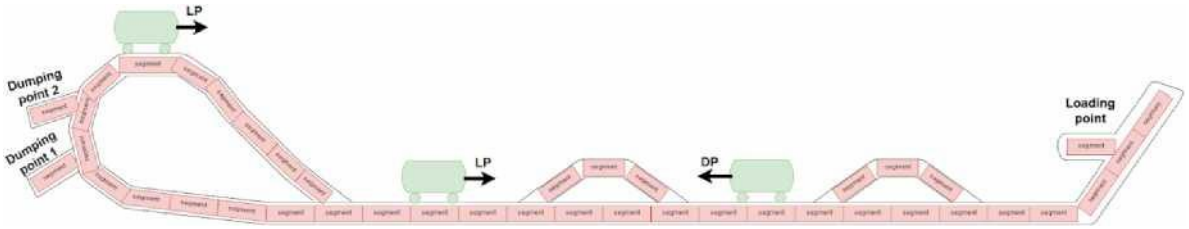

Рис. 2.4. Спрощене представлення макета, який використовувався під час навчання та тестування

На рисунку 2.4 зелені машини можна побачити в місцях, де вони могли б бути розташовані під час скидання, зі стрілками та текстом, що вказує їхній напрямок руху та цілі місії.

Одна машина розміщується у випадковому сегменті циклу, розташованого ліворуч від макета. Точкою його місії було встановлено точку завантаження, яку видно в крайній правій частині макета. Дві машини були розміщені сегментами (не частиною прохідного відсіку) на довгій прямій посередині макета. Одна з машин встановлена в лівій частині довгої прямої з

метою досягнення точки завантаження, тоді як одна була встановлена в крайньому правому сегменті з метою досягнення однієї з точок скидання, які видно на петлі.

У прогонах, де шляхи використовувалися для представлення стану, будь-який сегмент, який не є частиною жодного шляху, розглядався як шлях. Тренування з використанням сегментів представляли весь стан із положеннями машин на сегментах. Дані щодо машин надавалися самою симуляційною платформою кожного разу, коли оновлювалася позиція машини. У сценарії моделювання, на якому тестувався цей модуль, були присутні три машин.

Висновки до розділу

В даному розділі виконана побудова методології застосування машинного навчання в технологічних процесах. Розглянуто аргументи для алгоритму Q-навчання та представлення основних параметрів, інструменти та фреймворки, що використовуються для реалізації методів машинного навчання. Проведено дослідження проблеми з використанням симуляційної платформи для машинного навчання.

РОЗДІЛ 3
ПРЕДСТАВЛЕННЯ ОТРИМАНИХ РЕЗУЛЬТАТІВ ВИКОРИСТАННЯ МЕТОДІВ МАШИННОГО НАВЧАННЯ ДЛЯ КОНТРОЛЮ ТЕХНОЛОГІЧНИХ ПРОЦЕСІВ

3.1 Дослідження концепції цифрового виробництва в контексті контролю технологічних процесів

Цифрове виробництво, засноване на широкому застосуванні комп'ютерної техніки в технологічних процесах виробництва продукції, дозволить вирішити це завдання. Внаслідок чого розробка нових методів для оптимізації процесів цифрового виробництва є актуальним завданням.

Цифрове виробництво – це концепція технологічної підготовки виробництва в єдиній віртуальному середовищі з допомогою інструментів планування, перевірки і моделювання виробничих процесів.

Ключовою складовою концепції цифрового виробництва є використання певного програмного забезпечення, що дозволяє технологам здійснювати свою діяльність більш ефективно.

Причому в більшості випадків мова йде не про те, що технолог виконує звичну йому роботу новим способом, а про зовсім нові, більш ефективні бізнес-процеси (наприклад, той ж технолог створює тривимірні інтерактивні інструкції і передає їх в цех по локальній мережі, що позбавляє його від необхідності створювати операційну карту в звичному розумінні), які здійснюються за допомогою передового програмного забезпечення і дозволяють підприємству отримувати реальну вигоду.

В контексті української промисловості завдання цифрового виробництва полягає не в кардинальній зміні укладу, побудові «розумних» фабрик – це дорого і довго. В першу чергу українська промисловість – це індустрія з територіально розподіленою і неоднорідною інфраструктурою.

Верстатний парк багатьох підприємств складається з обладнання не тільки різних виробників, але і поколінь. Підвищення продуктивності і ефективність того, що є, тобто є задачею перетворити виробництво в керований комп'ютером в реальному часі процес і доповнити його штучним інтелектом. Виробничі підприємства мають потенціал у багатьох напрямках – від віртуальних асистентів до робототехніки, використовуючи швидкозростаючі обсяги даних для оптимізації основних процесів в режимі реального часу, автоматизації виробничих ліній, зниження кількості помилок і часу простою, зменшення витрат сировини, часу доставки, або, нарешті, підвищення якості продукції.

Зараз поле побудови цифрових формул і іншого цифрового виробництва дуже широко завдяки можливостям збільшення обсягів виробництва на поточних потужностях, зниження витрат і собівартості.

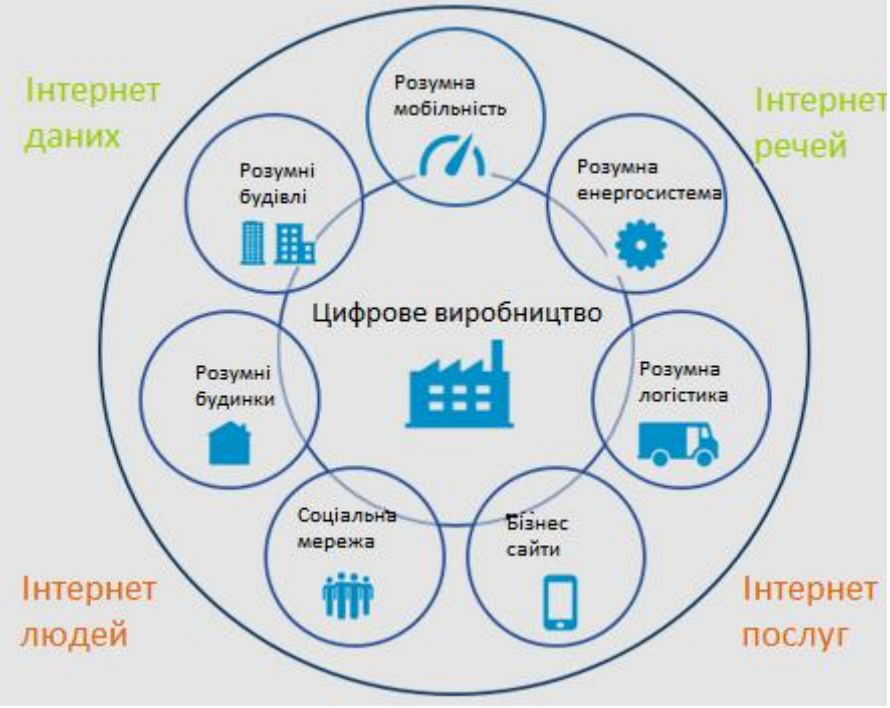

Рис. 3.1. Структура цифрового виробництва

Тому кожне підприємство намагається використовує цей потенціал по різному.

Отримані таким чином результати не до кінця виправдовує очікування. Нестандартне рішення найчастіше неможливо масштабувати, потрібно багато ресурсів на його підтримку. На все це йде багато часу і грошей, а готового стандартного рішення в підсумку немає.

Для того, щоб підприємство стало цифровим, воно повинно підключити до промислового інтернету все, що можна, почати моніторинг і отримувати достовірні дані. В світі, 90% обладнання не підключено ні до яких мереж, з цього обладнання не фіксуються поточні дані.

Люди дуже часто плутають процес автоматизації і процес впровадження цифрового виробництва (рис. 3.1).

В першу чергу необхідно навчитися збирати дані будь-які процесів і з будь-якого обладнання в промисловості. Це дозволить фіксувати роботу і це найпростіше і ефективне рішення.

3.2 Методи оптимізації технологічних процесів цифрового виробництва

Існує безліч способів застосування кластерного аналізу. Найчастіше він виступає як інструмент, що дозволяє поглянути на дані в цілому. Також кластерний аналіз може використовуватись для попередньої обробки або як проміжний етап інших алгоритмів, таких як класифікації або прогнозування.

В задачах за допомогою кластерного аналізу створюється комплексне зведення даних для класифікації, відбувається виявлення шаблонів, формування і перевірка гіпотез.

Кластеризація - це автоматичне розбиття елементів (об'єкти, дані, вектора характеристик) на групи (кластери) за принципом схожості [1]. На рис. 3.2. представлена класифікаційна схема алгоритмів кластеризації, яка може бути використана для класифікації і групування деталей на класи, види, групи або типи.

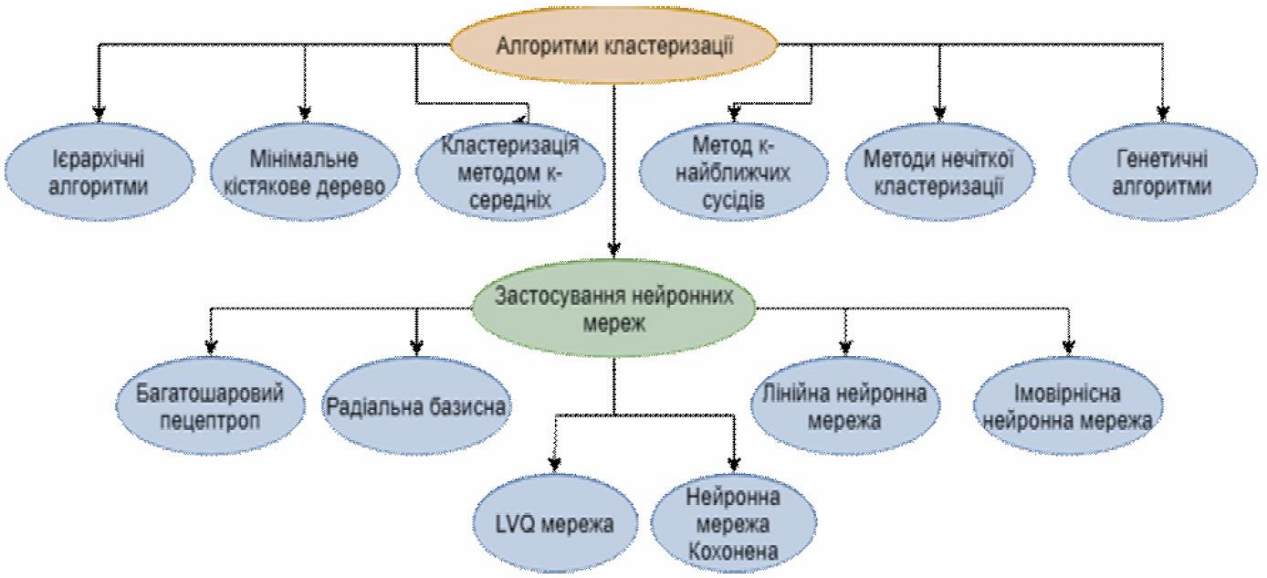

Рис. 3.2. Класифікація алгоритмів кластеризації

Методи оптимізації проектних технологічних процесів цифрового виробництва за допомогою нейронних мереж.

Нейронні мережі використовуються для вирішення складних завдань, які потребують аналітичних обчислень подібних тим, що робить людський мозок. Найпоширенішими застосуванням нейронних мереж є:

– класифікація – розподіл даних по параметрах;

– попередження – можливість передбачати наступний крок;

– розпізнавання – в даний час, саме широке застосування нейронних мереж.

Вони діляться на три основних типи: вхідний, прихований і вихідний. У тому випадку, коли нейромережа складається з великої кількості нейронів, вводять термін шару. Відповідно, є вхідний шар, який отримує інформацію, n прихованих шарів, які її обробляють і вихідний шар, який виводить результат.

У кожного з нейронів є 2 основних параметра. Вхідні дані (input data) і вихідні дані (output data). У випадку вхідного нейрона: input=output. В інших, у полі input потрапляє сумарна інформація всіх нейронів з попереднього шару, після чого, вона нормалізується, за допомогою функції активації (поки що просто уявімо її f(x)) і потрапляє в поле output [2].

На рисунку 3.3 зображені основні параметри нейрона.

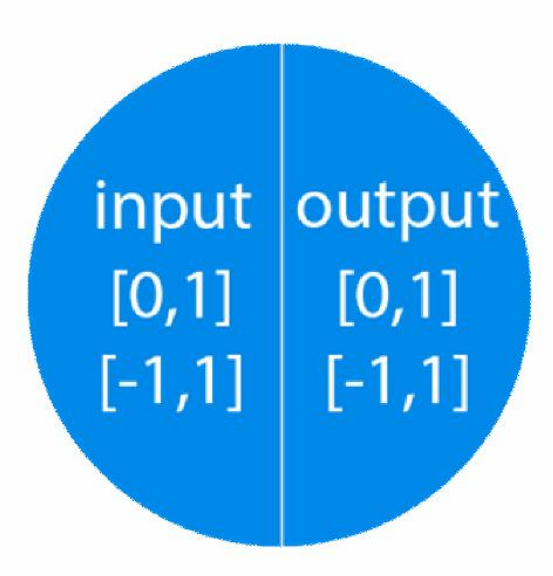

Рис. 3.3. Основні параметри нейрона

3.3 Методи машинного навчання у цифровому виробництві

В сучасному світі постають важливі питання щодо методів навчання. З розвитком технологій термін «навчання» постійно розширюється, знаходячи застосування не лише серед навчання людей, а й серед навчання машин.

Машини, обладнані штучним інтелектом, можуть замінити людину у виконанні небезпечних та складних завдань, де необхідна велика сила, швидка увага, або обробка великої кількості схожої інформації.

Розвиток інформаційних технологій має на меті створити засоби, що будуть полегшувати життя: удосконалювати виробництво тощо.

Дерево прийняття рішень - засіб підтримки прийняття рішень, який використовує граф або модель прийняття рішень, а також можливі наслідки їх роботи, включаючи вірогідність настання ситуації, витрати ресурсів та корисності [3].

Важливо пам'ятати, що нейрони оперують числами, які знаходяться в діапазоні [0,1] або [-1,1]. На рисунку 3.4 зображена структура дерева.

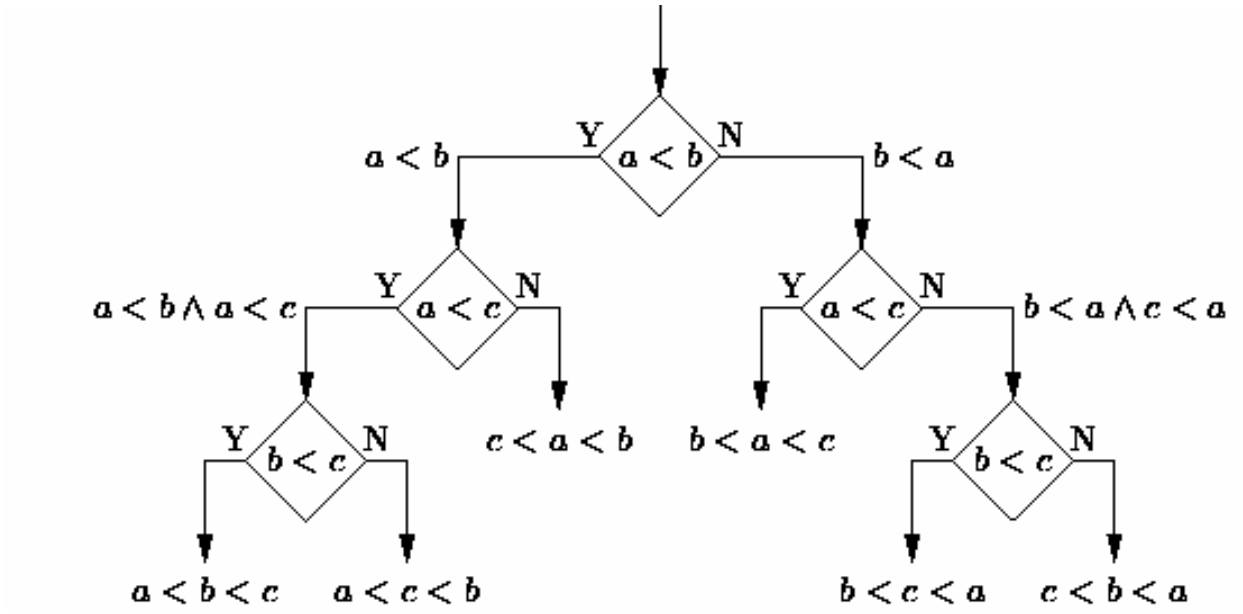

Рис. 3.4. Структура дерева прийняття рішень

З точки зору цифрового виробництва, дерево класифікації є мінімальною кількістю максимальних і мінімальних значень з датчиків. Якщо розглядати дерево як метод, то воно дозволяє підійти до вирішення проблеми зі структурованою і систематичної боку, щоб в результаті прийти до логічного висновку за даними датчиків.

Метод найменших квадратів.

Даний метод виступає в ролі методу для реалізації лінійної регресії. Найчастіше вона представляється у вигляді завдання прямої лінії, що проходить через безліч точок. Є кілька варіантів її здійснення, і метод найменших квадратів - один з них. Можна намалювати лінію, а потім виміряти відстань по вертикалі від кожної точки до лінії і «перенести» цю суму вгору.

Необхідної лінією буде та конструкція, де сума відстаней буде мінімальною. Іншими словами, крива проводиться через точки, що мають нормально розподілене відхилення від істинного значення. Приклад графику найменших квадратів зображено на рисунку 3.5.

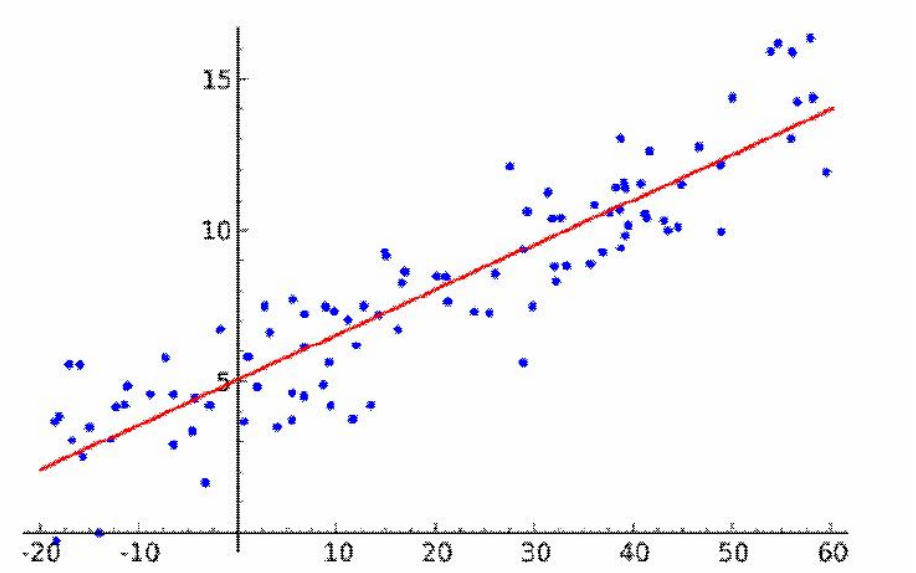

Рис. 3.5. Приклад графіку методу найменших квадратів

Логістична регресія є потужний статистичний метод прогнозування ймовірності виникнення деякої події з однією або декількома незалежними змінними. Основною метою логіт-регресії є виявлення зв'язку між декількома незалежними змінними і залежною змінною. В аналізі даних принцип логістичної регресії дозволяє не тільки класифікувати спостережувані події, а також моделювати зв'язок між ними.

Розробка логістично-регресійної моделі Дж. Олсоном [15] висвітлювала важливість факторів для прогнозування ймовірності. Logit-моделі засновані на прагненні більш досконалого апарату попередження виникнення кризи у суб’єктів господарювання та базуються на побудові кумулятивних функцій, при цьому не вимагають, щоб незалежні змінні дотримувались багатофактор ного нормального розподілу, а також було виключено обмеженість до лінійних рівнянь (характерних для MDM моделей). Таку модель найчастіше відносять до моделей бінарного типу, причиною цьому є використання характеристик типу 0 та 1, що відповідно пояснюються як значення ймовірності виникнення банкрутства на підприємстві або його відсутність.

Логістична регресія визначає ступінь залежності між категоріальної залежної й однієї або декількома незалежними змінними шляхом

використання логістичної функції, що є акумулятивним логістичним розподілом [4].

Даний алгоритм активно використовується в цифровому виробництві, а саме при:

– оцінці водостійкості обладнання;

– вимірі показників успішності виробництва;

– прогнозі доходів з певного продукту;

– обчисленні можливості виникнення аварійної ситуації на виробництві.

Приклад графіку методу логічної регресії зображено на рисунку 3.6.

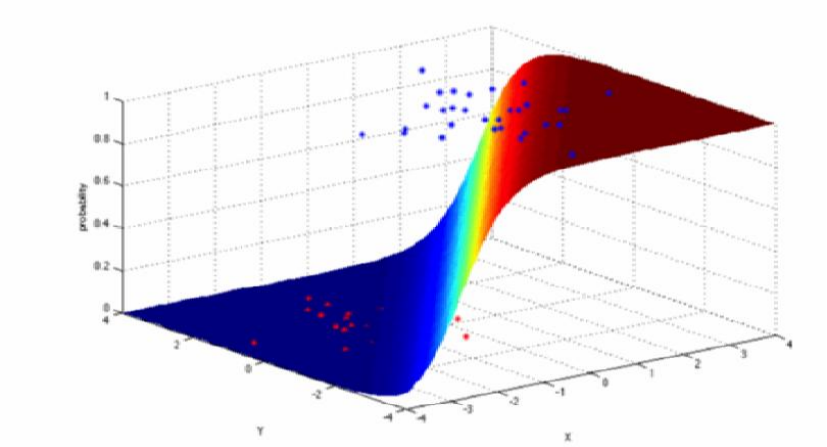

Рис. 3.6. Приклад графіку методу логістичної регресії

Отже, оптимізації цифрового виробництва є важливим завданням для стабільної роботи економіки кожної країни. Використання методів машинного навчання дозволяє підвищити рівень надійності та швидкості виробництва.

3.4 Аналіз даних отриманих під час навчання з використанням шляхів у представленні стану

У цьому розділі будуть представлені статистичні результати процесу навчання. Результати були зібрані за допомогою розробленого модуля ML та метасимулятора.

Агент RL і мета-симулятор були налаштовані для запуску тестового сценарію, попередньо налаштованого на існуючій платформі моделювання, наданій AF.

Кожен тестовий сеанс було налаштовано на виконання 30 разів із різними обмеженнями на загальну кількість епізодів. Після досягнення бажаної кількості проходів, яка в даному випадку становить 720 проходів, модуль можна вважати достатньо навченим. Для маржі була поставлена ціль у 1000 передач. Q-таблиця, яка використовується для цього запуску, зберігається для перевірки. Після кожного сеансу Q-таблиця скидається, щоб агент знову почав вивчати без попереднього знання про раніше оптимальну політику. Це робиться для того, щоб можна було зібрати статистику щодо того, як часто модуль фактично досягає цільової продуктивності.

Швидкість навчання (a) була обрана рівною 0,2, швидкість дослідження (e) була встановлена рівною 0,2 для алгоритму Q-навчання з SARSA без використання e, а коефіцієнт дисконтування (y) був встановлений рівним 0,9. Ці параметри були обрані на основі короткого емпіричного тестування.

Графіки в цьому підрозділі відображають кожну наступну тренувальну сесію з кроком у 100 епізодів, починаючи з 200 епізодів до 1300. Обмеження епізодів після 1300 були з кроком у круглих тисячах до 5000, оскільки спостерігалося, що рівень успіху модуля досяг плато або зменшувався. що межа. Під час цих сеансів підраховувалась кількість досягнутих пропусків, і ця інформація була зібрана та показана тут з інтервалом у 100 епізодів. Обмеження в 100 епізодів було виключено з тестування, оскільки агент не міг досягти цілі з обмеженням у 200 епізодів і тому не досягне її з нижчим лімітом.

Для цих тестів агент використовував стратегію вибору дій e-greedy. Типова тенденція прогону з використанням шляхів починається близько до нуля на початку, коли агент досліджує середовище, і різко зростає до кінця, коли рівень експлуатації зростає, як можна побачити на рисунку 3.7.

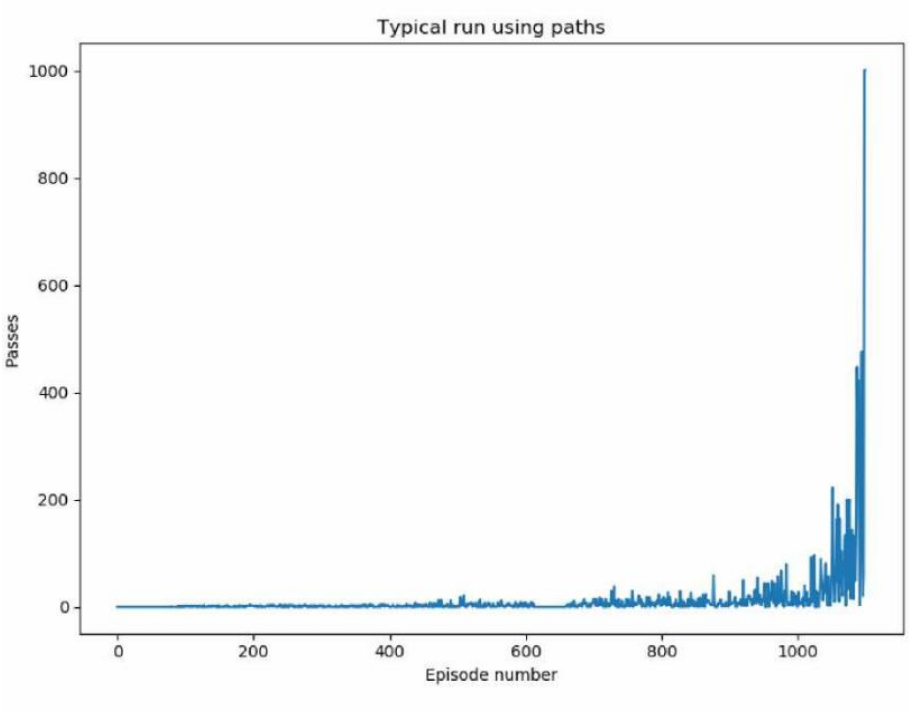

Рис. 3.7. Приклад того, як продуктивність типового запуску з використанням шляхів у складках з часом зростає

Це збільшення пов’язане з падінням значення e, і велике збільшення можна спостерігати, коли e досягає 0, і агент припиняє дослідження та прагне лише використовувати вивчену оптимальну політику.

Тенденція щодо кількості середніх проходжень, досягнутих за прогін, однакова для всіх тестових сесій. Дані прогонів понад ліміт у 1300 епізодів показали, що навчання вийшло на плато та знизилося. Це можна пояснити тим фактом, що після цього моменту все більша кількість запусків у кожному сеансі не досягла цільової продуктивності. Це можна побачити на рисунку 3.8, де після цього обмеження єдиним іншим сеансом, який також мав 50% успіху, був сеанс із обмеженням 2000**.**

Рівень успіху – це те, що принаймні один епізод досягає цільової продуктивності за один запуск. Це означає, що, наприклад, якщо епізод 900 і 901 досягають цільової продуктивності, зараховується лише перший раз, коли вони досягають цілі, оскільки вони все ще виконуються.

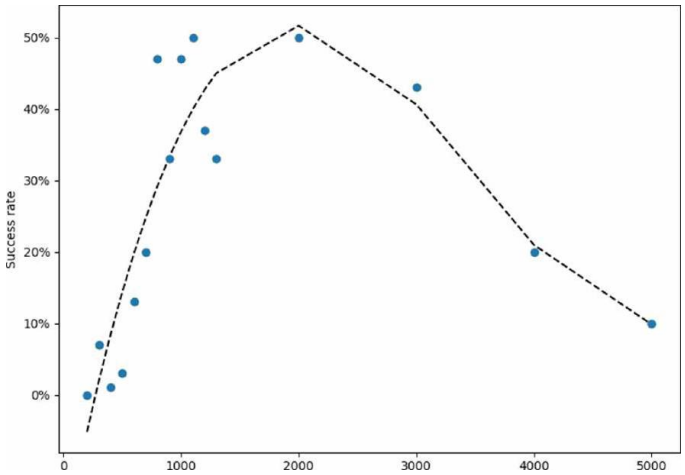

Рис. 3.8. Рівень успішності всіх окремих сеансів тестування за лімітом епізоду. Пунктирна лінія показує загальну тенденцію

Жоден запуск не досяг цільової ефективності при ліміті в 200 епізодів. Кожен сеанс після ліміту в 200 епізодів досягає цільової продуктивності принаймні один раз. Найкращий показник успішності цих сеансів становив 15 із 30 запусків або 50%, де сеанс із найнижчим лімітом епізодів становив 1100, а найвищим 2000. При ліміті 1100 епізодів навчання досягає піку, а рівень успішності знижується після епізоду. ліміт 2000.3.2. Зібрані дані за допомогою сегментів у представленні Рисунок 3.9 Запуск із використанням шляхів у представленні стану, у якому модуль не досягає цільової продуктивності.

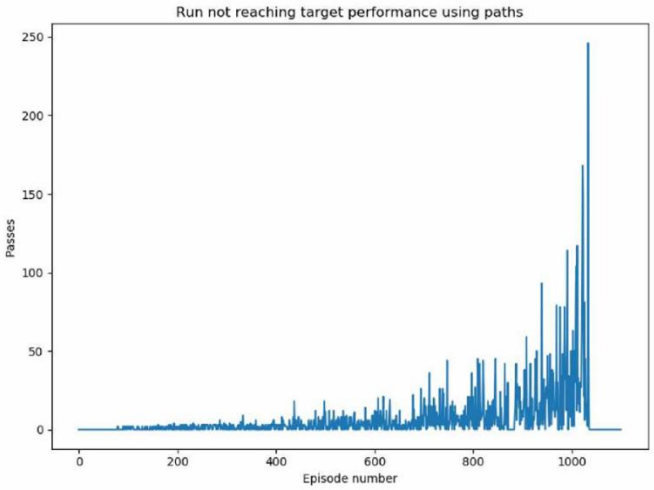

Рис. 3.9 Запуск із використанням шляхів у представленні стану, у якому модуль не досягає цільової продуктивності.

Агент досягає максимуму 249 проходів, а потім падає до нуля, після чого навчання повністю припиняється.

3.5 Дослідження та пердсатвлення отимакних даних за допомогою сегментів у представленні стану

Дані, зібрані під час тестової сесії за допомогою сегментів у представленні стану, були зібрані в 30 прогонах. Ці прогони мали верхню межу 10 000 епізодів або доки не було досягнуто цільової продуктивності, а потім було перезапущено. Під час використання сегментів у представленні стану було помічено, що агент не завершить більше 2 проходів у порівняно великій кількості епізодів N>10000 із стратегією e-greedy. Через це агент був налаштований на використання алгоритму SARSA. Типовий запуск із застосуванням цього методу можна побачити на рисунку 3.10.

Тенденція дещо відрізняється від тих, що спостерігалися під час тестування з використанням шляхів для представлення стану. Як показано на рисунку 3.4, кількість проходів залишалася низькою 10 до епізоду 3000, де кількість проходів зростала, хоча й повільніше, ніж метод із використанням шляхів.

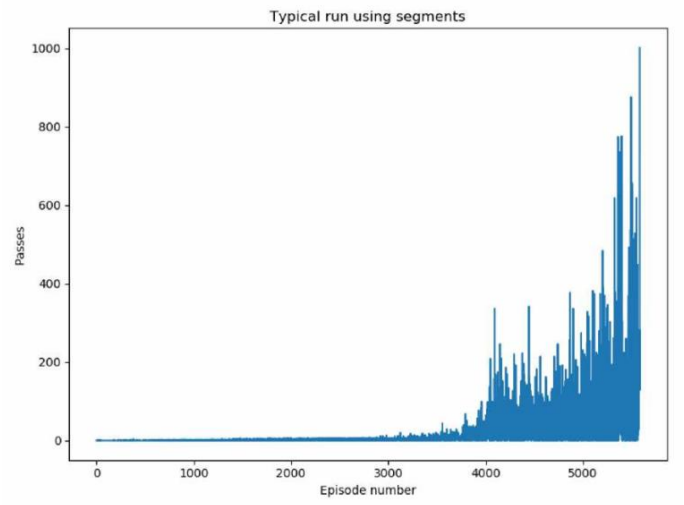

Рис. 3.10. Кількість проходів, які агент виконав за епізод у типовому циклі, використовуючи сегменти для представлення стану

Під час навчання кількість разів, коли агент досяг високої винагороди, щоб потім знизитися до нуля, спостерігалося двічі з 30 разів або 93%. Коли відбулися ці падіння, агент не зміг покращитися далі, і цикл було перезапущено після 10 000 епізодів . Один із цих циклів можна побачити на рисунку 3.11

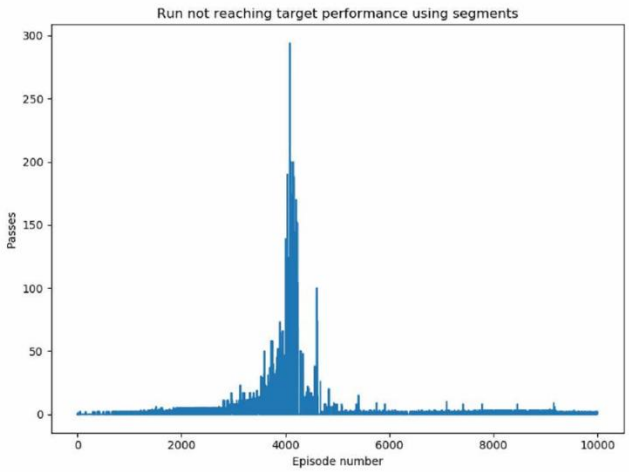

Рис. 3.11 Прогон із використанням сегментів і вибору дій на основі SARSA, де навчання було зупинено після приблизно 4000 епізодів

Цей запуск досягає максимальної продуктивності приблизно через 300 проходів, після чого агент відмовляється від корисної поведінки протягом приблизно 500 епізодів.

3.6 Оцінка точності метасимулятора та узагальнення результатів

Щоб підтвердити, що політики модуля досягають цільової ефективності після завершення навчання, Q-таблиці всіх успішних тренувань було збережено. Під час перевірки вивчених політик усі функції навчання вимкнено, і агент обирає лише оптимальні дії. Цей тест виконується 100 разів з кожною з вибраних Q-таблиць, щоб перевірити, чи політики можуть бути неточними.

З-поміж тестових сеансів із використанням шляхів найвищим показником успішності та найкоротшим часом навчання був той із обмеженням у 1100 епізодів. Тестування цих Q-таблиць призвело до того, що агент досяг цільової ефективності в 100% випадків. Перевірка Q-таблиць з використанням сегментів у представленні стану показала, що агент міг досягти цільової функції.

Оцінка точності метасимулятора

Точність метасимулятора порівнювали з фактичним симулятором і вважали достатньо точним шляхом порівняння виявлення взаємоблокувань і пошуку шляху з оригінальним тренажером. Використовуючи Q-таблицю, сформовану після останнього епізоду навчання, і дозволяючи агенту виконувати лише оптимальні дії, метасимулятор і фактичний симулятор отримували однакові винагороди, якщо всі стани зустрічалися під час навчання. Це було не завжди так, що означало, що в симуляторі були деякі неточності. Навчання та збір даних можна було б зробити набагато швидше за допомогою метасимулятора, а Q-таблицю можна було б пізніше використовувати з фактичною TCS для тестування конкретних сценаріїв.

У разі використання шляхів для представлення стану найкращий ліміт епізоду становив 1100 і використовувався для порівняння. Найкоротший успішний пробіг тривав 21 хвилину, а найдовший – 1 годину. Середній час тренування всіх успішних заїздів у цій сесії склав 36,1 хвилини. З сегментами середня кількість досягла цільової продуктивності в 5032 епізодах, а медіана досягла цільової величини в епізоді 4249. Час, потрібний для досягнення цільової продуктивності з використанням сегментів, становив у середньому 2 години 58 хвилин, найкращий результат – 29 хвилин. а в найгіршому випадку – 5 годин 51 хвилина.

Використовуючи сегменти в представленні стану та алгоритм SARSA, агент RL успішно досяг цільової продуктивності в межах встановленого ліміту

епізоду в 93% випадків. Це значне збільшення показника успіху порівняно з 50% кращих під час навчання з використанням шляхів і Q-навчання.

Спосіб зупинки навчання на шляхах і сегментах відрізняється один від одного. Здається, що продуктивність за допомогою шляхів миттєво падає до нуля на рисунку 3.3 , тоді як запуск із використанням сегментів на рисунку 3.5, здається, з часом відмовляється від корисної поведінки.

Перевірка політик, створених за допомогою шляхів у представленні стану, досягла цільової ефективності в 100% випадків, тоді як політики, що використовують сегменти, досягали цільової ефективності в 82% випадків.

Під час тестування застосовності згенерованих політик за допомогою симулятора AF тестування показало ознаки неточностей у метасимуляторі. Оскільки деякі стани недоступні метасимулятору, політики не можуть досягти цільової продуктивності 24-годинного середовища виконання з TCS AF.

Після періоду навчання з використанням метасимулятора модуль було протестовано на симуляторі AF. Згідно з емпіричним тестуванням, агент виконає правильну дію, що відповідає заданому стану , проте були випадки, коли агент раніше не бачив цього стану і тому не міг щоразу дати правильну відповідь. Ймовірно, це пов’язано з тим, що метасимулятор не є один-на-один із наданою платформою моделювання. Фактори, які можуть вплинути на точність симулятора і, отже, на результати, наприклад прискорення та уповільнення, не були враховані під час побудови метасимулятора.

Таким чином, надана платформа моделювання може досягти станів, яких не може досягти метасимулятор. Можливе вирішення цієї проблеми полягало б у оцінці станів, які ще не досягнуті, наприклад, перегляді k-найближчих сусідів або використання нейронної мережі для оцінки стану. Іншими можливими рішеннями може бути підвищення точності метасимулятора, щоб відповідати
наданому симулятору майнінгу.

Простір станів моделі

Загальний спостережуваний простір станів системи при використанні шляхів можна приблизно оцінити як i8 3 = 5832, що є кількістю зайнятих шляхів у степені кількості всіх машин (без урахування сегментів, які розглядаються як шляхи, які лише одна машина може займати одночасно). Зі збільшенням складності макетів і кількості машин це число різко зросте. Обмеження дискретного простору станів платформи моделювання та метасимулятора може не проявлятися у відносно простому випадку, як цей, але просте додавання іншої машини призводить до простору станів 104976 (18 4) . Це 18-кратне збільшення простору станів, яке вимагатиме принаймні 18-кратного збільшення часу навчання для досягнення подібних результатів до менш складної моделі, хоча збільшення, ймовірно, буде більшим через збільшення логічних труднощів керування іншою машиною .

При використанні сегментів для представлення стану простір станів різко збільшився з 5832. Приблизний простір станів можна обчислити, помноживши кількість способів розміщення двох машин на 157 сегментів на кількість NSP у макеті. Для макета, який використовується в цьому дослідженні, це призводить до (157 виберіть 2) x 9 = ii02i4states. Однак багато з цих станів недосяжні через логічні правила симуляції. Деякі стани можуть бути недоступні через те, що вони миттєво призводять до безвихідних блокувань. Інші можуть бути недоступні через попередні стани, що призводять до тупикових блокувань, тобто стани, що минули від попереднього стану, недоступні. Було помічено, що для досягнення цільової продуктивності агент знайшов 8087 унікальних станів або менше десяти відсотків усіх можливих станів.

У майбутніх дослідженнях більші простори станів можуть оброблятися подібно до того, що було згадано раніше, апроксимація станів за допомогою KNN або CMAC або використовувати інший підхід ML, наприклад

DQN, хоча все одно слід очікувати збільшення часу навчання, якщо більші макети або використовується більше машин.

3.7 Опис можливих помилок через методологію тестування моделі та вибір параметра

Можливі джерела помилок для результатів включають вибрані значення для параметрів, які використовуються алгоритмом Q-навчання (a, y, e). Це можна вважати джерелом помилки. Перевірка різних значень для різних параметрів може вплинути на стабільність навчання та час навчання, особливо зі змінами того, як a та e зменшуються з часом. Тренування проводилося 30 разів з одним набором параметрів. Для кожного зміненого параметра необхідно витратити однакову кількість часу на навчання з цим зміненим параметром, щоб дані були узгодженими. Цього не було зроблено, оскільки деякі сценарії навчання можуть тривати години, залежно від ліміту епізоду та кількості проходів, яких потрібно виконати агенту, щоб вважати його достатньо навченим. Натомість параметри були обрані на основі короткого емпіричного тестування.

Альтернативою фіксованій швидкості навчання може бути використання «правила розміру кроку», яке могло б значно змінити швидкість навчання агента. Однак через невідповідності в процесі навчання, які спостерігалися під час його використання, остаточних результатів не вдалося зібрати, тому він не використовувався. Необхідно провести більше роботи з конфігурацією параметрів для розрахунку правила розміру кроку, щоб визначити його вплив на час навчання та чи можна його зменшити.

Обмеження в 30 сеансів, а також обмеження вибраних епізодів для кожного тестового сеансу також можуть бути розміром вибірки, який не точно відображає продуктивність модуля. Обмеження додаткових епізодів не перевірялися, оскільки спостерігалося, що під час тренування з траєкторіями

тренування стають на плато біля позначки 1100 епізодів. Обмеження під час тренувань із сегментами було встановлено на рівні 10 000, і 28 із 30 пробіжок були успішними. Здавалося, він не вийшов з ладу через низький ліміт, оскільки невдалі запуски, здавалося, вийшли з ладу через те, що навчання в якийсь момент було нестабільним.

Проблеми з нестабільним навчанням. Вибір методу вибору дії.

Як видно з результатів, агент часто досягає точки, коли навчання раптово припиняється, коли використовує шляхи для представлення стану. Це не так часто трапляється при використанні сегментів у представленні стану, які можуть бути пов’язані зі збільшеним простором станів. Це означатиме, що використання шляхів може спричинити надто спрощення стану. У цьому випадку це може призвести до того, що агент не зможе відрізнити два різні стани, які потребують різних дій для досягнення успіху . Однак тестові прогони з використанням сегментів для представлення стану все одно показали таку поведінку. Можливим поясненням цього є те, що порядок, у якому машини досягають різних NSP, може визначати результат виконаної дії. Коли це трапляється, наприклад, машини пройшли повне коло й опинилися в тій самій конфігурації, що й на старті, порядок, у якому вони запитують агента, може відрізняються, що може спричинити дії, які призведуть до результатів, відмінних від тих, що спостерігалися раніше, і спричинити тупикову ситуацію. Це, у свою чергу, призводить до того, що Q-значення для цієї дії стану p air суттєво зменшується, і агент більше не буде віддавати перевагу дії , яка в минулому була оптимальною через цей інший результат. Той факт, що машини можуть запитувати модуль ML лише один раз за зміну стану, може бути причиною посилення проблеми під час використання шляхів, оскільки кількість переходів станів набагато менша.

Рішення цієї проблеми може полягати в тому, щоб включити контекстну інформацію про те, що або der були надіслані запити, тобто який

попередній стан або стани були. Це збільшило б простір станів, особливо при використанні сегментів для представлення стану. Це, у свою чергу, може бути вирішено за допомогою методу наближення стану (наприклад, KNN або CMAC) та/або використання глибшої архітектури, такої як DQN, яка не використовує дискретні стани, а швидше, як Minh et al. згадують у своєму дослідницькому листі «Контроль на рівні людини через глибоке навчання з підкріпленням», оцінює оптимальну функцію дії-цінності за допомогою нейронної мережі [28]. Необхідно провести подальше тестування, щоб підтвердити, чи зможе глибша архітектура вирішити цю проблему.

Вибір методу вибору дії.

Оскільки один неправильний рух може спричинити тупикову блокування та скинути симуляцію до початкового стану, можливо, що використання алгоритму e-greedy є неправильним вибором для проблеми, яку має вирішити модуль ML. Це пов’язано з тим, що дослідження може викликати у агента труднощі з послідовним досягненням станів, більш віддалених від початкового, оскільки кожна дія має шанс вибрати неоптимальну дію. Це проблема, оскільки мета навчання полягає в тому, щоб не спричинити тупик, і тому агент RL повинен досягати та досліджувати стани, розташовані далеко від початкової позиції. Стратегією уникнення цієї проблеми було використання SARSA із сегментами в представництві штату. Під час навчання з використанням сегментів і стратегії вибору дій e-greedy було помічено, що навчання було дуже повільним, аж до точки, коли майже не було досягнуто прогресу після 10 000 епізодів. Ось чому було використано SARSA. Можлива -проблема з використанням SARSA може полягати в тому, що оскільки він досліджує лише найкращий прямий варіант, не гарантовано знайде найоптимальніше рішення. Це може спричинити небажану поведінку, наприклад зупинку агента більше, ніж це необхідно. Альтернативний метод вибору дій міг би покращити стабільність і час підготовки. Можливою

альтернативою було б зробити вибір на основі функції м’якого максимуму, такої як розподіл Гіббса/Больцмана.

3.8 Оптимізація поведінки агента для мінімальних зупинок системи

Не можна було виконати жодного тесту, щоб визначити, як часто агент вибирав найбільш оптимальну дію щодо мінімізації непотрібної зупинки машин. Причиною цього було відсутність набору даних про оптимальну дію в певному стані. Таким чином, можуть бути випадки, коли агент вживає неоптимальних дій і спричиняє більше зупинок, ніж необхідно. Одним із способів вимірювання цього може бути підсумовування загальної кількості дій, які виконав агент, щоб досягти цільової продуктивності. Однак це не означає, наскільки політика була близькою до вибору оптимальних дій. Також неможливо порівняти кількість дій, виконаних агентом з використанням різних представлень стану, оскільки модель, що використовує сегменти, генерує більше запитів, ніж модель, яка використовує шляхи.

Було відмічено, що при використанні іншої структури винагороди — невеликої винагороди за зупинку та трохи більшої винагороди за вихід, агент майже завжди вирішував зупинитися замість вибору йти, за винятком випадків, коли йти було необхідно, щоб уникнути тупикової ситуації. Така поведінка не спостерігалася під час використання структури винагороди.

Одним із способів мінімізації кількості зроблених зупинок може бути встановлення обмеження часу для агента замість цільової кількості пропусків. За допомогою цього методу агент замість лише оптимізації кількості проходів, отриманих без взаємоблокування, оптимізував би -кількість проходів, досягнутих протягом обмеження часу. Це, у свою чергу, може зменшити кількість непотрібних зупинок.

Відмінності в продуктивності між протестованими методами

Як показано в результатах, метод із використанням шляхів навчається набагато швидше, у середньому за 31,6 хвилини порівняно з методом із використанням сегментів, у середньому за 2 години 58 хвилин. Ця відмінність, швидше за все, пов’язана зі збільшенням простору станів. Під час перевірки того, чи можуть згенеровані Q-таблиці працювати протягом 24 годин безперервно, шляхи також показали вищий рівень успішності – 100% порівняно з 82%. Зменшення рівня успіху при використанні сегментів можна пояснити великим простором станів r, а також тим фактом, що навчальні сесії були потрібні лише для завершення одного прогону, де було досягнуто цілі.

Як згадувалося раніше, найбільшою проблемою з використанням шляхів є нестабільність навчання, яка є відносно нестабільною на рівні 50% порівняно з 93% при використанні сегментів. Ймовірно, ця нестабільність пов’язана з браком інформації. Це призначено для зменшення простору станів, але в цьому випадку може спричинити певні проблеми. Підвищення успішності навчання при використанні сегментів у представленні стану також посилює цю гіпотезу.

Проблема з розміщенням машин у випадковому сегменті полягає в тому, що існує набагато більше початкових станів, які агент повинен досліджувати, коли використовує сегменти замість шляхів. Значні падіння продуктивності, які видно на рисунку 3.5 між епізодами при використанні сегментів, ймовірно, пов’язані з цим, тоді як при використанні шляхів причиною може бути як це, так і стратегія e-greedy. Таким чином, одноразове досягнення цільової продуктивності не гарантує її досягнення в усіх початкових станах. Агенту може знадобитися досягти цільової продуктивності більше одного разу з тією самою Q-таблицею, щоб оптимізувати більше пар станів і дій, щоб навчання вважалося завершеним.

Слід зробити вибір стосовно того, який метод найкраще підходить для майбутнього використання та розвитку модуля на основі переваг і недоліків -

обох методів. Оптимальним шляхом було б розробити модуль, який добре працює як під час навчання, так і під час використання, якщо це можливо. Розробка модуля з об'єднанням можливих удосконалень і рішень проблем, які раніше обговорювалися в цій главі, може стати кроком для вдосконалення модуля для отримання бажаних характеристик.

Оскільки розроблений модуль працює для наданого макета, його можна використовувати для зменшення часу, необхідного для налаштування правил дорожнього руху. Час, щоб налаштувати ці правила вручну, займає принаймні робочий тиждень для простіших макетів, подібних до того, що використовувався в експерименті

Таким чином, використання модуля значно зменшить вартість розгортання, оскільки цей модуль може встановлювати правила, щоб уникнути взаємоблокувань за набагато менший час. У гіршому випадку з сеансом із 1100 епізодів із використанням шляхів тренування тривало приблизно одну годину й успішно досягло цільової продуктивності в 50% випадків. При використанні сегментів, які мають більші шанси на успіх у 93%, час, потрібний для досягнення цільової ефективності, становив у гіршому випадку 5 годин і 51 хвилину. Вартість виконання цих симуляцій незначна порівняно з вартістю ручного налаштування правил дорожнього руху. Це може призвести до зниження витрат під час розгортання нових макетів, якщо припустити, що згадані раніше проблеми метасимулятора та самого модуля ML вирішено.

Якби готовий модуль використовувався AF, можна було б уникнути повторної ручної конфігурації та перевірки правил дорожнього руху, а отже, інженери зможуть зосередитися на більш захоплюючій роботі. Ця автоматизація може позитивно вплинути на такі соціальні фактори, як покращення морального духу на робочому місці шляхом мінімізації цих повторюваних завдань.

Як екологічні, так і етичні аспекти не застосовуються до роботи, виконаної в цій дипломній роботі. Екологічні аспекти не застосовуються, оскільки робота транспортних засобів для майнінгу не змінюється, а існуючої комп'ютерної інфраструктури достатньо для роботи з модулем. Одним із можливих аспектів може бути незначне збільшення енергоспоживання комп'ютерів, на яких працює модуль і симуляція, однак це не буде значним збільшенням екологічного сліду видобутку. Етичні аспекти, такі як безпека працівників, які працюють поблизу або в шахті, обробляються нижчими рівнями інфраструктури автоматизації видобутку, тому робота, виконана в цій дисертації, не впливає на ці фактори.

Висновки до розділу

В даному розділі виконано аналіз отриманих результатів використання методів машинного навчання для контролю технологічних процесів. Виконано дослідження концепції цифрового виробництва в контексті контролю технологічних процесів, наведені методи оптимізації технологічних процесів цифрового виробництва. Також описано методи машинного навчання у цифровому виробництві.

ВИСНОВКИ

В магістерській роботі досліджено моделі та методи застосування машинного навчання для контролю технологічних процесів на прикладі оптимізації етапів цифрового виробництва.

Представлено підхід машинного навчання до налаштування правил трафіку для координації кількох автономних машин для майнінгу, щоб уникнути взаємоблокувань за допомогою алгоритмів Q-навчання та SARSA. Точність методів під час перевірки згенерованого набору правил становила 100% для спрощеної моделі. Для більш складної моделі точність згенерованого набору правил склала 82%. З цією моделлю SARSA був єдиним методом, який міг досягти цільової продуктивності протягом встановлених часових обмежень. Це показує, що потрібно зробити більше, щоб підвищити точність до 100%, щоб мінімізувати витрати на ручне усунення помилок.

Визначено, що цільова продуктивність модуля була досягнута в середньому за 2 години 58 хвилин при неспрощеному представленні стану та при спрощеному в 36.1 хвилин. Однак спрощене представлення стану призвело до того, що модуль ML не завершив навчання в 50% випадків, тоді як неспрощений модуль завершив навчання в 93% випадків. Результати показують, що можливо використовувати навчання з підкріпленням для координації роботи трьох машин протягом 24+ годин без участі людини. У ході дослідження було виявлено, що застосування моделей та методів машинного навчання для контролю технологічних процесів у промислових системах є актуальним та перспективним напрямком досліджень. Аналіз технологічних процесів виявив їхню складність та динаміку, що робить актуальним використання адаптивних підходів.

Розроблені моделі машинного навчання виявилися ефективними в прогнозуванні та управлінні технологічними процесами, забезпечуючи високу точність та адаптивність до змін у середовищі. Впровадження цих моделей у

реальні умови дозволило покращити якість контролю та оптимізувати хід технологічних процесів у промислових системах.

Важливим етапом дослідження була оптимізація стратегій контролю, що призвело до підвищення ефективності та ресурсозбереження в промислових виробництвах. Аналіз отриманих результатів підтверджує придатність розроблених моделей та методів для практичного використання.

Висновки дослідження свідчать про значний внесок у розвиток автоматизованого контролю технологічних процесів за допомогою машинного навчання. Розроблені моделі та методи можуть бути використані для підвищення продуктивності, зниження витрат та покращення загальної ефективності в промислових галузях, де контроль технологічних процесів є ключовим елементом.

СПИСОК ПОСИЛАНЬ НА ДЖЕРЕЛА

1. П. Онгсулі, "Штучний інтелект, машинне навчання та глибоке навчання", 2017 р.15-та Міжнародна конференція з ІКТ та інженерії знань (ICT&KE), Бангкок, 2017, стор. 1-6. Процитовано 25-03-2019. ISBN: 978-1-5386-2117-2 https://ieeexplore.ieee.org/document/8259629

2. О. Мохаммед і Дж. Кіанфар, "Підхід машинного навчання до короткотерміновихПрогнозування транспортних потоків: тематичний аналіз міжштатної автомагістралі 64 у Міссурі", 2018 IEEE In international Smart Cities Conference (ISC2), Канзас-Сіті, Міссурі, США, 2018 р., стор. 1-7.Процитовано 05-04-2019.ISBN:978-1-5386-5959-5 https://ieeexplore.ieee.org/ document/ 8656924

3. З. Бартлетт, Л. Хан, Т. Т. Нгуєн і П. Джонсон, «Машинне навчання на основі Підхід до прогнозування транспортного потоку на урбанізованих магістральних дорогах», 2018 IEEE 20th International Conference on High Performance Computing andКомунікації; 16-та міжнародна конференція IEEE про розумне місто; IEEE 4-й Міжнародна конференція з науки про дані та систем (HPCC/SmartCity/DSS), Ексетер, Велика Британія, 2018, стор. 1285-1292. Процитовано 05-04-2019. ISBN: 978-1-5386-6614-: https://ieeexplore.ieee.org/document/8622953

4. D. Kang, Y. Lv, Y. Chen, "Short-term traffic flow prediction with LSTM recurrent neural network", 2017 IEEE 20th International Conference on Intelligent Транспортні системи (ITSC), Йокогама, 2017, стор. 1-6. Процитовано 29-03-2019. ISBN:978-1-4799-0356-6 https://ieeexplore.ieee.org/document/8317872

5. Y. Lee and O. Min, "Long Short-Term Memory Recurrent Neural Network for Прогноз міського транспорту: практичне дослідження Сеула", 2018 21-ша міжнародна конференція з інтелектуальних транспортних систем (ITSC), Мауї, Гаваїна, 2018, стор. 1279-1284. Процитовано 29-03-2019. ISBN: 978-1-7281-0323-5 https://ieeexplore.ieee.org/document/8569620

6. С. Квон, К. Й. Лі, «Генералізація навчання з підкріпленням» з cmac». Тегу, Корея, Університет штату Пенсільванія, США, томи матеріалів IFAC: том 38, випуск 1, 2005, стор.360-365. Процитовано 06-05-2019.: ttps://www.sciencedirect.com/science/article/pii/S1474667016371506

7. К. Джейкоб і Б. Абдулхай, «Інтегроване керування транспортним коридором за допомогою машининавчання", 2005 Міжнародна конференція IEEE з систем, людини та кібернетики, Waikoloa, HI, 2005, стор. 3460-3465 Том. 4. Процитовано 28-03-2019. https://ieeexplore.ieee.org/document/1571683

8. A. Salkham, R. Cunningham, A. Garg and V. Cahill, "A Collaborative Reinforcement Learning Approach to Urban Traffic Control Optimization", 2008 Міжнародна конференція IEEE/WIC/ACM з веб-розвідки та інтелектуAgent Technology, Sydney, NSW, 2008, стор. 560-566. Процитовано 28-03-2019.ISBN: 978-0-7695-3496-1 https://ieeexplore.ieee.org/document/47406849.

9. S. Mikami and Y. Kakazu, "Genetic reinforcement learning for cooperative traffic signal control", Proceedings of the First IEEE Conference on EvolutionaryОбчислення. Всесвітній конгрес IEEE з обчислювального інтелекту, Орландо, Флорида, 1994 р. С. 223-228 т.1. Процитовано 29-03-2019. https://ieeexplore.ieee.org/document/350012

10. Р. С. Саттон і А. Г. Барто, «Навчання з підкріпленням: вступ 2:nd» Видання". Кембридж, Массачусетс, Лондон, Англія: The MIT Press; 2018 рік Розділ 1.1: Навчання з підкріпленням; С. 1-2. ISBN: 978-0262193986

11. Р. С. Саттон і А. Г. Барто, «Навчання з підкріпленням: вступ. 2:nd Видання". Кембридж, Массачусетс, Лондон, Англія: The MIT Press; 2018 рік розділ 1.3 Елементи навчання з підкріпленням, с. 6. ISBN: 978-0262193986

12. С. Саттон і А. Г. Барто, «Навчання з підкріпленням: Вступ. 2:nd Видання". Кембридж, Массачусетс, Лондон, Англія: The MIT Press; 2018 рік розділ 6.5: Q-навчання: контроль TD поза політикою, стор. 131-132. ISBN: 978-0262193986

13. C. Watkins і P. Dayan, “Q-learning”, Machine learning, 1992, стор. 279-292,DOI: https://doi.org/10.1007/BF00992698

14. С. Саттон і А. Г. Барто, «Навчання з підкріпленням: вступ. 2:nd Видання”. Кембридж, Массачусетс, Лондон, Англія: The MIT Press; 2018 рік розділ 6.4 SARSA: контроль TD за політикою, стор. 129.

15. K. De Asis,1 J.F.Hernandez-Garcia,1 G.Zacharias Holland,1 R.S.Sutton, «Лабораторія навчання підкріплення та штучного інтелекту», розділ «Експерименти: світ стохастичної вітряної сітки», Університет Альберти . Процитовано 29-03-2019.: https://arxiv.org/pdf/1703.01327.pdf

16. X. Ma, K. Driggs-Campbell і M. J. Kochenderfer, «Покращена міцність» та безпека для автономного керування автомобілем із навчанням з протилежним підкріпленням", 2018 IEEE Intelligent Vehicles Symposium (IV), Changshu, 2018, стор. 1665-1671 рік. Процитовано 28-03-2019 https://ieeexplore.ieee.org/document/8500450

17. A. Gosavi, «Simulation Based Optimization Parametric Optimization Techniques and Reinforcement Learning Second Edition», стор. 210, 2015 р. ISBN 978-1-4899-7491-4 (електронна книга), Springer New York Heidelberg Dordrecht London. Цитується 01-04-2019.: https://link.springer.com/book/10.1007%2F978-1-4899-7491-4

18. D. L. Poole and A. K. Mackworth, Artificial Intelligence: Foundations of Computational Agents, 2nd Edition, chapter 12.5 Exploration and Exploitation опубліковано Cambridge University Press. https://artint.info/html/ArtInt_266.html

19. І. Гудфеллоу, Ю. Бенгіо, А. Курвіль, «Глибоке навчання». Кембридж, Массачу Сетс, Лондон, Англія: The MIT Press, 2016, стор. 358-359. ISBN: 9780262035613

20. A. Graves, A. Mohamed and G. Hinton, "Speech recognition with deep recurrent neural networks", 2013 IEEE International Conference on Acoustics, Speechі обробка сигналів, Ванкувер, Британська Колумбія, 2013, стор. 6645-

6649. Процитовано 27-03-2019. ISBN: 978-1-4799-0356-6 https://ieeexplore.ieee.org/document/6638947

21. І. Гудфеллоу, Ю. Бенгіо, А. Курвіль, «Глибоке навчання». Кембридж, Массачу Сетс, Лондон, Англія: The MIT Press, 2016, стор. 194-195. ISBN: 9780262035613

22. І. Гудфеллоу, Ю. Бенгіо, А. Курвіль, «Глибоке навчання». Кембридж, Массачу Сетс, Лондон, Англія: The MIT Press, 2016, стор. 369. ISBN: 9780262035613

23. Р. Паскану, Т. Міколов, Ю. Бенгіо, «Про труднощі навчання рекурентних нейронних мереж» на Міжнародній конференції з машинного навчання, 2013 р. (стор. 1310-1318). Процитовано 04.01.2019. https://arxiv.org/abs/1211.5063

24. А. Аміді та С. Аміді. “Шпаргалка з повторюваними нейронними мережами”. Цитовано 28-03- 2019. https://stanford.edu/%7Eshervine/teaching/cs-230/cheatsheet-recurrent-neural networks#architecture

25. І. Гудфеллоу, Ю. Бенгіо, А. Курвіль, «Глибоке навчання». Кембридж, Массачу Сетс, Лондон, Англія: The MIT Press, 2016, стор. 398. ISBN: 9780262035613

26. І. Гудфеллоу, Ю. Бенгіо, А. Курвіль, «Глибоке навчання». Кембридж, Массачу Сетс, Лондон, Англія: The MIT Press, 2016, стор. 278. ISBN: 9780262035613

27. І. Гудфеллоу, Ю. Бенгіо, А. Курвіль, «Глибоке навчання». Кембридж, Массачу Сетс, Лондон, Англія: The MIT Press, 2016, стор. 392. ISBN: 9780262035613

28. В. Мніх та ін. al, «Контроль на рівні людини через глибоке навчання з підкріпленням», Лондон, Англія, Macmillan Publishers, Nature том 518, 2015, стор. 529–533. DOI: https://doi.org/10.1038/nature14236

29. Добруцький, К. А. Огляд та порівняльний аналіз алгоритмів та методів кластеризації та регресії [Текст] /К. А. Добруцький. – М.: Алгоритмів та Методи, 2017. – 272 с..

30. Нейронні мережі для початківців. Частина 1 [Електронний ресурс]. – Режим доступу: http://it-ua.info/news/2016/10/12/neyronn-merezhdlya-pochatkvcv-chastina-1.html.

31. Дерево прийняття рішень [Електронний ресурс]. – Режим доступу: https://basegroup.ua/community/articles/description.

32. Оцінка результатів лінійної регресії [Електронний ресурс]. – Режим доступу: https://habr.com/post/195146/.

33. Соколова О.А., Вислоух С.П. Застосування методу евристичної самоорганізації моделей при розв’язанні технологічних задач. Погляд у майбутнє приладобудування : зб. праць XII Всеукр. наук.-практ. конф. студентів, аспірантів та молодих вчених. Київ : Центр учбової літератури, 2019. С. 190-192.

34. What is an artificial neural network? Here’s everything you need to know. Digital Trends : веб-сайт. URL: https://www.digitaltrends.com/cool-tech/what-is-an-artificial-neuralnetwork/

35. 5 Ways Machine Learning Is Leading to Smarter Manufacturing. business.com : веб-сайт. URL: https://www.business.com/articles/machine-learning-and-manufacturing/

36. Asset Monitoring & Predictive Maintenance. Deloitte Digital : веб-сайт. URL: https://www2.deloitte.com/content/dam/Deloitte/us/Documents/about-deloitte/ us-a-turnkey-iotsolution-for-manufacturing.pdf

37. Smartening up with Artificial Intelligence (AI) - What’s in it for Germany and its Industrial Sector? Digital/McKinsey : веб-сайт. URL: https://www.mckinsey.com/~/media/McKinsey/Industries/Semiconductors/Our%20Insights/S

38. Why Machine Learning Is Important For Smarter Manufacturing. towards data science : веб-сайт. URL: https://towardsdatascience.com/why-machine-learning-is-importantfor-smarter-manufacturing-27da545de779

39. What is zero trust privilege? Centrify : веб-сайт. URL: https://www.centrify.com/education/what-is-zero-trust-privilege/

40. Top 27 Artificial Neural Network Software. PatResearch : веб-сайт. URL: https://www.predictiveanalyticstoday.com/top-artificial-neural-network-software/

41. An end-to-end open source machine learning platform. TensorFlow : веб-сайт. URL: https://www.tensorflow.org/

42. Best Artificial Neural Network Software. G2 : веб-сайт. URL: https://www.g2.com/categories/artificial-neuralnetwork?utf8=%E2%9C%93&order

43. Neural Network Toolbox. Packtx : веб-сайт. URL: https://subscription.packtpub.com/book/big_data_and_business_intelligence/9781788398435/1/01lvl1sec14/neural-network-toolbox