МАГІСТЕРСЬКА РОБОТА

МР. ІПм - 50.00.00.000 ПЗ

Група ІПм-22-1

Бойко Назар

2024

Івано-Франківський національний технічний університет нафти і газу

Інститут інформаційних технологій

Кафедра інженерії програмного забезпечення

Бойко Назар Іванович

(прізвище, ім'я, по батькові)

УДК 004.942
(індекс)

МАГІСТЕРСЬКА РОБОТА

Моделі оптимізації профілів ігрових персонажів героїв в реальному часі з використанням алгоритмів машинного навчання

(назва роботи)

Інженерія програмного забезпечення

(назва освітньої програми)

121 - Інженерія програмного забезпечення

(шифр і назва спеціальності)

Бойко Н.І
(підпис, ініціали та прізвище здобувача освітнього ступеня)

Науковий керівник Лютак Ігор Зіновійович, д.т.н., професор
(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

Освітньо-кваліфікаційний 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

Бойку Назару Івановичу

(прізвище, ім’я, по-батькові)

1. Тема магістерської роботи **“**Моделі оптимізації профілів ігрових персонажів героїв в реальному часі з використанням алгоритмів машинного навчання**”**
керівник проекту (роботи) Лютак Ігор Зіновійович, д.т.н., доцент
затверджені наказом закладу вищої освіти від **“** 18 ” грудня 2023 р. № 738/7
2. Строк подання студентом проекту (роботи) 15 січня 2024 р.
3. Вихідні дані до проекту (роботи) Архітектура, формальний опис та алгоритми машинного навчання
4. Зміст розрахунково - пояснювальної записки(перелік питань, які потрібно розробити)
1. Теоретичні відомості про алгоритми машинного навчання та способи їх реалізації
2. Сучасні технології алгоритмів машинного навчання та отримання даних для тренування
3. Алгоритм оптимізації профілів ігрових персонажів та огляд існуючих програмних засобів
4. Розробка алгоритму та програмної реалізації системи оптимізації профілів ігрових персонажів реальному часі
5. Перелік графічного матеріалу (з точним зазначенням обов’язкових креслень)
1. Архітектура LSTM (рис. 2.6, ст. 45)
2. Графік вибору здібності в послідовності драфту (рис. 3.1, ст. 55)
3. Порівняння CBOW та Slip-gram (рис. 3.6, ст. 59)
4. Результати тренувань (рис. 3.8, ст. 64)
5. Приклад інтерфейсу з однією рекомендацією (рис. 4.9, ст. 87)

6. Консультанти розділів проекту (роботи)

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 04 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	20.09.2023	виконано
2	Аналіз сучасних технологій машинного навчання	01.10.2023	виконано
3	Способи отримання даних для тренування моделей	20.10.2023	виконано
4	Дослідження алгоритмів машинного навчання та огляд існуючих рішень	15.11.2023	виконано
5	Формулювання вимог та алгоритмів функціонування системи	03.12.2023	виконано
6	Програмна реалізація рішення	22.12.2023	виконано
7	Затвердження пояснювальної записки роботи завідувачем кафедри	15.01.2024	виконано

Студент – магістр _____
(підпис)

Керівник роботи _____
(підпис)

АНОТАЦІЯ

Магістерська робота: 93 с., 33 рис., 40 джерел.

Тема: Моделі оптимізації профілів ігрових персонажів героїв в реальному часі з використанням алгоритмів машинного навчання.

Об'єкт дослідження: моделі та алгоритми машинного навчання для оптимізації профілів ігрових персонажів героїв.

Мета роботи: розробка моделі та програмного застосунку для оптимізації профілів ігрових персонажів героїв в реальному часі.

Предмет дослідження: алгоритми машинного навчання та інтеграція моделі в програмний застосунок.

Результати дослідження:

Виконано аналіз існуючих алгоритмів машинного навчання в області передбачення результатів матчу і на його основі запропоновано власну модель та створення застосунку для роботу моделі в реальному часі.

Висновок:

В результаті досліджень було отримано власну модель машинного навчання, яка базується на даних попередніх матчів, що вирішує проблему оптимізації профілів ігрових персонажів героїв в реальному часі.

АЛГОРИТМИ МАШИННОГО НАВЧАННЯ, ШТУЧНА НЕЙРОННА МЕРЕЖА, РОЗРОБКА ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ, ДОБУВАННЯ ДАНИХ.

ANNOTATION

Master's thesis: 93 p., 33 figures, 40 sources.

Theme: Models for optimizing game character profiles of heroes in real time using machine learning algorithms

Object of research: models and machine learning algorithms for optimizing the profiles of game characters.

Purpose: development of a model and software application for optimizing the profiles of game characters in real time.

Subject of research: machine learning algorithms and integration of the model into a software application.

Research results:

An analysis of existing machine learning algorithms in the field of match prediction was performed and, based on it, our own model was proposed and an application was created for the model to work in real time.

Conclusion:

As a result of the research, we have obtained our own machine learning model based on the data of previous matches, which solves the problem of optimizing the profiles of game characters of heroes in real time.

MACHINE LEARNING ALGORITHMS, ARTIFICIAL NEURAL NETWORK, SOFTWARE DEVELOPMENT, DATA MINING.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ, СКОРОЧЕНЬ І ТЕРМІНІВ

API - прикладний програмний інтерфейс

LSTM - Довга короткочасна пам'ять

OpenCV - бібліотека комп’ютерного зору з відкритим вихідним кодом

RNN - Рекурентна нейронна мережа

KNN - Метод k-найближчих сусідів

MOBA - Багатокористувацька онлайнова бойова арена

ВСТУП

Актуальність роботи

Актуальність роботи полягає в її глибоких наслідках для швидкозмінного ландшафту конкурентних ігор, особливо в контексті Dota 2. Як висококонкурентний кіберспорт, Dota 2 вимагає стратегічного вибору героя, а оптимізація профілів персонажів має вирішальне значення для успіху. Дослідження алгоритмів машинного навчання, зокрема довготривалої короткочасної пам'яті (LSTM), у цій галузі дає цінну інформацію для покращення процесів прийняття рішень під час створення героїв.

Важливість дослідження підкреслюється його багатогранним підходом, що враховує як тонкощі взаємодії героїв, так і індивідуальні навички гравців. Кидаючи виклик загальноприйнятій думці про однакову майстерність гравців, дослідження підкреслює важливість врахування специфічних чинників гравців у процесі драфту. Цей відхід від традиційних підходів відкриває шляхи для більш тонких та ефективних стратегій у змагальних іграх.

Крім того, запропонований метод, що працює в режимі реального часу, з його інтеграцією в інтерфейс PyQt, надає теоретичним засадам практичної застосовності. Модульний дизайн програмного забезпечення, що розділяє користувальницький інтерфейс, логіку та компоненти моделі, що навчається, не тільки спрощує процес реалізації, але й створює основу для масштабованості та майбутніх удосконалень. По суті, ця робота має значення не лише через її внесок у сферу оптимізації героїв, керованих ШІ, але й через її потенціал формувати майбутній ландшафт змагальних ігор, надаючи дієві ідеї та практичні інструменти для гравців, команд та ігрової спільноти загалом.

Порівняння роботи з відомими розв’язаннями проблеми

Порівнюючи представлену роботу з оптимізації профілів ігрових персонажів у Dota 2 з існуючими рішеннями, очевидно, що це дослідження вирізняється кількома

ключовими аспектами. У той час як попередні дослідження були зосереджені переважно на взаємодії героїв і складах команд, ця робота представляє комплексний підхід шляхом включення індивідуальних навичок гравців у процес прийняття рішень під час створення героя.

Існуючі рішення, наприклад, ті, що використовують механізми рекомендацій або статистичний аналіз вибору героїв, часто не враховують нюанси внеску окремих гравців. Включення алгоритмів машинного навчання, зокрема мереж довготривалої короткочасної пам'яті (LSTM), вирізняє цю роботу з-поміж інших завдяки динамічному моделюванню взаємодії між гравцями та героями. Це дозволяє краще зрозуміти динаміку гри, що призводить до персоналізованих та контекстно-орієнтованих рекомендацій для героїв.

Крім того, реалізація запропонованої моделі в реальному часі за допомогою інтерфейсу PyQt додає шар практичності, що відрізняє її від багатьох теоретичних підходів. Модульний дизайн програмного забезпечення, що розділяє користувацький інтерфейс, логіку та навчені компоненти моделі, відповідає найкращим практикам розробки програмного забезпечення. Це не тільки покращує ремонтопридатність і розширюваність системи, але й сприяє безперешкодній інтеграції з технологіями та наборами даних, що розвиваються.

По суті, порівняння роботи з існуючими рішеннями показує її інноваційний внесок у подолання розриву між індивідуальними навичками гравців та стратегіями створення героїв, пропонуючи цілісне та ефективне рішення для оптимізації профілів ігрових персонажів у Dota 2.

Мета і задачі дослідження

Метою розробка моделі та програмного застосунку для оптимізації профілів ігрових персонажів героїв в реальному часі.

Досліджувана модель повинна реалізовувати взаємодію між компонентами програмного забезпечення та тренованої моделі для оптимізації профілів ігрових

персонажів героїв в реальному часі. Система повинна забезпечувати достатньо високий рівень точності рекомендованих здібностей та бути зручною у використанні.

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

1)огляд існуючих способів оптимізації профілів ігрових персонажів героїв в реальному часі;

2)дослідження алгоритмів машинного навчання;

3)аналіз та порівняння механізму оптимізації профілів ігрових персонажів героїв в реальному часі;

4)вибір оптимальної моделі машинного навчання та обґрунтування доцільності його використання;

5)програмна реалізація оптимізації профілів ігрових персонажів героїв та інтеграція тренованої моделі.

Об'єктом дослідження є моделі та алгоритми машинного навчання для оптимізації профілів ігрових персонажів героїв в реальному часі.

Предметом дослідження є алгоритми машинного навчання та інтеграція моделі в програмний застосунок.

Методи дослідження є багатогранний підхід, що поєднує збір даних через OpenDota API, застосування алгоритмів машинного навчання з фокусом на LSTM-мережі та розробку інтерфейсу PyQt для прийняття рішень у реальному часі під час створення героїв. Ця цілісна методологія дозволила детально дослідити взаємозв'язок між індивідуальними навичками гравців та вибором героя, що призвело до створення надійної програмної реалізації. Ефективність моделі була ретельно оцінена, що підкреслило її потенціал для покращення процесів прийняття рішень в ігровому процесі Dota 2.

Наукова новизна одержаних результатів

Наукова новизна дослідження полягає в інтеграції мереж LSTM для прийняття рішень у реальному часі в процесі вибору здібностей Dota 2 - унікальному застосуванні в галузі машинного навчання та ігор. Використовуючи здатність LSTM фіксувати довгострокові залежності, дослідження впроваджує інноваційний підхід до оптимізації профілів ігрових персонажів, сприяючи розвитку ландшафту додатків ШІ в сфері кіберспорту. Результати демонструють нові ідеї, отримані в результаті моделювання взаємодії гравця та героя, розширюючи межі прогностичного моделювання в ігрових сценаріях, що передбачають конкуренцію.

Практичне значення одержаних результатів.

На основі проведеного дослідження було створено та натреновано модель, яка інтегрована в програмне забезпечення, яке може бути використано для покращення рівню гри на будь-якому рівні компетентності.

Структура магістерської роботи.

Магістерська робота викладена на 93 сторінках друкованого тексту, який складається з вступу, чотирьох розділів, висновків, списку використаних джерел (40 найменувань). Робота містить 33 рисунків.

РОЗДІЛ 1
ОЗНАЙОМЛЕННЯ З МОДЕЛЮ ОБРАНОЇ ГРИ. ОГЛЯД ЛІТЕРАТУРИ ТА ІСНУЮЧИИХ ПРОГРАМНИХ РІШЕНЬ

1.1 Опис ігрового середовища комп'ютерної гри Dota 2

Dota 2, розроблена корпорацією Valve, є дуже популярною і складною відеогрою у жанрі багатокористувацька онлайн бойова арена (MOBA). Випущена як продовження оригінальної гри Defense of the Ancients (DotA), Dota 2 стала основною в кіберспортивній спільноті з моменту її запуску в 2013 році. Гра створена на ігровому рушію Source 2, в ній змагаються дві команди з п'яти гравців, кожна з яких керує унікальним героєм з певними здібностями та ролями. Основна мета полягає в стратегічній навігації полем бою, відомим як карта, і знищенні стародавньої споруди команди суперника.

Змагальність Dota 2 полягає у великому виборі героїв, складній деталізації предметів та динамічній механіці гри. Герої поділяються на три основні категорії: сила, спритність та інтелект, кожна з яких впливає на їхні сильні та слабкі сторони. Синергія командного складу, індивідуальних навичок гравців та прийняття стратегічних рішень визначає результат матчів. Метагра, що постійно розвивається, та постійні оновлення сприяють її довговічності, формуючи віддану спільноту гравців та яскраву кіберспортивну екосистему.

Крім того, завдяки своїй складності Dota 2 слугує чудовою платформою для досліджень та експериментів, надаючи широкі можливості для вивчення додатків машинного навчання у сценаріях прийняття рішень у реальному часі в умовах високої конкуренції в ігровому середовищі. Складний характер гри робить Dota 2 ідеальним об'єктом для вивчення та оптимізації профілів ігрових персонажів за допомогою передових обчислювальних моделей. Dota 2, з її складним і динамічним ігровим процесом, слугує чудовою моделлю для вивчення та застосування алгоритмів машинного навчання завдяки кільком ключовим факторам.

Складність і різноманітність

Dota 2 може похвалитися великим і різноманітним пулом героїв, кожен з яких має унікальні здібності. Широке поєднання героїв та їхніх здібностей забезпечує високий рівень складності, надаючи широкі можливості моделям машинного навчання для вивчення та оптимізації профілів персонажів на основі конкретних ігрових сценаріїв.

Предметизація

У грі представлений широкий набір предметів, які можна придбати та екіпірувати героїв. Алгоритми машинного навчання можуть досліджувати оптимальні стратегії екіпіровки відповідно до мінливих потреб матчу, враховуючи такі фактори, як вибір суперників, склад команди та хід гри.

Прийняття рішень у реальному часі

Стратегія Dota 2 в режимі реального часу та постійна динаміка ігрового середовища створюють виклик для прийняття рішень. Моделі машинного навчання можна навчити приймати рішення в реальному часі щодо пересування героя, використання навичок та купівлі предметів, адаптуючись до динамічного потоку матчу.

Невизначеність, притаманна змагальній грі, в поєднанні зі складністю простору прийняття рішень, забезпечує реалістичне середовище для тестування адаптивності та можливостей прийняття рішень алгоритмами машинного навчання.

Поведінка і стратегія гравців

Dota 2 має різноманітну базу гравців з різними стилями гри, стратегіями та рівнями навичок. Вивчення поведінки та стратегій гравців за допомогою моделей машинного навчання може привести до розуміння оптимальних моделей прийняття рішень, допомагаючи адаптувати профілі персонажів для різних типів гравців.

Метагра в Dota 2 постійно розвивається, оскільки з'являються нові стратегії, а герої отримують баффи або нерфи. Алгоритми машинного навчання можуть аналізувати тенденції в метагрі, виявляючи закономірності, які можуть вплинути на оптимальні профілі персонажів і стратегії прийняття рішень.

Кіберспорт як бенчмарк

Популярність Dota 2 на кіберспортивній сцені є еталоном для оцінки ефективності моделей машинного навчання. Вивчення того, як ці моделі працюють проти професійних гравців у змаганнях з високими ставками, може виявити справжній потенціал і обмеження алгоритмів.

Dota 2 пропонує розгалужену систему повторів, що дозволяє дослідникам аналізувати минулі матчі та використовувати ці дані для навчання та тестування моделей машинного навчання. Такий підхід полегшує розробку алгоритмів, які можуть навчатися на історичних даних, щоб оптимізувати профілі персонажів для майбутніх матчів.

Міждисциплінарні ідеї

Можливості для співпраці: Перетин ігор, штучного інтелекту та машинного навчання надає унікальну можливість для міждисциплінарної співпраці. Дослідники з ігрової індустрії, комп'ютерних наук та науки про дані можуть працювати разом, щоб поглибити розуміння оптимального прийняття рішень у складних середовищах у режимі реального часу.

Загалом, багате і багатогранне ігрове середовище Dota 2 пропонує захоплюючу платформу для вивчення потенціалу алгоритмів машинного навчання в оптимізації профілів ігрових персонажів. Знання, отримані в результаті таких досліджень, можуть виходити за рамки ігор, сприяючи ширшому застосуванню штучного інтелекту та прийняттю рішень у динамічних і невизначених сферах.

Хоча Dota 2 представляє інтригуюче і складне середовище для вивчення алгоритмів машинного навчання, важливо визнати кілька недоліків і проблем, пов'язаних з використанням гри в якості моделі:

Висока розмірність

Великий функціональний простір: Величезна кількість змінних, включаючи здібності героїв, предмети, рухи гравців і склади команд, сприяє створенню високорозмірного простору ознак. Моделі машинного навчання можуть боротися з прокляттям розмірності, що призводить до збільшення обчислювальної складності та потенційного перенавчання.

Динамічний і непередбачуваний ігровий процес

Невизначеність: Динамічна природа Dota 2 вносить у гру невизначеність і непередбачуваність. Раптові зміни в стратегії, несподівані дії гравців і тенденції в метаіграх, що розвиваються, можуть ускладнити для моделей машинного навчання точне прогнозування та оптимізацію профілів персонажів.

Обмежена доступність маркованих даних

Збір маркованих навчальних даних для моделей машинного навчання в Dota 2 може бути складним завданням через розрідженість певних подій і результатів. Анотовані дані, такі як точна інформація про прийняття оптимальних рішень або наміри гравців, можуть бути обмеженими, що впливає на здатність моделі до ефективного узагальнення.

Складність прийняття рішень гравцями

Розуміння та моделювання процесу прийняття рішень людиною в Dota 2 за своєю суттю є складним. Гравці можуть приймати рішення на основі інтуїції, досвіду або командної комунікації - аспектів, які може бути важко точно врахувати в рамках машинного навчання.

Виправлення та мета-зміни

У Dota 2 регулярно виходять оновлення, включаючи зміни балансу героїв та зміни в ігровій механіці. Ці систематичні патчі можуть впливати на ефективність навчених моделей, вимагаючи постійної адаптації та перенавчання, щоб залишатися актуальними.

Обчислювальні ресурси

Високі обчислювальні витрати: Навчання моделей машинного навчання, особливо моделей глибокого навчання, таких як LSTM, на великих наборах даних з Dota 2 може вимагати значних обчислювальних ресурсів. Це може створити проблеми для дослідників з обмеженим доступом до високопродуктивної обчислювальної інфраструктури.

Етичні міркування

Вплив на досвід гравців: Впровадження моделей машинного навчання, які оптимізують профілі персонажів, може потенційно вплинути на загальний досвід гравців. Непередбачувані наслідки, такі як дисбаланс в ігровому процесі або розчарування серед гравців, які зіткнулися з оптимізованими профілями, повинні бути ретельно розглянуті.

Узагальнення за рівнями навичок

Dota 2 має широкий спектр рівнів майстерності, від випадкових гравців до професійних кіберспортсменів. Розробка моделей машинного навчання, які можуть узагальнювати цей спектр навичок, є нетривіальним завданням, оскільки оптимальні стратегії можуть суттєво відрізнятися.

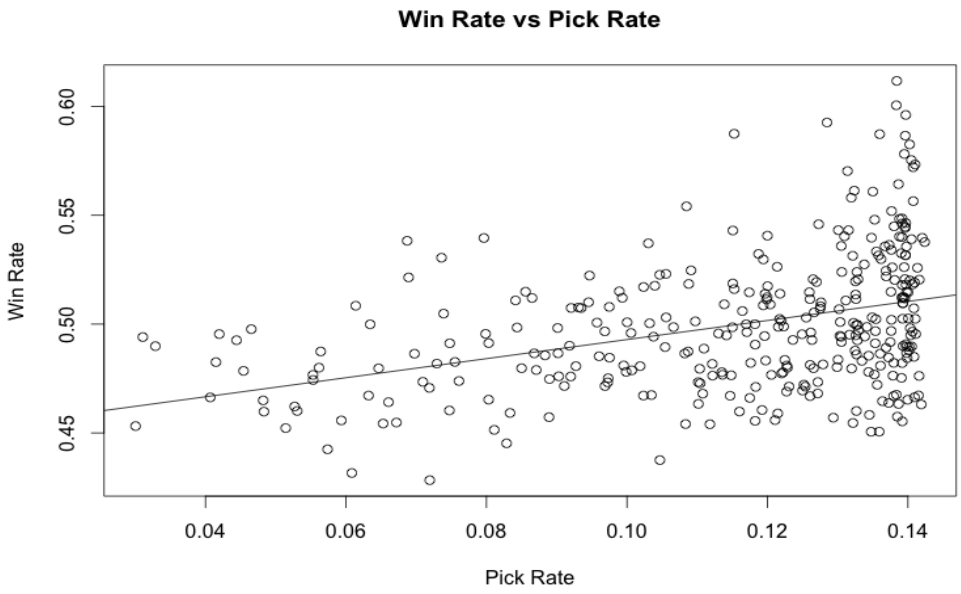


Рис. 1.1. Залежність частоти вибору до частоти перемоги

Міждисциплінарні виклики

Незважаючи на ці проблеми, їх вирішення може призвести до цінних ідей і досягнень як в галузі ігор, так і в галузі машинного навчання. Ретельний аналіз і чітко визначена стратегія дослідження є важливими для подолання цих недоліків і використання Dota 2 як значущої моделі для вивчення алгоритмів машинного навчання.

1.2 Аналіз літературних джерел по темі дослідження

У нашому огляді літератури ми заглибилися в сучасні статті та різноманітні змагання, присвячені прогнозуванню перемоги у Dota 2. Також розглянули методики вдосконалення збору даних. Зокрема, в одному з досліджень [6] були використані стандартні моделі машинного навчання та методи оптимізації, що дозволило досягти вражаючої точності до 85% протягом перших 5 хвилин ігрового процесу. У дослідженні використовувався набір даних з 5,7 тис. випадків з престижного міжнародного турніру з Dota 2 2017 року.

Інший вартий уваги внесок, як зазначено в [12], використовував двоетапний підхід. На початковому етапі автори провели комплексну кількісну оцінку героїв Dota 2 на основі 17 різних аспектів, що значно підвищило точність. Згодом було запропоновано новий метод представлення драфтів героїв. Для цього було створено таблицю пріоритетів, що охоплює 113 героїв, з використанням попередніх знань для підтримки розробленої методології. У дослідженні було проведено ретельне порівняння та аналіз оціночних показників різних методів машинного навчання, застосованих до цього завдання, що дало цінну інформацію про ефективність різних підходів.

У роботі [8] основна увага була зосереджена на вдосконаленні результатів прогнозування існуючої моделі шляхом ретельного вдосконалення збору даних, вилучення ознак та кодування ознак. Автори представили нову концепцію, яка називається доповненою регресією, де ймовірність успіху визначалася як відношення обраних героїв, які досягли успіху, до загальної кількості обраних героїв. Крім того, остаточна ймовірність успіху визначалася як середнє значення між ймовірністю регресії та розрахованою ймовірністю успіху. Якщо остаточна ймовірність успіху перевищувала 0,5, робився прогноз "Блискучої" перемоги; в іншому випадку робився прогноз "Похмурої" перемоги. Метрики, що використовувалися, включали індивідуальний внесок героя та синергію між героями в матчі.

Дослідження показало, що точність чистої логістичної регресії асимптотично наближається до 69,42%. Цікаво, що автори підкреслили важливий висновок - вирішальний вплив вибору героя на результати гри. Завдяки доповненій логістичній регресії показник точності тесту помітно покращився, досягнувши 74,1%. Таке порівняння двох моделей підкреслило ефективність запропонованого доповнення у вдосконаленні прогностичних можливостей, проливаючи світло на нюанси динаміки вибору героя та його глибокий вплив на результати матчів.

Конлі та Перрі першими визнали ключову роль інформації, отриманої на етапі підготовки до гри, застосувавши у своїй фундаментальній роботі як логістичну регресію, так і метод k-найближчих сусідів (kNN) [2]. Їхня перша спроба застосувати логістичну регресію дала точність тесту 69,8% на великому наборі даних, що складався з 18 000 прикладів. Однак вони виявили обмеження: Логістична регресія не змогла врахувати складні синергії та антагоністичні стосунки між героями, як всередині команд, так і між ними.

У відповідь на цей виклик автори запровадили kNN, застосувавши кастомні ваги для сусідів і використовуючи різні метрики відстані. За допомогою суворого процесу 2-кратної перехресної перевірки на наборі даних з 20 000 матчів вони систематично визначали оптимальні параметри для kNN, особливо зосереджуючись на параметрі розмірності (d). Після ретельного налаштування вони визначили, що оптимальна розмірність d дорівнює 4, в результаті чого точність перехресної перевірки становить 67,43%. Ця вдосконалена kNN-модель продемонструвала точність тестування 70% на значному наборі даних, що складається з 50 000 випадків. Ретельне дослідження різних алгоритмів і параметрів підкреслило важливість нюансованого підходу до використання машинного навчання для прогнозування перемоги у Dota 2, особливо на етапі драфту гри.

Результати дослідження [10] роблять значний внесок у вивчення моделей машинного навчання для прогнозування перемоги у Dota 2. Автори провели всебічне порівняння різних моделей машинного навчання, включаючи наївний класифікатор

Байєса, логістичну регресію (що повторює попередні результати), градієнтні дерева рішень, машини факторизації та дерева рішень. Вибір моделей був зроблений зі стратегічної точки зору, враховуючи здатність моделей обробляти розріджені дані. Метрики оцінки, такі як AUC (площа під кривою) і Log-Loss, виміряні за допомогою десятикратної перехресної перевірки, показали багатообіцяючу якість прогнозування. Зокрема, XGBoost продемонстрував результат AUC у 70%, перевершивши логістичну регресію та наївний Байєс з 68%. Порівняння було доповнено оцінкою результатів за різними наборами навичок: нормальним, високим і дуже високим.

Крім того, Куан'ян Сонг, Тяньї Чжан та Чао Ма [11] дослідили прогностичні можливості логістичної регресії у визначенні переможця в іграх Dota 2 на основі складів героїв. Для збору даних вони використовували API, наданий розробником гри. Завдяки помилкам навчання та тестування вони представили вичерпну ілюстрацію результатів, зробивши значущі висновки зі своїх знахідок.

На противагу цьому, [7] використовували окрему методологію, використовуючи повтори в Dota 2 і тестуючи різні класифікатори. Варто зазначити, що ці статті та наше поточне дослідження розходяться у своїх підходах та методологіях. У той час як ми зосереджені на застосуванні моделей машинного навчання, вищезгадані дослідження вивчають альтернативні методи та класифікатори для аналізу своїх проектів. Таке розмаїття методологій у різних дослідженнях збагачує загальне розуміння прогностичний моделювання в контексті Dota 2.

У захоплюючому дослідженні, представленому в [1], для аналізу часових рядів змін у стані здоров'я героїв було використано модель ARMA (авторегресійне ковзне середнє). За допомогою аналізу особливостей дослідники досягли вражаючої точності, що трохи перевищує 80%, у прогнозуванні ознак значних змін у стані здоров'я. Крім того, моделі лінійної та логістичної регресії продемонстрували досить точні результати в цьому контексті.

У роботі [4] фокус змістився на прогнозування результатів матчів шляхом включення таких факторів, як досвід за хвилину (xpm), золото за хвилину (gpm),

вбивства за хвилину (kpm), ефективність на доріжці та одиночний змагальний ранг. Автори провели порівняння між машинами опорних векторів (SVM) та нейронними мережами (NN), щоб виявити ефективність цих моделей у контексті прогнозування матчів Dota 2.

Дослідження, викладене в [3], заглибилося в різні аспекти, включаючи зміни зон, розподіл членів команди та методи часових рядів. Автори дійшли висновку, що поведінка команд MOBA (Multiplayer Online Battle Arena) свідчить про командні навички, проливаючи світло на кореляцію між стратегічними елементами та результативністю команди.

Дослідники з Університету HSE, як детально описано в [9], вивчали статистику ігор та True Skills, подібну до рейтингу Ело в шахах. Їхній комплексний аналіз поширювався як на CSGO, так і на Dota 2, використовуючи такі метрики, як AUC, відкликання та значення точності для порівняння результатів.

Wang, W., в роботі, висвітленій в [13], зосередився на прогнозуванні результатів матчів на основі драфтів героїв, використовуючи нейронні мережі на графічному процесорі та методи логістичної регресії. Цей підхід демонструє потенціал передових обчислювальних методів у покращенні прогнозування результатів матчів.

Спільні зусилля Іфана Яна, Тянь Циня та Ю-Хенга Лея, описані в [14], були зосереджені на прогнозуванні результатів матчів за допомогою логістичної регресії з використанням даних повторних матчів. Їхні ітеративні зусилля призвели до помітного покращення точності, яка зросла з 58,5% до вражаючих 71,5%.

Ці різноманітні дослідження вносять свій внесок у розвиток аналітики Dota 2, охоплюючи широкий спектр методологій і моделей прогнозування. Кожне дослідження надає унікальну інформацію, що в сукупності поглиблює наше розуміння багатогранної динаміки в сфері конкурентних ігор і моделей прогнозування. Кожне дослідження надає унікальну інформацію, що в сукупності поглиблює наше розуміння багатогранної динаміки в сфері конкурентних ігор.

1.3 Огляд існуючих програмних рішень

Після ознайомлення із статтями та дослідженнями у сфері передбачення результатів матчу у Dota 2 за допомогою нейромереж. Почався пошук готового рішення. Після ретельного пошуку засобу для допомоги вибору скілів у реальному часі прямих конкурентів не було знайдено. Але існує сайт, який збирає статистику всіх матчів Ability Draft, які знаходяться у відкритому доступі. Цей сайт має на меті стати інструментом для покращення та оцінки ігрового процесу в Ability Draft: вимірювання сили героїв, керівництва у виборі здібностей, а також відстеження ваших результатів після гри. Основними функціями якого є відображення статистика здібностей, героїв та синергій. Також, реалізовано списки лідерів для пошуку найкращих гравців.

Ability Statistics (7.35, 92388 games)

Search

Ability	Pick %	Win %	Avg Pick #	Value
Arctic Burn	99.97%	61.41%	3.16	92.73%
Reincarnation	99.99%	61.38%	7.04	97.62%
Heartstopper Aura	100.00%	58.78%	2.81	65.98%
Dispersion	99.98%	57.59%	6.43	58.86%
Aftershock	98.61%	57.54%	7.37	59.62%
Corrosive Skin	99.96%	56.51%	11.80	55.22%
Borrowed Time	99.87%	56.40%	11.26	53.40%
Shukuchi	100.00%	56.30%	2.83	41.17%
Thirst	100.00%	56.01%	2.47	37.85%
Vengeance Aura	99.97%	55.81%	5.57	39.96%
Glaives of Wisdom	99.99%	55.67%	3.17	35.35%
Reflection	99.08%	55.34%	16.17	49.25%

Рис. 1.2. Відображення відсотків перемоги скілів

На рис.1.2 можемо побачити таблицю що містить в собі дані про назву здібності, процент вибору(чи залишилася здібність після завершення драфту),процент перемоги, середній номер вибору, і значення загальної вартості. Всі ці дані можуть бути використання при побудові потрібної нам моделі, тобто цей сайт можу стати основним постачальником даних.

Hero Statistics

Search

Hero	7.34d	Δ	7.35
Nature's Prophet	57.60%	-0.44%	57.16%
Gyrocopter	57.45%	-0.51%	56.94%
Leshrac	55.56%	0.86%	56.42%
Techies	55.57%	0.85%	56.42%
Outworld Devourer	55.25%	0.39%	55.64%
Centaur Warrunner	54.19%	1.23%	55.43%
Shadow Fiend	54.38%	0.82%	55.20%
Hoodwink	55.46%	-0.31%	55.15%
Bane	54.65%	0.48%	55.13%
Snapfire	54.31%	0.83%	55.13%

Рис. 1.3. Відображення відсотків перемоги героїв

На рис.1.2 зображена таблиця героїв та їх вінрейт в залежності від поточного патчу, і різницю між двома балансними правками, що відображає потужність зміни характеристики персонажів.

Ability Pair Statistics (1500 most frequent pairs in 7.35)

Search

Ability One			Ability Two			Combined		
Picture	Name One	Win %	Picture	Name One	Win %	Sample size	Win %	Synergy Δ
	Doppelganger	53.52%		Spirits	54.92%	193	78.24%	24.02%
	Aftershock	57.55%		Proximity Mines	51.42%	329	77.20%	22.81%
	Mirror Image	52.28%		Spirits	54.92%	193	74.09%	20.51%
	Sand Storm	53.74%		Mystic Flare	48.51%	223	70.85%	19.79%
	Heartstopper Aura	58.71%		Infest	49.15%	250	72.40%	18.69%
	Vengeance Aura	55.83%		Chaotic Offering	54.12%	210	73.33%	18.36%
	Aftershock	57.55%		Ball Lightning	52.74%	282	73.40%	18.31%

Рис. 1.4. Сторінка з можливими синергіями

Синергія - це гармонійна взаємодія між обраними здібностями, що створює комплексний ефект, який перевищує суму окремих компонентів. Створення синергетичного героя є тонким мистецтвом, що вимагає від гравців не тільки зосередитися на силі окремих умінь, але й на тому, як вони доповнюють і підсилюють один одного в запалі бою. Успішна синергія може посилити сильні сторони героя, пом'якшити його слабкі сторони та розблокувати унікальний стиль гри, який застане супротивників зненацька. Коли гравці переходять до фази драфту, уважне вивчення синергії додає додатковий стратегічний вимір, спонукаючи до продуманого вибору,

який виходить за рамки безпосередньої сили однієї здатності. Цей акцент на синергії не тільки підвищує складність драфту здібностей, але й винагороджує гравців більш глибоким розумінням взаємодії героїв, сприяючи стратегічному розмаїттю та несподіваним тактичним інноваціям у гонитві за перемогою. Саме статистику цієї основної механіки зображено на рис. 1.4.

Під час пошуку схожих існуючих програмних рішень не було знайдено моделі яка буде працювати для режиму Ability Draft, але є моделі які підохоть для простих дота матчів які оцінують драфти та інші можливі фактори для передбачування результатів, одне з таки досліджень це “Player compability and win prediction in Dota 2 using Graph Neural Networks”[15]. Це дослідження має на меті заповнити цю прогалину, використовуючи графові нейронні мережі (ГНМ) для комплексного моделювання взаємодій. ГНМ, запропоновані Гілмером та ін. (2017) [16], забезпечують потужну основу для поширення інформації через графи та генерування вбудовань.

Мета цього дослідження двояка: по-перше, згенерувати вбудовування для окремих гравців, що полегшить ідентифікацію гравців на заміну зі схожими стилями гри, а по-друге, спрогнозувати ймовірність перемоги на основі початкових даних драфту, що включають гравців та героїв. Вибір GNN узгоджується з їхньою доведеною ефективністю у представленні складних взаємозв'язків у графових структурах, що робить їх ідеальним кандидатом для моделювання взаємодії гравця з героєм та гравця з гравцем, яка присутня у матчах Dota 2.

Створення вбудовувань для гравців передбачає вилучення значущих представлень із взаємопов'язаного графа, зображеного на Рисунку 1.5. Ця підмножина графа складається з вершин, що представляють гравців та героїв, з'єднаних між собою зв'язками гравець-герой. Для полегшення генерації вкладок за допомогою випадкових блукань припускається, що зв'язки між гравцями та героями є неорієнтованими.

Кожен вузол гравця і героя в цьому графі асоціюється з певним набором характеристик, що включає такі атрибути, як ранг, перемоги і співвідношення вбитих до померлих для гравців, а також сила, спритність, інтелект і атака для героїв. Ребра від

гравця до героя інкапсулюють статистичний зв'язок між гравцем і конкретним героєм. Це завдання стає завданням навчання представлення в області гетерогенних мереж.

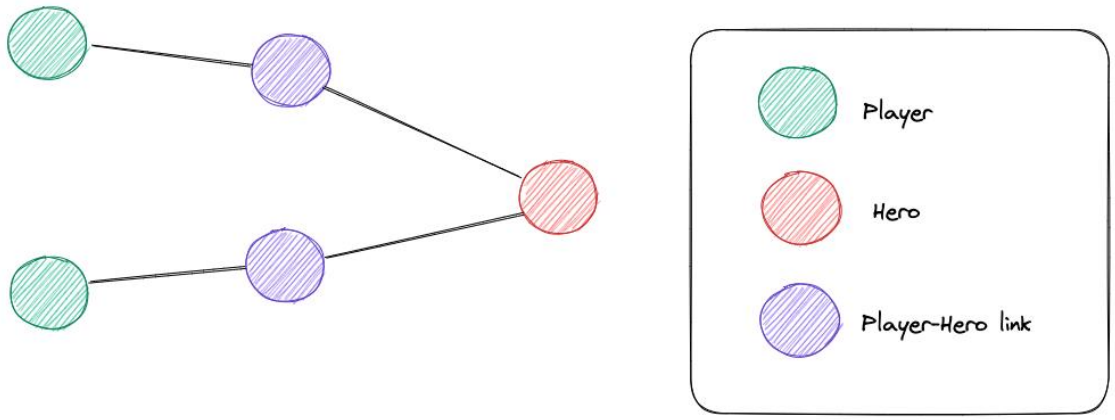


Рис. 1.5. Вилучення значущих представлень із взаємопов'язаного графа

Для вбудовування вузлів гравців використовується модель metapath2vec. Враховуючи, що ребра в нашому графі також мають властивості, їх можна розглядати як вузли з властивостями. Налаштування графа є фундаментальним для цього процесу.

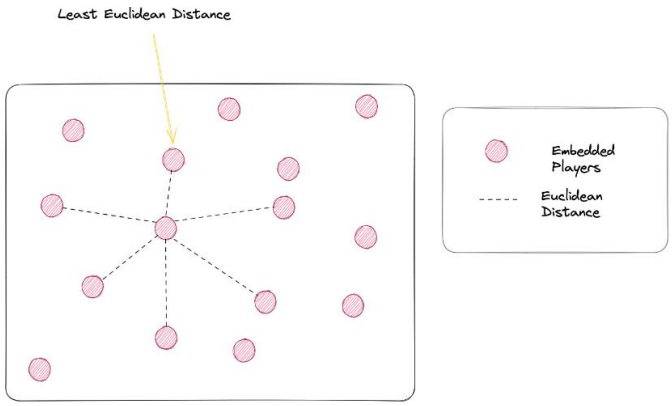


Рис. 1.6. Вилучення значущих представлень із взаємопов'язаного графа

Меташлях ["гравець", "гравець-герой-зв'язок", "герой", "гравець-герой-зв'язок", "гравець"] слугує планом для створення вбудовувань. Ця метафора, по суті, окреслює подорож крізь стосунки "гравець-герой", інкапсулюючи суть взаємопов'язаних

характеристик. Отримані в результаті включення характеризуються низькою розмірністю і латентними атрибутами для кожного гравця, що відповідає основній меті цього етапу.

Ці вбудовування, подібно до відбитків пальців, фіксують особливий стиль гри та статистику кожного гравця. Отже, визначення заміни для даного гравця передбачає просту, але потужну стратегію - пошук гравця з найменшою евклідовою відстанню. Цей спрощений підхід сприяє швидкому визначенню підходящих замін на основі подібності, закладених у латентних репрезентаціях гравців.

Це дослідження розгортається на тлі стрімкого розвитку ігрової індустрії, де штучний інтелект і машинне навчання продовжують переосмислювати досвід гравців і процес прийняття стратегічних рішень. Використовуючи GNN для аналізу складної взаємодії суб'єктів у матчах Dota 2, це дослідження прагне зробити внесок не лише в аналітику кіберспорту, а й у ширшу сферу застосування штучного інтелекту в іграх. У наступних розділах будуть розглянуті методологічні деталі, емпіричні аналізи та висновки, отримані в результаті цього дослідження зв'язку між ШІ, Dota 2 та змагальними іграми.

1.4 Висновки до розділу

В даному розділі було проведено аналіз комп'ютерної гри Dota 2, як моделі для створення дослідження. Ми заглибилися у всебічне дослідження, яке охоплює діапазон від фундаментальної механіки гри до хитросплетінь режиму "Ability Draft". Завдяки ретельному вивченню суміжних робіт ми з'ясували ландшафт існуючих досліджень і технологічних зусиль, заклавши основу для нашого внеску в цю галузь.

Також, в розділі розглянуті існуючі програмні рішення, що дає змогу провести ретельний аналіз технологічного ландшафту, який оточує оптимізацію профілю гравців Dota 2. Зокрема, наш аналіз виявив явну відсутність подібних програмних рішень, що свідчить про новизну та відмінність нашого підходу.

Оскільки ми перебуваємо на перетині ігор, машинного навчання та комп'ютерного зору, синтез цих сфер розгортається, як цікавий напрям для дослідження. Відсутність ідентичних програмних рішень в огляді підкреслює інноваційний характер нашої роботи і позиціонує її як новітню в області оптимізації профілю в Dota 2 в режимі Ability Draft.

РОЗДІЛ 2
ДОСЛІДЖЕННЯ АЛГОРИТМІВ МАШИННОГО НАВЧАННЯ ДЛЯ ПОСТАВЛЕНОГО ЗАВДАННЯ

2.1 Дослідження Random Forest

Для того, щоб зрозуміти важливість Random Forest, необхідна ретельна деконструкція його складових елементів. Дерева рішень, що слугують елементарними будівельними блоками, втілюють ядро алгоритму. Ці дерева, створені за допомогою алгоритму дерева класифікації та регресії (CART), функціонують як базові одиниці ансамблю, з притаманним їм пошуком оптимального розбиття даних, що регулюється такими метриками, як домішка Джині, інформаційний виграш або середньоквадратична помилка.

Хоча дерева рішень уособлюють фундаментальні сутності керованого навчання, їхня схильність до упереджень і надмірного пристосування вимагає ретельного вивчення. Трансформаційний потенціал випадкового лісу полягає в його здатності зменшувати ці вразливості за допомогою оркестрування некорельованих дерев рішень в ансамблі.

Теоретичне підґрунтя, закладене методологіями ансамблевого навчання, зокрема, пакування та бустінгу, полегшує цілісне розуміння алгоритму випадкового лісу. Пакування передбачає автономне навчання моделей на рандомізованих підмножинах даних, ефективно зменшуючи дисперсію в наборах даних, насичених шумом. Внутрішні оцінки відстежують похибку, міцність і кореляцію, і вони використовуються, щоб показати реакцію на збільшення кількості ознак, що використовуються в розділенні. Ці ідеї були вперше описані Лео Брейманом у 1996 році в його роботі “Random Forests”[20]. Вони стали основою для подальшого вивчення та покращення алгоритму. Хоча від першого згадування пройшло немало часу даний алгоритм залишається актуальним і в наш час.

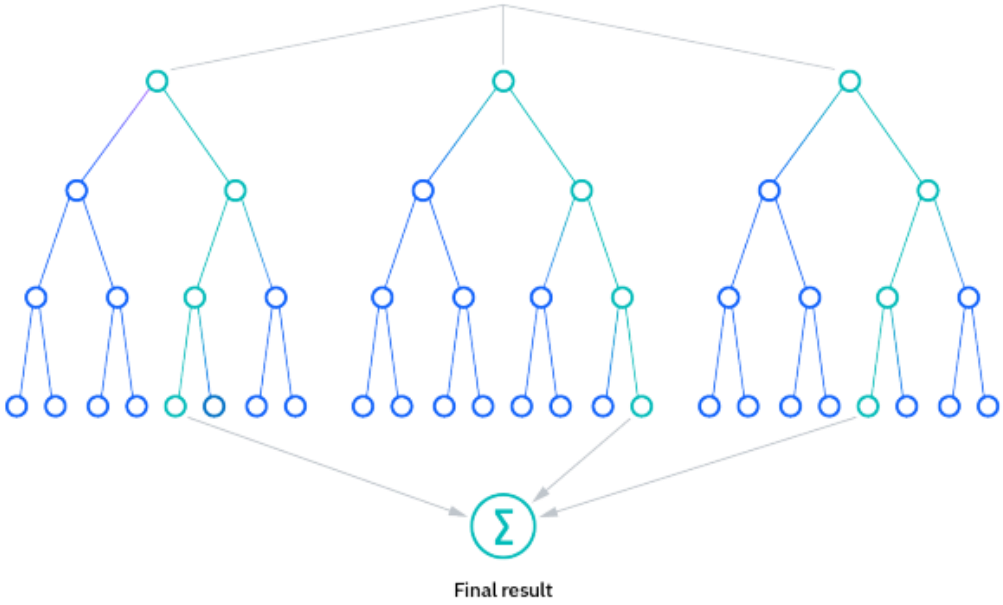


Рис. 2.1. Схематична модель Random Forest

Алгоритм Random Forest виходить за рамки зачатків пакування, асимілюючи як пакування, так і випадковість ознак. Останнє, що позначається як "пакування ознак" або "метод випадкового підпростору", вводить додатковий рівень складності, обмежуючи розгляд ознак випадковою підмножиною на кожному кроці. Це навмисне відхилення від традиційного підходу до дерева рішень, в якому ретельно розглядаються всі ознаки, слугує для пом'якшення кореляцій між деревами, зображення можливої моделі на рисунку 2.1, тим самим підвищуючи стійкість моделі.

Врахування варіативності даних, аналогічно до розбіжностей у парадигмах опитування, є відмінною рисою методу випадкових лісів. Цей методологічний стрижень дає змогу розробити стратегію зменшення ризиків, пов'язаних із надмірним припасуванням, упередженістю та загальною дисперсією, що призводить до підвищення точності прогнозування.

Операціоналізація випадкового лісу вимагає ретельної конфігурації гіперпараметрів - розміру вузла, кількості дерев та кількості ознак, що відбираються. Тонкий вплив цих параметрів на ефективність алгоритму у вирішенні задач регресії та класифікації підкреслює кількісну складність алгоритму.

Охоплюючи репертуар дерев рішень, кожне з яких тренується на бутстреп-вибірці, оркестр алгоритму поширюється на вибірку поза пакетом, вносячи додатковий шар випадковості. Об'єднання прогнозів окремих дерев, що досягається шляхом усереднення в задачах регресії або голосуванням більшості в класифікації, завершується прогнозом, остаточно підтвердженим перехресною перевіркою з використанням вибірки out of bag.

Ансамблева архітектура зменшує ризик надмірної підгонки, обґрунтовуючи стійкість моделі через усереднення некорельованих дерев .Наділений здатністю вправно орієнтуватися як в задачах регресії, так і в задачах класифікації, Random Forest стає кращим інструментом для науковців, що працюють з даними. Ефективність мішків ознак поширюється на точні оцінки навіть за наявності відсутніх даних. Алгоритм полегшує оцінку важливості змінних, надаючи уявлення про внесок кожної ознаки за допомогою таких метрик, як важливість Джині та середнє зменшення домішок.

Обчислювальна потужність, яку демонструють алгоритми випадкового лісу в обробці об'ємних наборів даних, супроводжується значними витратами часу на обробку. Здатність алгоритму працювати з великими наборами даних вимагає значних обчислювальних ресурсів для зберігання та обробки. Розшифровка прогнозів, що виходять з безлічі дерев рішень у випадковому лісі, створює додатковий рівень складності у порівнянні з інтерпретацією одиночного дерева рішень.

Підсумовуючи, у цьому розділі було представлено всебічне і науково обґрунтоване дослідження алгоритму випадкового лісу. Від його теоретичних основ у вигляді дерев рішень до ретельної організації парадигм ансамблевого навчання та аспектів його переваг і викликів, цей розділ закладає основу для тонкого розуміння емпіричних застосувань Random Forest у наступних розділах цієї роботи.

2.2 Дослідження Gradient Boost

У широкому спектрі машинного навчання Gradient Boosting став витонченим і високоефективним алгоритмом, який долає обмеження окремих моделей шляхом

ітеративної побудови надійної прогнозуючої моделі за допомогою послідовного поєднання слабких моделей, що навчаються. У цьому розділі розглядаються основні концепції, алгоритмічний робочий процес і практичні наслідки градієнтного бустінгу, висвітлюється його значення для підвищення точності прогнозування.

Теоретичні основи градієнтного бустінгу

Ансамблеве навчання, що характеризується об'єднанням декількох слабких учнів для формування надійної моделі прогнозування, слугує концептуальним наріжним каменем Gradient Boosting. На відміну від традиційних ансамблевих методів, які будують моделі незалежно, Gradient Boosting використовує послідовний підхід, ітеративно вдосконалюючи прогностичні можливості моделі. Термін "слабкі учні" відноситься до моделей з точністю прогнозування трохи кращою, ніж випадковість. Бустінг, ключова концепція Gradient Boosting, фокусується на послідовному вдосконаленні моделі шляхом надання більшої ваги екземплярам, які були неправильно класифіковані на попередній ітерації. Цей ітеративний процес підвищує продуктивність моделі, роблячи її здатною вловлювати складні закономірності в даних.

Градієнтне підсилення мінімізує функцію втрат, спускаючись по градієнту втрат відносно прогнозів моделі. Цей процес ітеративно вдосконалює модель, зменшуючи помилки та покращуючи загальну продуктивність. Градієнт визначає напрямок і величину найкрутішого зростання функції втрат, спрямовуючи алгоритм до оптимального рішення. Градієнтне підсилення включає методи регуляризації, щоб запобігти надмірному пристосуванню. Параметри регуляризації, такі як глибина дерев і мінімальна кількість вибірок, необхідних для розбиття вузла, допомагають контролювати складність ансамблю і покращують його здатність узагальнювати невидимі дані.

Основою для цих ідей є дослідження "Stochastic Gradient Boosting"[20] автора Jerome H. Friedman в якому він зробив висновок, що метод може бути застосований до відповідних функцій витрат як алгоритм оптимізації. Метод зазнав подальшого розвитку в досліджені "Boosting Algorithms as Gradient Descent"[21], де основною ідеєю є

оптимізації функцій витрат шляхом ітеративного вибору слабких гіпотез або функції з від'ємним градієнтом.

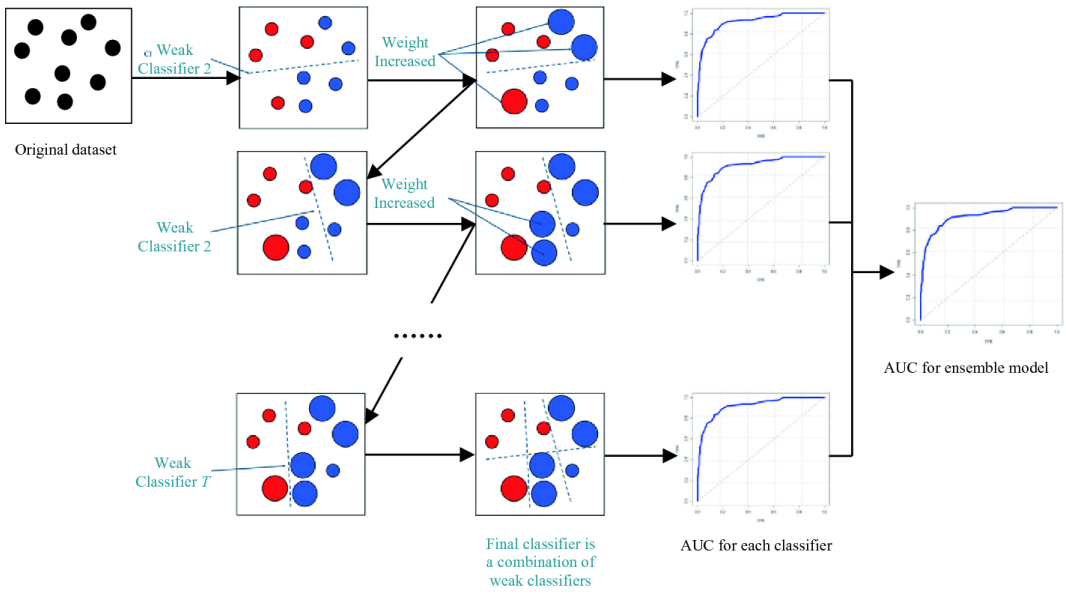


Рис. 2.2. Алгоритмічний робочий процес градієнтного бустінгу

Gradient Boosting починається з ініціалізації моделі за допомогою простого базового учня, як правило, неглибокого дерева рішень. Наступні ітерації спрямовані на виправлення помилок (залишків), зроблених моделлю на попередньому кроці. Кожен новий слабкий учень навчається на залишках, поступово зменшуючи помилки прогнозування і покращуючи загальну точність. Під "градієнтом" в Gradient Boosting мається на увазі використання оптимізації градієнтного спуску для мінімізації функції втрат. Під час кожної ітерації алгоритм обчислює градієнт функції втрат по відношенню до прогнозів моделі, і слабкий учень підбирається таким чином, щоб мінімізувати цей градієнт. Прогнози слабкого учня потім поєднуються з існуючою моделлю, приділяючи більше уваги областям, де модель раніше показувала незадовільні результати.

Практичне значення та інтерпретованість моделі

Gradient Boosting - це універсальний алгоритм, який підходить як для задач регресії, так і для задач класифікації. Його здатність працювати з різними типами даних і фіксувати складні взаємозв'язки між ознаками робить його кращим вибором у різних

галузях, включаючи фінанси, охорону здоров'я та маркетинг. Хоча Gradient Boosting вирізняється високою точністю прогнозування, інтерпретація моделі може бути проблематичною через її ансамблеву природу. Однак ретельний облік гіперпараметрів, таких як швидкість навчання, глибина дерева і кількість ітерацій, дозволяє фахівцям точно налаштувати продуктивність моделі і досягти балансу між точністю і інтерпретованістю.

Переваги та міркування щодо градієнтного бустінгу

Gradient Boosting часто перевершує інші алгоритми за точністю прогнозування, особливо при роботі зі складними наборами даних. Алгоритм дає уявлення про важливість ознак, допомагаючи зрозуміти внесок кожної змінної в прогнози моделі. Однак навчання моделі Gradient Boosting може бути трудомістким, особливо при роботі з великими наборами даних і складними моделями. Без належної регуляризації та налаштування гіперпараметрів моделі Gradient Boosting можуть бути чутливими до перенастроювання.

Отже, Gradient Boosting - це потужний алгоритм у прогнозному моделюванні, що пропонує потужне поєднання точності та універсальності. У цьому розділі було описано теоретичні основи, алгоритмічний робочий процес і практичні міркування щодо градієнтного бустінгу, створивши підґрунтя для його застосування і дослідження в наступних розділах цієї роботи.

2.3 Дослідження Gaussian Naive Bayes

У середовищі машинного навчання гаусівський наївний Байєс (Gaussian Naive Bayes, GNB) виділяється як алгоритм ймовірнісної класифікації, відомий своєю простотою, ефективністю та результативністю, особливо в ситуаціях, коли ознаки мають неперервний розподіл значень. У цьому розділі розглядаються основоположні принципи, припущення про незалежність ознак і практичне застосування GNB у різних

галузях. Теоретичну основу для вивчення наївного Байєса було закладено в роботі “Artificial Intelligence: A Modern Approach”[30].

Теоретичні основи гауссівського наївного Байєса

Гауссівський наївний Байєс ґрунтується на теоремі Байєса, фундаментальній концепції теорії ймовірностей. Він використовує умовну ймовірність для прогнозування на основі спостережуваних даних. "Наївний" в його номенклатурі випливає з припущення про незалежність ознак, що означає, що кожна ознака робить незалежний і рівний внесок у класифікаційне рішення.

Припущення про незалежність ознак

Суть гауссівського наївного Байєса полягає в припущенні, що ознаки в наборі даних є умовно незалежними з огляду на мітку класу. Хоча це припущення може здатися надто спрощеним, воно часто виконується на практиці, і, незважаючи на свою простоту, GNB продемонстрував помітний успіх у різних застосуваннях.

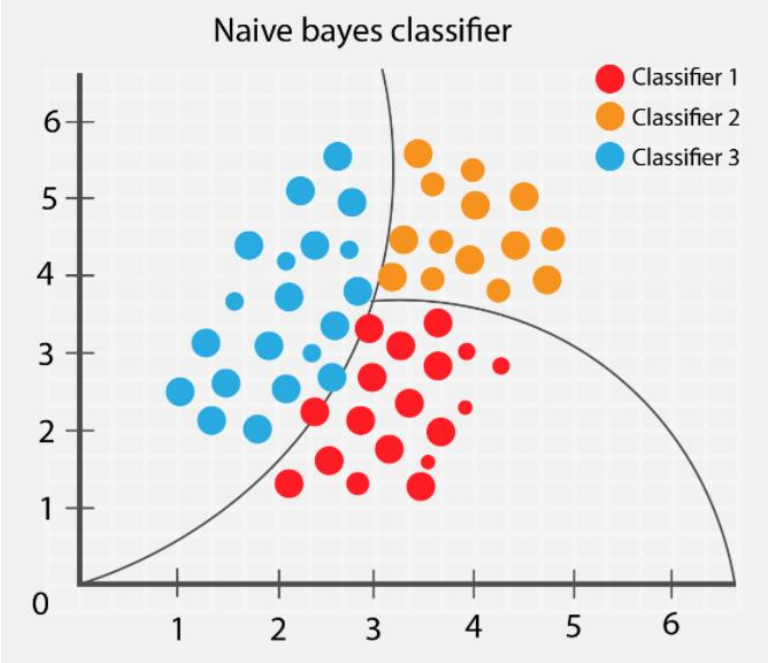


Рис. 2.3. Схематичний результат гауссівського наївного Байєса

Алгоритм роботи гауссівського наївного Байєса

1. Обчислення попередньої ймовірності кожного класу на основі частоти кожного класу в навчальній вибірці.
2. Для кожної комбінації ознаки та класу оцінити середнє та дисперсію розподілу ознаки, використовуючи навчальні дані.
3. Для кожної нової точки даних обчисліть ймовірність того, що вона належить до кожного класу, використовуючи функцію щільності розподілу ймовірності Гауса.
4. Застосуйте теорему Байєса для обчислення апостеріорної ймовірності кожного класу.
5. Призначити клас з найвищою апостеріорною ймовірністю як передбачуваний клас для даної точки даних.

Практичне застосування гауссового наївного Байєса

Гауссівський наївний Байєс знаходить застосування в різних сценаріях реального світу:

- класифікація тексту - GNB широко використовується в задачах обробки природної мови, таких як фільтрація спаму та аналіз настроїв, де розподіл частот слів часто відповідає гаусівському розподілу.
- медична діагностика - в охороні здоров'я GNB використовується для задач медичної діагностики, де такі характеристики, як результати тестів або атрибути пацієнта, мають неперервний розподіл.
- виявлення шахрайства - GNB підходить для виявлення шахрайства у фінансових транзакціях, оскільки він може ефективно моделювати розподіл ознак законних і шахрайських транзакцій.
- рекомендаційні системи - GNB може бути корисним для систем рекомендацій, особливо коли йдеться про постійні вподобання або рейтинги користувачів.

Переваги та міркування щодо гауссівського наївного Байєса

Переваги:

- Обчислювальна ефективність. GNB є обчислювально ефективним, що робить його добре придатним для великих наборів даних.
- Простота та інтерпретація. Простота алгоритму полегшує інтерпретацію, що сприяє розумінню та поясненню моделі.

Міркування:

- Припущення про незалежність ознак. Хоча припущення про незалежність ознак спрощує модель, воно може не виконуватися у всіх випадках, що потенційно впливає на продуктивність.
- Обмежена виразність. GNB припускає гаусівський розподіл для ознак, який може не відображати складні взаємозв'язки в даних.

Гауссівський наївний Байєс, спираючись на байєсівські принципи та припущення про незалежність ознак, пропонує елегантне та ефективне рішення для задач ймовірнісної класифікації. У цьому розділі ми детально розглянули теоретичні основи, алгоритм роботи та практичне застосування гауссівського наївного Байєса, підтвердивши його актуальність та корисність у різноманітних контекстах машинного навчання.

2.4 Дослідження KNN

K-Nearest Neighbors (KNN) - це універсальний та інтуїтивно зрозумілий алгоритм, який широко використовується в машинному навчанні як для класифікації, так і для регресії. Він працює за принципом близькості, роблячи прогнози на основі класу більшості або середнього значення k-найближчих точок даних у просторі ознак. У цьому розділі ми розглянемо основні принципи, алгоритм роботи та практичні міркування КНМ, проливаючи світло на його простоту, адаптивність та застосування в різних галузях.

Основи для цього алгоритму були закладення в дослідженні “Discriminatory Analysis. Nonparametric Discrimination: Consistency Properties” [23], а потім покращенні та виведені в окремий алгоритм в роботі “Nearest neighbor pattern classification” [24].

Теоретичні основи К-найближчих сусідів

В основі KNN лежить принцип близькості, що відрізняє його від інших алгоритмів. Прогнози базуються на класі більшості або середньому значенні k-найближчих точок даних у просторі ознак. Цей розділ заглиблюється в теоретичні основи, які роблять KNN особливим і потужним підходом у розпізнаванні образів.

Алгоритмічний процес роботи К-найближчих сусідів

- Представлення даних: кожна точка даних у наборі даних представляється як точка в n-вимірному просторі ознак, що забезпечує основу для аналізу на основі близькості.
- Обчислення відстані: вибір метрики відстані, наприклад, евклідової або манхеттенської, має вирішальне значення, оскільки KNN обчислює відстань від заданої точки даних до кожної іншої точки в наборі даних.
- Вибір сусідів: на основі розрахованих відстаней визначаються k найближчих сусідів, які формують опорні точки для прогнозування.

Для задач класифікації, клас більшості серед k найближчих сусідів визначає прогнозований клас. Для задач регресії прогноз є середнім значенням цільових значень k-найближчих сусідів.

Практичне застосування методу k-найближчих сусідів

KNN знаходить різноманітне застосування в різних галузях, демонструючи свою адаптивність та ефективність.

- Розпізнавання зображень:

  У комп'ютерному зорі KNN використовується для класифікації зображень на основі схожості їхніх характеристик.
- Медична діагностика:

KNN допомагає в охороні здоров'я для класифікації хвороб і діагностики пацієнтів, враховуючи схожість атрибутів пацієнтів.

- Рекомендаційні системи:

  KNN допомагає системам рекомендацій, визначаючи користувачів зі схожими вподобаннями та пропонуючи товари, які подобаються їхнім сусідам.

- Виявлення аномалій:

  У кібербезпеці KNN допомагає виявляти аномальні патерни, порівнюючи поведінку мережевих об'єктів з їхніми найближчими сусідами.

Переваги та міркування щодо K-найближчих сусідів

Переваги K-Nearest Neighbours можна віднести:

- KNN легко зрозуміти і реалізувати, що робить його доступним як для початківців, так і для експертів у цій галузі. Його інтуїтивний підхід, заснований на близькості, відповідає здоровому глузду.
- на відміну від багатьох алгоритмів машинного навчання, які вимагають фази навчання, KNN базується на прикладах і не передбачає явного процесу навчання. Під час прогнозування модель узагальнює весь набір даних.
- KNN добре працює з різноманітними розподілами даних. Він є непараметричним і може адаптуватися до різних форм і структур у просторі ознак, що робить його універсальним у різних областях.
- KNN можна без проблем застосовувати як для класифікації, так і для регресійних задач. Його гнучкість дозволяє вирішувати широкий спектр сценаріїв предиктивного моделювання.
- KNN робить мінімальні припущення про розподіл вихідних даних. Це робить його придатним для наборів даних зі складними закономірностями або нелінійними зв'язками.

Міркування щодо K-найближчих сусідів:

- основні обчислювальні витрати в KNN пов'язані з обчисленням відстаней між точкою запиту і всіма іншими точками в наборі даних. Зі збільшенням набору даних зростає часова складність алгоритму, що впливає на його ефективність.
- KNN чутливий до зашумлених даних та пропусків. Пропуски або нерелевантні ознаки можуть непропорційно впливати на прогнози, що призводить до потенційних неточностей.
- характеристики, які не мають відношення до завдання прогнозування, можуть негативно вплинути на продуктивність ШНМ. Важливо попередньо обробити дані і вибрати релевантні ознаки, щоб забезпечити значущі прогнози.

Оскільки для прогнозування KNN покладається на зберігання всього набору даних у пам'яті, він може займати багато пам'яті, особливо для великих наборів даних. Це міркування стає вирішальним у сценаріях, де ресурси пам'яті обмежені.

Оптимальний вибір K

Вибір параметра "k" (кількість сусідів) є критично важливим і може суттєво вплинути на продуктивність моделі. Вибір невідповідного значення "k" може призвести до недостатнього або надмірного припасування, що підкреслює необхідність ретельного налаштування.

Оптимальний вибір "k" часто залежить від характеристик набору даних і закономірностей, що лежать в його основі. Методи перехресної перевірки, коли набір даних багаторазово розбивається на навчальний і перевірочний набори, можуть допомогти у визначенні найбільш підходящого "k" для конкретної проблеми. Експерименти з різними значеннями "k" та оцінка їх впливу на продуктивність моделі є важливим кроком у налаштуванні алгоритму ШНМ для отримання оптимальних результатів у різноманітних додатках. Також, потрібно розуміти коли обраний ближній сусід дійсно має певне значення, про це детально йдеться в роботі "When is "nearest neighbor" meaningful?"[25].

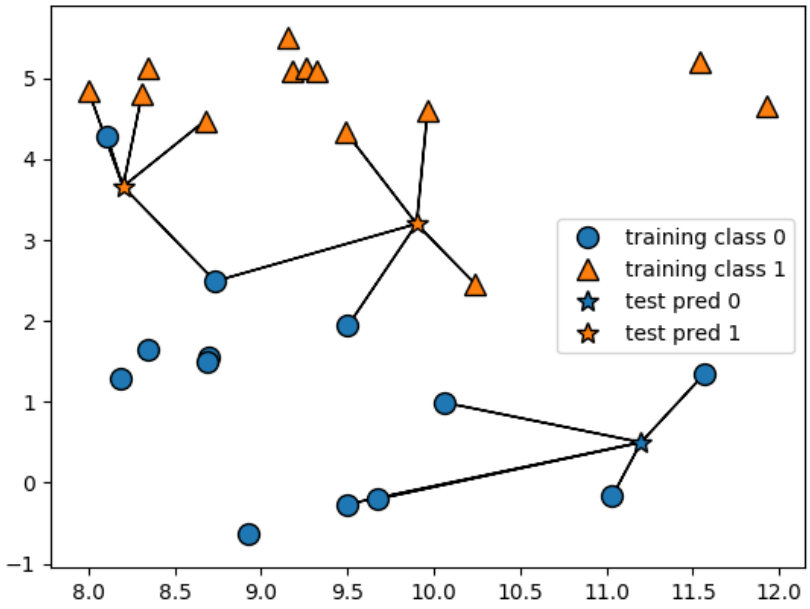


Рис. 2.4. Графік результату моделі KNN

Розуміння цих переваг і нюансів дозволяє фахівцям приймати обґрунтовані рішення при виборі ШНМ для конкретного завдання. Хоча в певних сценаріях KNN демонструє відмінні результати, важливо визнати його обмеження і розглянути альтернативні алгоритми, засновані на специфічних характеристиках набору даних і характеру завдання прогнозування.

На закінчення, K-найближчих сусідів, з його опорою на близькість і простоту, є надійним інструментом як для класифікації, так і для регресійних задач. У цьому розділі ми розглянули теоретичні основи, алгоритмічний робочий процес і практичні застосування KNN, продемонструвавши його універсальність і актуальність у різних контекстах машинного навчання.

2.5 Дослідження Logistic Regression

Логістична регресія є фундаментальним і широко використовуваним статистичним методом у машинному навчанні, ефективно поєднуючи сфери регресії та класифікації. У цьому розділі розглядаються теоретичні основи, алгоритм роботи та

практичне застосування логістичної регресії, демонструючи її ключову роль у прогнозному моделюванні.

Теоретичні основи логістичної регресії

Логістична регресія розширює принципи лінійної регресії для прогнозування ймовірності бінарного результату. Вона моделює логарифм ймовірності, використовуючи сигмоїдну функцію для перетворення результату в діапазон від 0 до 1. Оцінка параметрів здійснюється на основі методу максимальної правдоподібності з метою максимізації ймовірності спостережуваного результату відповідно до припущеного логістичного розподілу.

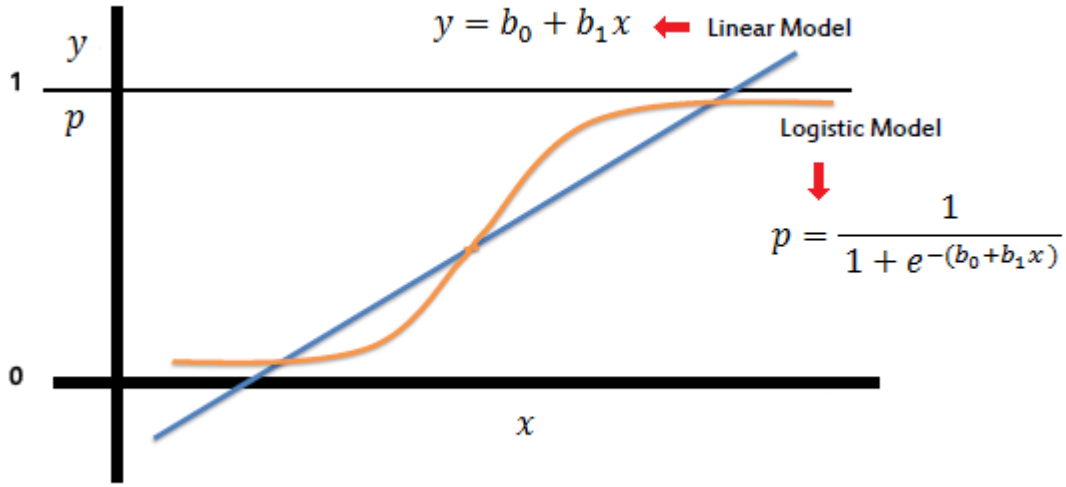


Рис. 2.5. Порівняння логістичної регресії та лінійної моделі

Алгоритм роботи логістичної регресії

Для задач бінарної класифікації логістична регресія потребує бінарної залежної змінної. Масштабування ознак часто виконується для забезпечення пропорційних внесків від усіх ознак. Навчання моделі передбачає оптимізацію параметрів за допомогою ітераційних алгоритмів, таких як градієнтний спуск. Сигмоїдна функція

встановлює межу рішення, гіперплощину, яка розділяє простір ознак на області, що відповідають різним класам.

Практичне застосування логістичної регресії

Логістична регресія знаходить широке застосування в задачах бінарної та багато класової класифікації, починаючи від виявлення спаму і медичної діагностики до моделювання ризиків у фінансах. Її вміння надавати оцінки ймовірності робить її цінною в сценаріях, де розуміння ймовірності події має вирішальне значення.

Переваги та міркування щодо логістичної регресії

Переваги:

- зрозумілість - коефіцієнти логістичної регресії можна інтерпретувати, що полегшує оцінку впливу кожної ознаки.
- ефективність - модель є обчислювально ефективною, здатною обробляти великі масиви даних.
- імовірнісний результат - логістична регресія надає ймовірнісні результати, що дозволяє приймати рішення з урахуванням нюансів.

Міркування:

- Припущення про лінійність - логістична регресія припускає лінійну залежність між ознаками та лог-шансами результату.
- Чутливість до викидів - викиди можуть непропорційно впливати на параметри, що вимагає надійних методів або кроків попередньої обробки.
- Припущення про незалежність ознак - логістична регресія припускає незалежність між ознаками, що впливає на стабільність та інтерпретованість.
- Обмежена виразність - модель може мати труднощі з відображенням складних взаємозв'язків у дуже нелінійних наборах даних порівняно з більш складними моделями.

Отже, логістична регресія відіграє ключову роль у прогнозному моделюванні, пропонуючи збалансоване поєднання інтерпретованості та ефективності. У цьому

розділі ми розглянули її теоретичні основи, алгоритмічний робочий процес і практичні застосування, підкресливши її актуальність у різних сферах і визнавши міркування щодо ефективного використання.

2.6 Дослідження LSTM

У сфері штучного інтелекту мережа довготривалої короткочасної пам'яті (Long Short-Term Memory, LSTM) є грізною архітектурою, особливо вправною у вирішенні проблем, пов'язаних з послідовними даними. У цьому розділі ми спробуємо розібратися в тонкощах LSTM, дослідити її фундаментальні принципи, архітектурні нюанси та трансформаційні застосування.

Розуміння LSTM

LSTM, абревіатура від Long Short-Term Memory (довга короткочасна пам'ять), являє собою спеціалізовану форму архітектури рекурентних нейронних мереж (RNN). Вона поширена в таких галузях, як обробка природної мови та розпізнавання мови, де моделювання довготривалих залежностей має вирішальне значення.

На відміну від звичайних RNN з простою структурою, LSTM може похвалитися складною структурою з додатковими комірками пам'яті та вентилями. Ці компоненти наділяють його чудовою здатністю вибірково запам'ятовувати або забувати інформацію з попередніх часових кроків, долаючи ключове обмеження традиційних ШНМ.

Компоненти LSTM-комірки

LSTM-комірка складається з декількох основних компонентів:

- вхідні вентелі - регулюють потік інформації з поточного входу та попереднього прихованого стану в комірку пам'яті.
- затвор забуття - керування потоком інформації від попередньої комірки пам'яті до поточної, що дозволяє вибірково забувати або запам'ятовувати минулу інформацію.

- комірка пам'яті - внутрішній стан LSTM, що зберігає інформацію, яка підлягає вибірковій модифікації за допомогою входів і вентилів забування.
- вихідні вентилі - керування потоком інформації з комірки пам'яті до поточного прихованого стану та виходу.

Архітектуру LSTM можна уявити як серію повторюваних блоків або комірок, кожна з яких містить взаємопов'язані вузли. Високорівневий огляд включає в себе:

1. Вхід. На кожному часовому кроці LSTM обробляє вхідний вектор (x_t), що представляє поточне спостереження або маркер у послідовності.
2. Прихований стан. Підтримка вектора прихованого стану (h_t), що позначає поточну "пам'ять" мережі. Ініціалізується нулями на початку послідовності.
3. Стан комірки. Керування вектором стану комірки (c_t), що відповідає за зберігання довгострокової інформації протягом послідовності. Ініціалізується нулями на початку послідовності.

Ці частини зображенні на рис. 2.6.

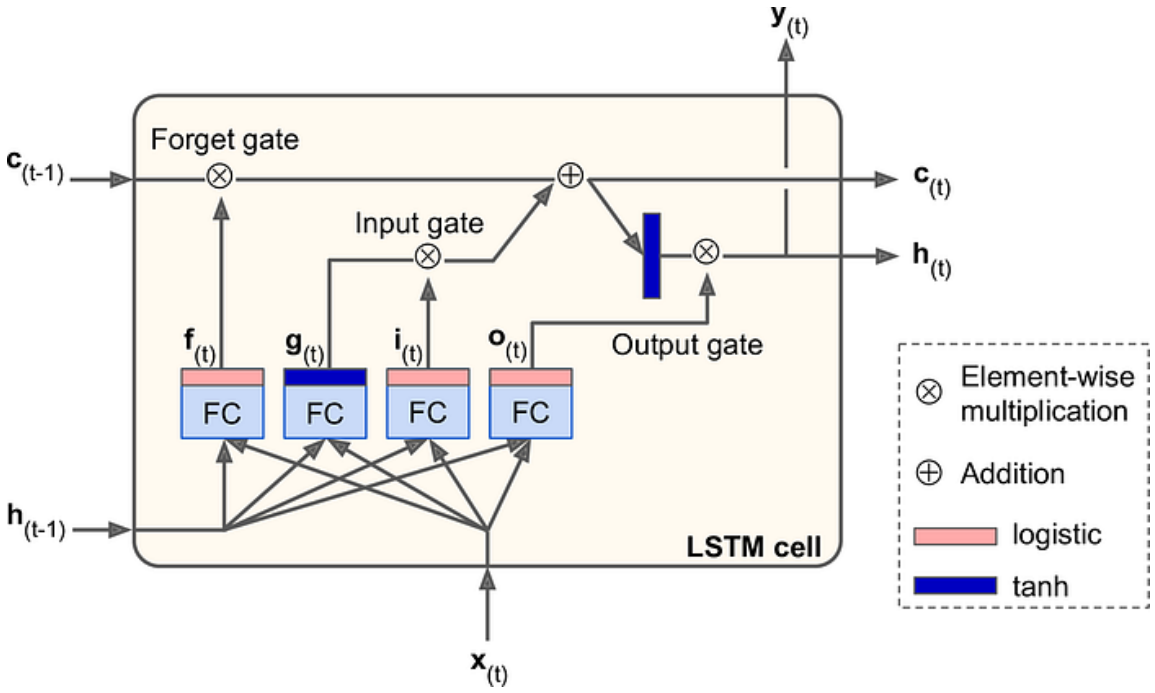


Рис. 2.6. Архітектура LSTM

Вентелі в LSTM

LSTM включають три ключові вентилі для регулювання потоку інформації:

- вентиль забуття - визначає, яку частину попереднього стану комірки забути, а яку запам'ятати, вибірково стираючи або зберігаючи інформацію з попереднього часового кроку.
- вхідний вентиль - диктує, яку частину поточного входу додати до стану комірки, вибірково включаючи або відкидаючи нову інформацію.
- вихідний вентиль - визначає, яку частину поточного стану комірки виводити як поточний прихований стан (h_t), дозволяючи вибірково фокусуватись або ігнорувати певні аспекти при обчисленні вихідних даних.

На кожному часовому кроці LSTM виробляє вихідний вектор (y_t), що інкапсулює передбачення мережі або кодування поточного входу.

Алгоритм роботи LSTM

Ключовим моментом в LSTM є стан комірки - горизонтальна лінія, що проходить через верхню частину діаграми.

Стан комірки схожий на конвеєрну стрічку. Вона проходить прямо по всьому ланцюжку, лише з деякими незначними лінійними взаємодіями. Дуже легко для інформації просто текти вздовж неї без змін.

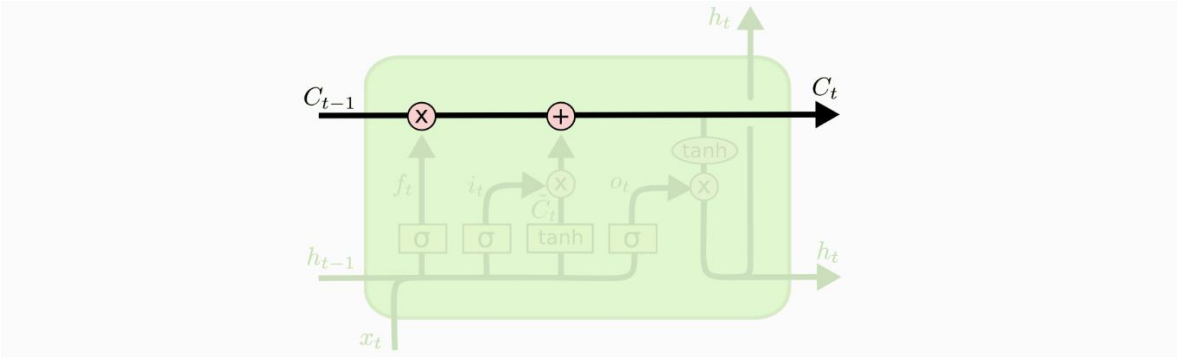


Рис. 2.7. Стан комірки, горизонтальна лінія, що проходить через верхню частину діаграми

LSTM має можливість видаляти або додавати інформацію до стану комірки, що ретельно регулюється структурами, які називаються воріть ми.

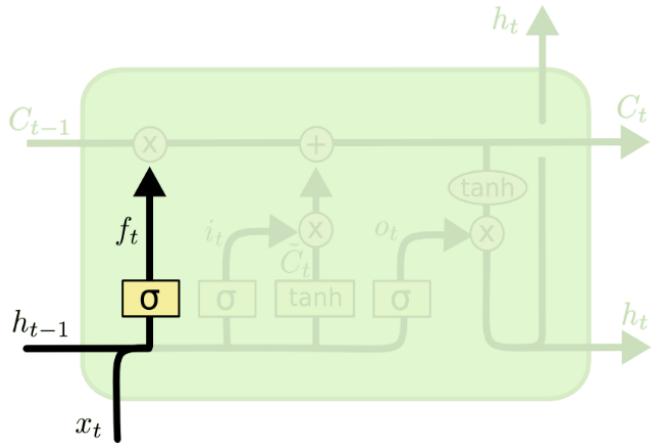
Ворота - це спосіб необов'язкового пропускання інформації. Вони складаються з сигмоїдного шару нейронної мережі та операції точкового множення.

Сигмоїдний шар виводить числа від нуля до одиниці, описуючи, скільки кожного компонента має бути пропущено. Значення нуль означає "нічого не пропускати", тоді як значення одиниця означає "пропустити все!". LSTM має три таких вентилі для захисту та контролю стану комірки.

Покрокова інструкція до LSTM

Перший крок у нашому LSTM - це вирішити, яку інформацію ми збираємося викинути зі стану клітини. Це рішення приймається сигмоїдним шаром, який називається "шар воріт забуття". Він дивиться на ht-1та xti виводить число від 0 та 1 для кожного числа у стані комірки Ct-1. Де 1 означає "повністю зберегти це", а 0 позначає "повністю позбутися цього".

Повернімося до нашого прикладу з мовною моделлю, яка намагається передбачити наступне слово на основі всіх попередніх. У такій задачі стан комірки може включати стать теперішнього об'єкта, щоб можна було використати правильні займенники. Коли ми бачимо новий об'єкт, ми хочемо забути стать старого об'єкта.



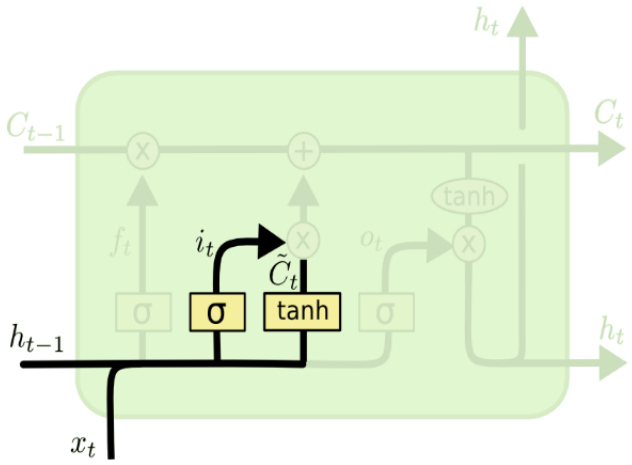
$$f_t = \sigma\left(W_f \cdot [h_{t-1}, x_t] \; + \; b_f\right)$$

Рис. 2.8. Сигмоїдний шар, який називається "шар вхідних воріт"

Наступний крок - вирішити, яку нову інформацію ми збираємося зберігати у стані клітини. Це складається з двох частин. По-перше, сигмоїдний шар, який називається "шар вхідних воріт", вирішує, які значення ми будемо оновлювати. Далі, тангенціальний шар створює вектор нових значень-кандидатів, C~t які можна додати до стану. На наступному кроці ми об'єднаємо ці два шари, щоб створити оновлення стану.

У прикладі нашої мовної моделі ми хочемо додати стать нового суб'єкта до стану комірки, щоб замінити стару, яку ми забули.

Настав час оновити старий стан комірки, Ct-1на новий стан клітинки Ct. На попередніх кроках ми вже вирішили, що робити, нам залишилося лише зробити це. Ми множимо старий стан на ft забуваючи про те, що ми вирішили забути раніше. Потім додаємо до нього *C~t.



$$i_t = \sigma\left(W_i \cdot [h_{t-1}, x_t] \; + \; b_i\right)$$

$$\tilde{C}_t = \tanh(W_C \cdot [h_{t-1}, x_t] \; + \; b_C)$$

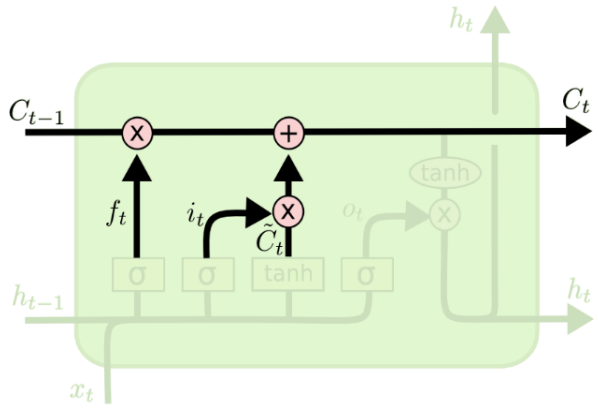
Рис. 2.9. Процес оновлення стану

.

Це нові значення-кандидати, масштабовані на те, наскільки ми вирішили оновити кожне значення стану.

У випадку з мовною моделлю, саме тут ми фактично відкидаємо інформацію про стан старого суб'єкта і додаємо нову інформацію, як ми вирішили на попередніх кроках.

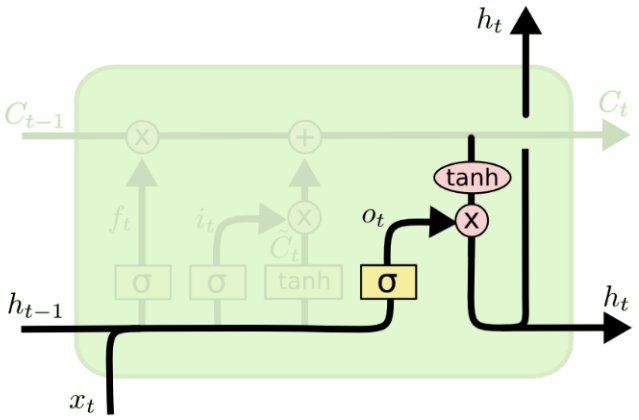
Нарешті, нам потрібно вирішити, що ми будемо виводити. Цей результат базуватиметься на стані нашої комірки, але буде відфільтрованою версією.



$$C_t = f_t * C_{t-1} + i_t * \tilde{C}_t$$

Рис. 2.10. Обрахунок нових кандидатів

Спочатку ми запускаємо сигмоїдний шар, який вирішує, які частини стану клітинки ми будемо виводити. Потім ми пропускаємо стан клітинки через tanh (щоб проштовхнути значення в діапазоні від -1 та 1) і множимо його на вихід сигмоїдного воріт, щоб вивести тільки ті частини, які ми вирішили вивести.



$$o_t = \sigma\left(W_o\left[h_{t-1}, x_t\right] + b_o\right)$$
$$h_t = o_t * \tanh\left(C_t\right)$$

Рис. 2.11. Фінальний крок

У прикладі з мовною моделлю, оскільки вона щойно побачила підмет, вона може захотіти вивести інформацію, пов'язану з дієсловом, на випадок, якщо це те, що буде далі. Наприклад, вона може вивести підмет в однині чи множині, щоб ми знали, в якій формі відмінювати дієслово, якщо це буде наступним.

Переваги та недоліки використання мереж LSTM

До переваг мереж LSTM можна віднести:

- моделювання довгострокових залежностей - LSTM чудово моделюють довготривалі залежності в послідовних даних. Їх унікальна здатність вибірково "запам'ятовувати" або "забувати" інформацію з часом робить їх безцінними для таких завдань, як розпізнавання мови, машинний переклад і генерація тексту.
- стійкість до зашумлених даних - LSTM демонструють більшу стійкість до зашумлених або відсутніх даних порівняно з іншими рекурентними нейронними мережами. Ця стійкість пояснюється їх здатністю вибірково відфільтровувати нерелевантну або зашумлену інформацію завдяки використанню воріт забування (forget gate).
- гнучкість - LSTM можуть похвалитися універсальністю і застосовністю в широкому спектрі завдань, включаючи класифікацію, регресію і навчання від послідовності до послідовності.
- зрозумілість - вектор стану комірки в LSTM, що представляє "пам'ять" мережі на кожному часовому кроці, покращує інтерпретованість. Ця особливість відрізняє їх від інших типів рекурентних нейронних мереж.

До недоліків мереж LSTM можна віднести:

- обчислювальні витрати - навчання та оцінювання LSTM може бути пов'язане з великими обчислювальними витратами, особливо для довгих послідовностей або великих наборів даних.
- схильність до перенавчання - LSTM схильні до перенавчання, особливо при роботі з невеликими наборами даних. Ризик посилюється з надто складними архітектурами моделей.
- налаштування гіперпараметрів - досягнення оптимальної продуктивності LSTM передбачає налаштування багатьох гіперпараметрів, включаючи кількість прихованих одиниць, швидкість навчання та частоту відсіву.

- вимоги до даних - LSTM потребують значного обсягу навчальних даних для ефективного розпізнавання складних патернів. Недостатня кількість даних може поставити під загрозу продуктивність моделі.

Синергетична взаємодія стану комірки, прихованого стану та вентилів дозволяє LSTM вибірково "запам'ятовувати" або "забувати" інформацію з часом. Ця адаптивна здатність робить LSTM добре придатними для задач, що вимагають моделювання довготривалих залежностей та складних послідовностей.

2.7 Висновки до розділу

В даному розділі виконано огляд алгоритмів машинного навчання та визначення найефективнішого алгоритму машинного навчання для оптимізації профілів ігрових персонажів у реальному часі в зрізі теоретичної інформації. Було проведено всебічне дослідження різноманітних методологій. Розділи, присвячені окремим алгоритмам - довгій короткочасній пам'яті (LSTM), випадковому лісу, градієнтному бусту, гаусівському наївному Байєсу, K-найближчих сусідів (KNN) та логістичній регресії, - надали цінну інформацію про їхні можливості та застосування.

Вивчення LSTM висвітлило його переваги в моделюванні довгострокових залежностей у послідовних даних, що робить його добре пристосованим до динамічної та еволюціонуючої природи профілів ігрових персонажів. Його унікальна архітектура, що включає вхідні ворота, ворота забування та вихідні ворота, дозволяє мережі вибірково запам'ятовувати або забувати інформацію, що є критично важливим атрибутом для оптимізації здібностей в ігрових сценаріях у реальному часі.

Хоча інші алгоритми, такі як Random Forest, Gradient Boost, Gaussian Naive Bayes, KNN та Logistic Regression, кожен з них має свої переваги .Але складні та нюансовані вимоги поставленої задачі краще узгоджуються з можливостями LSTM.

Представлені розділи в сукупності вказують на довготривалу короткочасну пам'ять як алгоритм, що демонструє найбільшу перспективність і придатність для

складнощів, пов'язаних з прийняттям рішень у реальному часі щодо здібностей ігрових персонажів. Оскільки ігровий ландшафт продовжує розвиватися, LSTM стає надійним та адаптивним рішенням для оптимізації викликів, що виникають у динамічних ігрових сценаріях.

РОЗДІЛ 3
РЕАЛІЗАЦІЯ ТА ПОРІВНЯННЯ LSTM З ІНШИМИ АЛГОРИТМАМИ МАШИННОГО НАВЧАННЯ

3.1 Збір даних та створення датасету для навчання моделі

У сфері конкурентних онлайн-ігор прийняття рішень у реальному часі має першорядне значення для успіху. У цьому розділі розглядається побудова та характеристики набору даних, призначеного для інструменту, спрямованого на прогнозування найкращих навичок героя в реальному часі за допомогою моделей обробки природної мови (NLP) та безперервного мішка слів (CBOW). Для збору даних для цього дослідження OpenDota API надав набір даних, що складається з 50 000 ігрових записів з високим рейтингом, стратегічно відібраних для їх ранжування, забезпечуючи фокус на кваліфікованих і досвідчених гравцях із середнім MMR понад 6000 і числом активних ігор в режимі абіліті драфт більше ніж 100.

Основна мета нашого набору даних - сприяти розробці інструменту, здатного передбачати найефективніші навички героя на етапі підготовки до матчу Dota 2. Використовуючи методи НЛП та модель CBOW, інструмент має на меті покращити процес прийняття рішень, надаючи в реальному часі інформацію про оптимальний вибір навичок на основі мінливого контексту гри. Дані для тренування були побудовані і обрані за методологією, яка була описана в роботах [33-35].

Побудова набору даних

Створення набору даних включало систематичний процес фіксації динаміки взаємодії героїв та вибору навичок під час матчів Dota 2 високого рівня. Ось ключові компоненти побудови набору даних:

1. Інформація про матч:

- Ідентифікатор матчу. Унікальний ідентифікатор, який присвоюється кожному матчу Dota 2, що полегшує детальний пошук інформації про матч.

- Тривалість. Тривалість матчу, що забезпечує контекст для розвитку навичок.

- Етап створення навички. Відмітки часу та деталі фази розробки навичок, що відзначають вирішальні моменти у прийнятті рішень. Ця фаза слугує ключовим моментом для фіксації стратегічних міркувань та взаємодії між героєм і вмінням.

2. Дані про героя та вміння:

- Вибір героя. Інформація про героїв, обраних кожною командою під час фази розробки, що відображає початковий стратегічний вибір.

- Вибір навичок. Послідовні дані про навички, обрані гравцями для своїх героїв, що відображають еволюцію вибору навичок протягом матчу.

- Рівень навички. Рівень, на якому набувається певна навичка, що додає деталізації до набору даних і показує прогрес у розвитку навички.

3. Текстові описи:

- Журнали ігрового чату. Витягнуті журнали чату на етапі розробки, що надають додаткову контекстну інформацію про спілкування гравців та динаміку прийняття рішень.

- Описи героїв та навичок. Текстові представлення атрибутів героя та функціональних можливостей навичок, що збагачують набір даних семантичними деталями та допомагають у проведенні аналізу на основі НЛП.

4. Ігрова статистика:

- Тенденції золота та досвіду. Тенденції накопичення золота та досвіду під час матчу, що дають уявлення про економічну динаміку, яка впливає на вибір навичок.

- Склад команди. Інформація про загальний склад команди кожної сторони, що впливає на вибір навичок і стратегічні міркування.

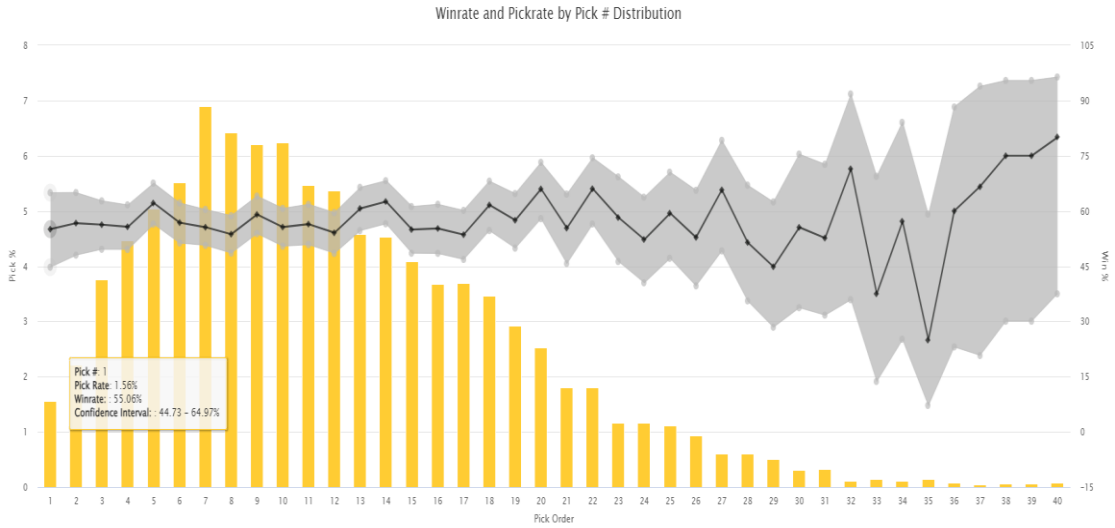


Рис. 3.1. Графік вибору здібності в послідовності драфту

Анотації та маркування

Для ефективного навчання і оцінки моделей NLP і CBOW, набір даних анотується мітками, що вказують на успіх або вплив кожного вибору навичок. Ці мітки виводяться з ігрових показників, таких як завдана шкода, результати командних боїв і загальний результат матчу, забезпечуючи кількісну оцінку ефективності навичок.

Різноманітність наборів даних

Щоб забезпечити надійність інструменту, набір даних охоплює матчі різних рівнів майстерності, починаючи від початківців і закінчуючи професіоналами. Така різноманітність дозволяє моделям NLP і CBOW навчатися на широкому спектрі ігрових сценаріїв і адаптуватися до різних стилів гри, підвищуючи узагальнюваність інструменту прогнозування.

Етичні міркування

Ми дбаємо про анонімність ідентичності гравців та конфіденційної інформації, дотримуючись етичних стандартів у дослідженнях ігор. Пріоритетними є конфіденційність гравців та відповідальне використання даних.

Процес отримання набору даних з OpenDota

Отримання повного та релевантного набору даних є фундаментальним для будь-якого дослідження, заснованого на даних, і в контексті аналізу Dota 2 API OpenDota є цінним ресурсом. OpenDota, платформа, керована спільнотою, надає інтерфейс, за допомогою якого дослідники та розробники можуть отримати доступ до великої кількості даних, пов'язаних з матчами Dota 2, гравцями та ігровою статистикою.

OpenDota API слугує шлюзом до величезної кількості даних про Dota 2. Користувачі зазвичай взаємодіють з цим API для отримання конкретної інформації, необхідної для їхнього аналізу. Для забезпечення безпечного та авторизованого доступу до API може знадобитися автентифікація.

Щоб отримати дані потрібно получити ключ для API. Його можна дістати на сайті opendota.com, що зображено на рис.3.2.

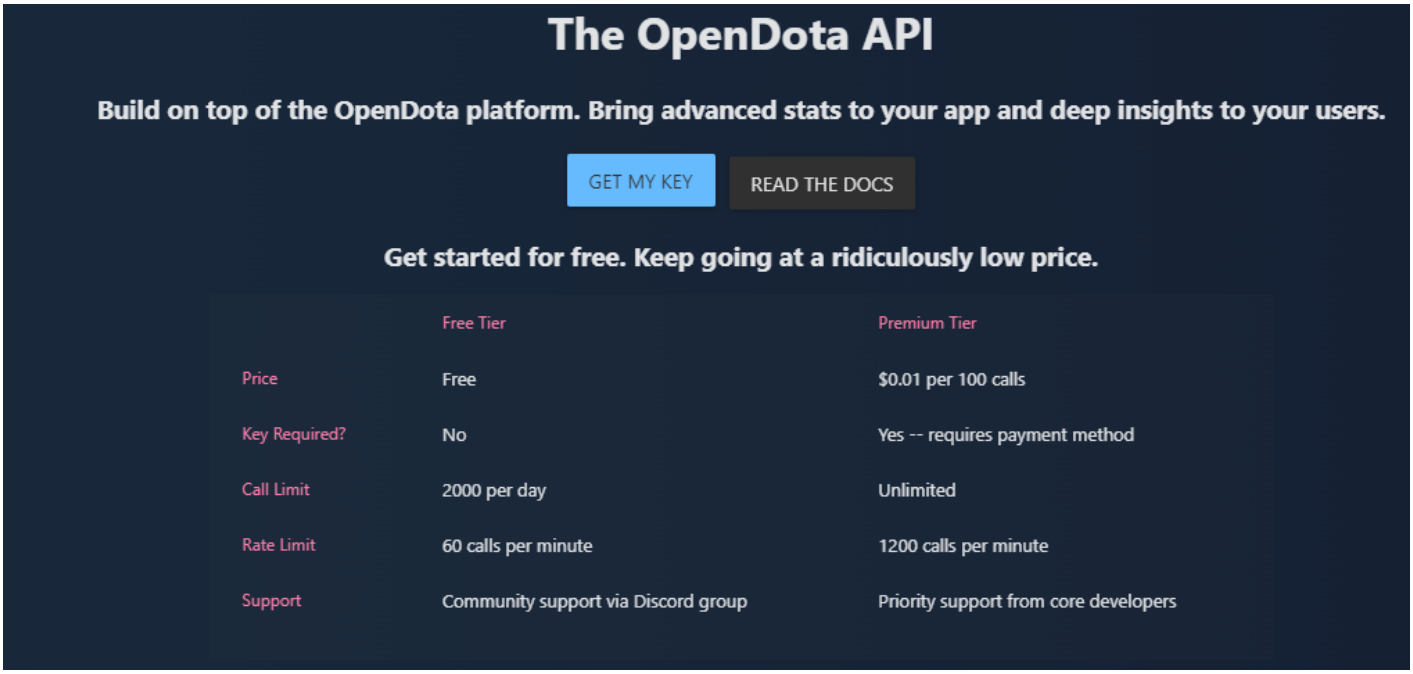


Рис. 3.2. OpenDota API отримання ключа

Далі потрібно вказати свої дані, це зроблено щоб зменшити навантаження на сервері, що покращує роботу цілого сервіса для всіх користувачів.

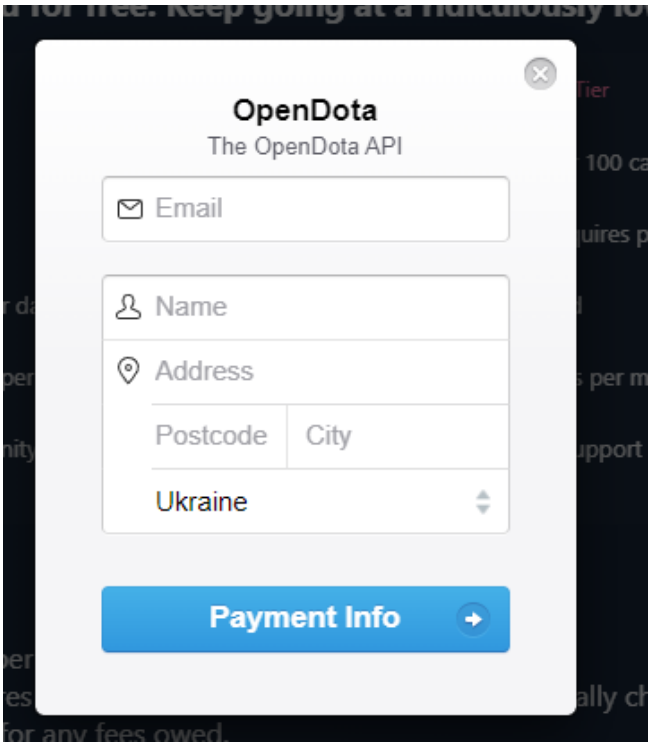


Рис. 3.3. Вибір країни і внесення свої даних

Наступним кроком потрібно зазначити дані своє картки для використання платних можливостей сайту.

Після цього ми отримуємо свій приватний ключ, як ілюстровано на рис.3.4.

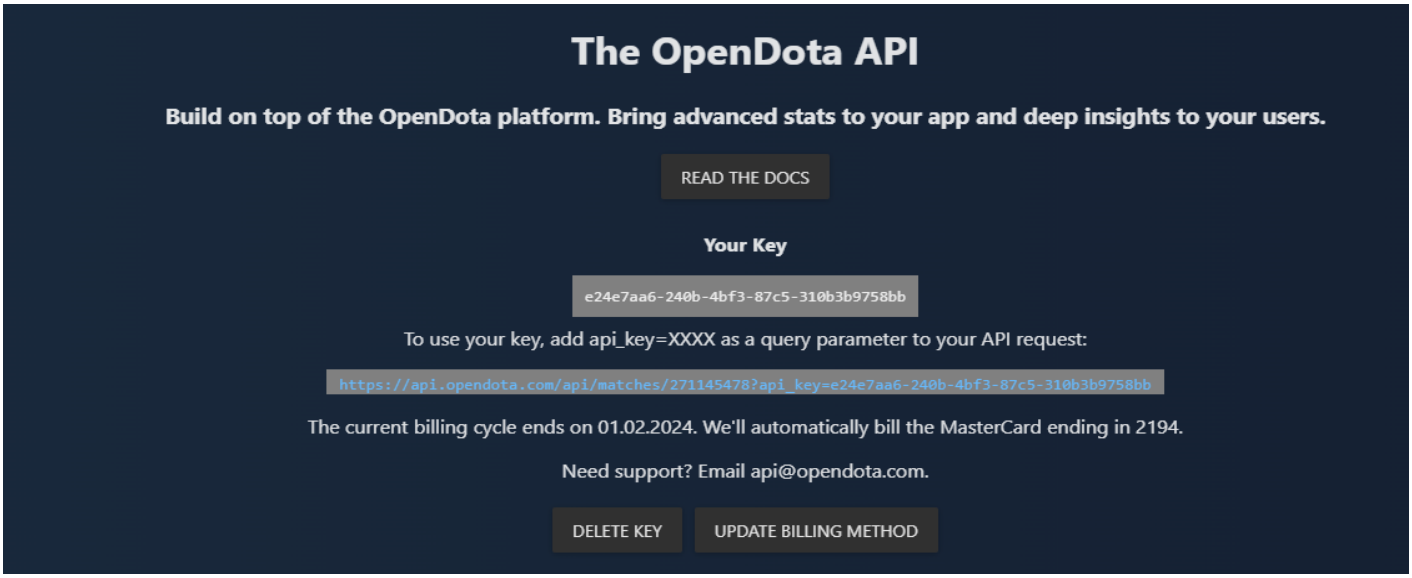


Рис. 3.4. Результат отримання ключа

Визначення параметрів дослідження

Перш ніж робити запити до API, користувачі повинні визначити параметри свого дослідження. Це включає в себе визначення типу необхідних даних, таких як деталі матчу, статистика гравців або інформація про героїв. Крім того, користувачі можуть встановлювати фільтри на основі таких критеріїв, як тривалість матчу, рівень майстерності або конкретні версії гри.

Створення запитів до API

Дослівно API розшифровується як Application Programming Interface. Це набір правил, що дозволяє програмам спілкуватися одна з одною. Розробник створює API на сервері та дозволяє клієнтам звертатися до нього.

REST – це архітектурний підхід, що визначає, як API мають виглядати. Читається як "Representational State Transfer". Цьому набору правил і слідує розробник під час створення свого застосунку. Одне з цих правил каже, що при зверненні до певної адреси ви повинні отримувати певний набір даних (ресурс).

Кожна адреса - маршрут, пакет даних - запит, у той час як результуючий ресурс – відповідь.

Запити до API формулюються на основі визначених параметрів. Ці запити можуть включати запит даних про матчі, профілі гравців, статистику героїв тощо. Користувачі можуть адаптувати свої запити для збору конкретної інформації, необхідної для аналізу.

Приклад створення запиту можна побачити на рис.3.4. Запит формується по посилання сайт opendota та ідентифікатора матчу який потрібно отримати.

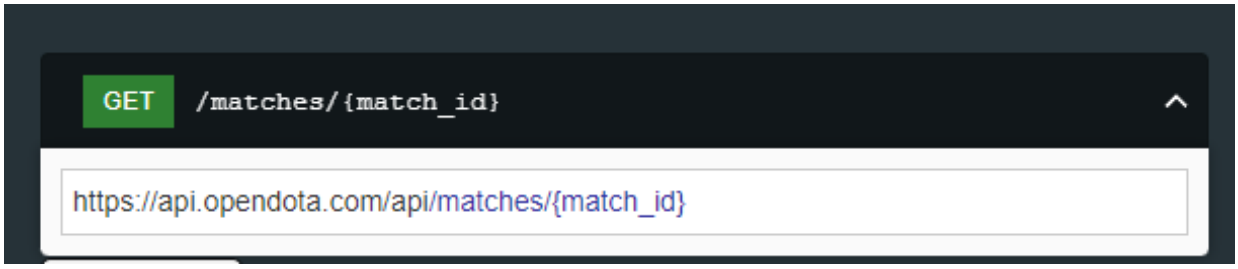


Рис. 3.5. Приклад get запиту

Обробка відповідей від API

На запити OpenDota API відповідає даними у структурованому форматі, зазвичай у JSON (JavaScript Object Notation). Потрібно впровадити механізми для аналізу та вилучення відповідної інформації з цих відповідей, гарантуючи, що набір даних відповідає цілям дослідження.

Рис. 3.6. Частина результату запиту про матч

Ретельна побудова набору даних, що включає дані матчів, текстові описи та різноманітні ігрові сценарії з ігор високого рівня, формує міцну основу для інструменту прогнозування в реальному часі. Цей комплексний набір даних слугує багатим ресурсом для навчання та оцінки моделей NLP і CBOW, створюючи основу для подальших

дискусій щодо впровадження моделей, методологій навчання та оцінки інструменту в контексті ігрового процесу Dota 2, з акцентом на прогнозуванні найкращих навичок героя в реальному часі.

2.2 Continuous Bag of Words (CBOW) модель

Спираючись на фундамент, закладений у попередніх розділах, цей розділ досліджує більш цілісний підхід до прогнозування навичок у Dota 2, інтегруючи міркування щодо поєднання героїв, вибору здібностей та синергії. Прогностична модель має на меті не лише врахувати індивідуальні навички героїв, але й заглибитися в динамічні процеси прийняття рішень, пов'язані з вибором здібностей та синергетичною взаємодією між героями.

У контексті прогнозування навичок модель безперервного мішка слів (CBOW) продовжує слугувати потужним інструментом для аналізу відповідності героїв. Однак, щоб збагатити наші можливості прогнозування, сферу застосування моделі було розширено, щоб включити послідовне прийняття рішень, пов'язаних з вибором здібностей, і потенційні синергії, що виникають в результаті цього вибору.

Вибір здібностей

Набір даних, спочатку створений для вибору героїв і замовлень на вибір, розширено, щоб врахувати еволюцію здібностей героїв протягом матчу. Це розширення охоплює динамічні рішення, які гравці приймають у процесі гри, що впливають на результат бою. Унікальний набір навичок кожного героя та порядок, в якому ці навички розвиваються, є невід'ємними компонентами, що формують траєкторію матчу

Послідовне прийняття рішень відіграє важливу роль для вибору здібностей. Глибокий аналіз того, як гравці адаптують свої набори навичок у критичні моменти або у відповідь на зміну динаміки матчу, має вирішальне значення. Набір даних повинен відображати послідовний процес прийняття рішень, який гравці використовують для оптимізації потенціалу свого героя по ходу гри.

Також важливо розуміти принципи адаптивності та контргри. Випадки, коли гравці відхиляються від стандартних наборів навичок, щоб адаптуватися до конкретних ігрових сценаріїв або протистояти героям-суперникам, повинні бути задокументовані. Адаптивність гравців додає додатковий рівень складності до прогностичної моделі, гарантуючи, що вона залишається універсальною в охопленні широкого спектру стратегічних виборів.

Синергія між здібностями

Сфера застосування прогностичної моделі розширюється за рахунок включення складних взаємозв'язків між здібностями героїв. Синергія, або скоординоване використання здібностей для створення вражаючих комбо, відіграє ключову роль у формуванні результату командних боїв і сутичок.

Комбо та ланцюгові реакції: У цьому наборі даних досліджуються сценарії, в яких герої, що володіють взаємодоповнюючими здібностями, стратегічно розгортають руйнівні комбо. Послідовне використання здібностей різними героями може призвести до ланцюгових реакцій, що посилюють їхній колективний вплив.

Контрсинергія є чатиною Розуміння ситуацій, коли певні здібності протидіють одна одній в одній команді або у відповідь на дії ворога, є не менш важливим. Контрсинергія може призвести до неоптимальних результатів і повинна бути врахована в прогностичній моделі, щоб забезпечити більш тонке розуміння стратегічного вибору.

Наслідки для прогностичних моделей

Інтеграція аналізу вибору здібностей і синергії в прогнозні моделі підвищує їх точність і релевантність в сценаріях реального часу. Враховуючи не тільки індивідуальні навички героя, але й еволюцію навичок та синергію в межах лінії, прогностична модель стає більш надійним інструментом для прийняття стратегічних рішень в ігровому процесі Dota 2.

Перспективи подальших досліджень

По мірі розвитку дослідження, майбутні дослідження повинні заглибитися у взаємодію між підбором героїв, вибором здібностей та синергізмом. Всебічне розуміння

цієї динаміки сприятиме розробці інструментів підтримки прийняття рішень, які надаватимуть гравцям детальні рекомендації, що в кінцевому підсумку підвищить їхню стратегічну майстерність у Dota 2.

У безперервному процесі роботи з даними ретельний процес попередньої обробки даних є ключовим мостом між необробленою інформацією та уточненими вхідними даними, які мають вирішальне значення для розробки моделі машинного навчання. Цей розділ заглиблюється в розширену перспективу попередньої обробки даних, проливаючи світло на включення моделей Word2Vec і матриць зустрічних матриць, сформульованих у log5, що завершується складною системою розділення та оцінки даних.

Інтеграція моделей Word2Vec

Перш ніж заглибитися в тонкощі розділення даних, необхідно підкреслити незамінну роль моделі Word2Vec. Ця модель, налаштована на розмір вектора 150, фіксує семантичні нюанси в наборі даних, орієнтованому на героїв. Надаючи кожному герою багатовимірне векторне представлення, модель Word2Vec покращує розуміння семантичних зв'язків, закладаючи основу для більш контекстно-орієнтованих моделей машинного навчання, що зображено на рис 3.7.

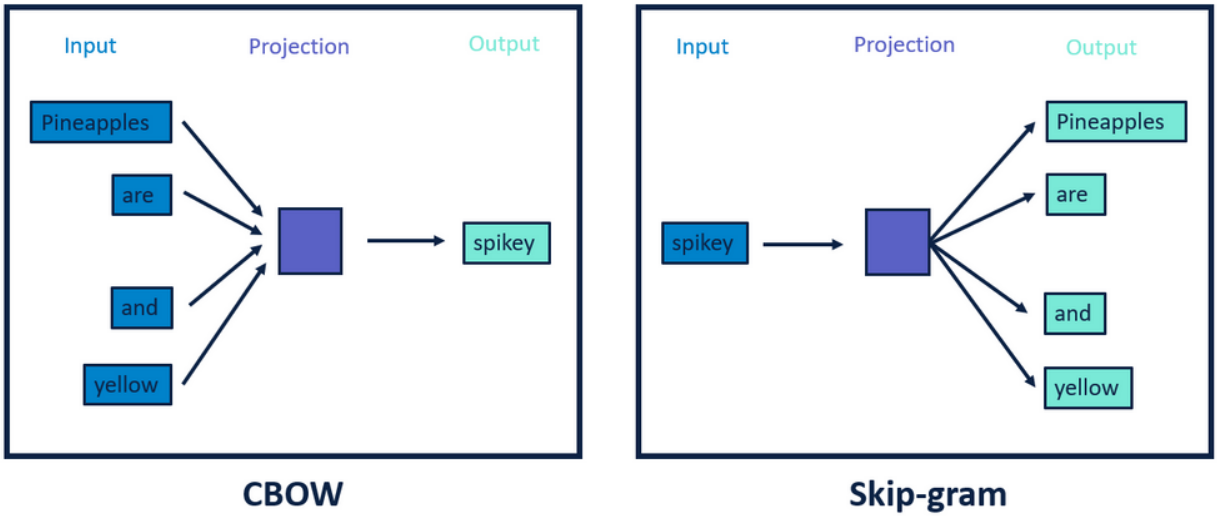


Рис. 3.7. Порівняння CBOW та Slip-gram

Матриці лічильників, сформульовані у форматі Log5: Кількісна оцінка зв'язків між героями

Доповнюючи модель Word2Vec, матриці лічильників, сформульовані в log5, додають кількісний вимір зв'язкам між героями. Обчислені ймовірності перемоги одного героя над іншим дають числове уявлення про стратегічну динаміку між героями. Ця подвійна інтеграція - семантичне розуміння і кількісні відносини між героями - створює всеосяжний набір даних, підготовлений для поглибленого аналізу.

Десятикратна перехресна перевірка

Шлях попередньої обробки даних збігається з нагальною потребою в надійному оцінюванні моделі. Застосування методу 10-кратної перехресної перевірки виходить за рамки традиційних практик розділення даних. На кожній ітерації набір даних розбивається на десять частин, одна з яких слугує для валідації, а решта дев'ять об'єднуються в навчальний набір. Цей ретельний процес повторюється десять разів, гарантуючи, що кожна складова по черзі стає валідаційним набором.

Подолання надмірного пристосування

Першочерговою метою десятикратної перехресної валідації є зменшення ризику надмірного пристосування. Піддаючи моделі машинного навчання різноманітним випробуванням, процес оцінювання сприяє цілісному розумінню їхньої здатності до узагальнення. Цей ітеративний підхід виходить за рамки обмежень одного розбиття навчання на тести, сприяючи створенню моделей, які можуть адаптуватися і надійно працювати в різних сценаріях.

Реальна прогностична здатність

Зрештою, складний танець попередньої обробки даних і надійне оцінювання, якому сприяє десятикратна перехресна перевірка, зливаються воєдино, щоб надати моделям машинного навчання здатність прогнозувати реальні ситуації. Отримані моделі не лише інкапсулюють семантичні уявлення про героїв та кількісні взаємозв'язки між

ними, але й демонструють адаптивність та точність, необхідні для здійснення обґрунтованих прогнозів у динамічних та непередбачуваних ігрових сценаріях Dota 2.

Source Text	Training Samples generated from source text			
I will have orange juice and eggs for breakfast	(will, I)	(will, have)	(will, orange)	
I will have orange juice and eggs for breakfast	(have, I)	(have, will)	(have, orange)	(have, juice)
I will have orange juice and eggs for breakfast	(orange, will)	(orange, have)	(orange, juice)	(orange, and)
I will have orange juice and eggs for breakfast	(juice, have)	(juice, orange)	(juice, and)	(juice, eggs)
I will have orange juice and eggs for breakfast	(and, orange)	(and, juice)	(and, eggs)	(and, for)
I will have orange juice and eggs for breakfast	(eggs, juice)	(eggs, and)	(eggs, for)	(eggs, breakfast)
I will have orange juice and eggs for breakfast	(for, and)	(for, eggs)	(for, breakfast)	

Рис. 3.8. Приклад роботи CBOW

Загалом, ця розширена система обробки даних та оцінки моделей собою наступний етап, що закладає основу для розробки покращених моделей машинного навчання, які підсумовують попередні дослідження, обіцяючи підвищену точність і розширені можливості узагальнення в сфері аналізу ігрового процесу Dota 2.

3.3 Реалізація LSTM та вибір гіперпараметрів

У цьому розділі детально описується вдосконалена архітектура моделі LSTM, розроблена спеціально для аналізу вибору здібностей у реальному часі в Dota 2. Модель втілює в собі кілька хитросплетінь, створених для того, щоб розгадати часову динаміку та стратегічні нюанси, притаманні процесу вибору оптимальною здібності.

Вхідні шари

Важливо розуміти що послідовне представлення вхідних даних є основою для поглибленого аналізу. Модель LSTM інтегрує чотири вхідні шари, стратегічно розроблені для інкапсуляції критично важливої інформації. Два шари призначені для послідовного представлення вибору здібностей для кожної команди, що дозволяє розкрити розгортання наративу п'яти героїв у часі. Кожен здібність, представлена 150-вимірним вектором ознак, робить свій внесок у послідовність, що розвивається. Водночас два інші вхідні шари, що відображають форму (4, 1), дають уявлення про середню частоту зустрічних піків для кожного героя, забезпечуючи часову перспективу розвитку динаміки зустрічної гри. Такий складний дизайн дозволяє моделі розпізнавати як послідовний характер виборів героїв, так і стратегічний вплив частоти зустрічних послідовностей.

Шари LSTM

Завдяки двом LSTM-шарам, кожен з яких має 150 прихованих одиниць, модель заглиблюється в часові абстракції вибору героїв. Вхідні форми цих шарів (batch_size, 4, 150) відображають складний танець між розміром партії, часовими кроками та вхідними вимірами. Діючи як часові кодувальники, ці шари LSTM вправно розплутують еволюційні патерни та залежності в послідовності вибірок здібностей. Їх роль є ключовою в розгадці складної динаміки вибору здібностей протягом декількох часових кроків, забезпечуючи модель всебічним розумінням динаміки збігів, що розвивається.

Механізм уваги

Для підвищення точності моделі введено механізм уваги. Маючи два шари уваги, по одному на шар LSTM, цей механізм покращує фокус на ключових сегментах вхідної послідовності. Працюючи через вхідний 3D тензор, механізм уваги використовує точковий добуток між вивченим ваговим вектором і виходом LSTM, динамічно призначаючи релевантність для різних часових кроків. Після застосування вихідна послідовність проходить по елементне множення з вектором, що представляє ваги уваги. Цей вибірковий акцент слугує для виділення найбільш впливових частин вхідної послідовності, сприяючи сфокусованій обробці інформації.

По елементне множення та конкатенація

Стратегічне злиття розгортається через шари множення, організовуючи об'єднання стосунків героїв та зустрічних впливів. Ці шари допомагають зважити важливість різних елементів на основі відповідних зустрічних показників, пропонуючи складну інтеграцію відносин героїв і зустрічних впливів. Шар Concatenate (Конкатенація) пов'язує ці результати, створюючи інтегрований тензор з розмірами (batch_size, time_steps, 300). Таке інтегративне злиття дозволяє моделі виокремити складні закономірності та взаємозв'язки, що є особливо важливим при оцінці динамічної синергії вибраних здібностей.

Щільні шари

Архітектура моделі складається з повністю пов'язаних щільних шарів, кожен з яких характеризується певною конфігурацією - від 300 до 256 одиниць, а потім від 256 до 64 одиниць. Активовані випрямленими лінійними одиницями (ReLU), ці шари утворюють ієрархічну структуру, що дозволяє моделі відображати складні патерни та ієрархічні особливості. Шар Flatten слугує для перетворення вихідних даних в одномірний вектор, що полегшує інтеграцію з наступними компонентами нейронної мережі.

Вихідний шар

Кульмінацією вихідного шару є сингулярний нейрон, прикрашений сигмоїдною функцією активації, який організовує остаточну прогностичну сутність. Навчений на бінарних показниках втрат перехресної ентропії та точності, цей шар вдосконалює свої параметри для оптимізації прогнозування. Основною метою моделі є не лише передбачення перемоги, але й надання детальної інформації в режимі реального часу про динамічний процес відбору навичок у Dota 2. Вона прагне надати гравцям складний аналіз, що дозволяє їм приймати обґрунтовані рішення щодо вибору здібностей, які відповідають динамічним вимогам кожного матчу.

Гіперпараметри LSTM

Продуктивність і поведінка моделі машинного навчання тісно пов'язані з її гіперпараметрами. Ці параметри впливають на здатність моделі до навчання, її узагальнення на нових даних, швидкість збіжності під час навчання та загальну обчислювальну ефективність. Вдумливий вибір та налаштування гіперпараметрів може значно підвищити продуктивність моделі та точність прогнозів. У цьому розділі ми представляємо ретельно підібрані гіперпараметри для нашої моделі, а також стисле обґрунтування їхнього вибору.

Кількість одиниць LSTM відіграє ключову роль у визначенні складності та репрезентативності шарів LSTM. Враховуючи, що розмір вектора в моделі CBOW дорівнює 150, що означає 150 рис кожного героя, використання шару LSTM зі 150 одиниць дозволяє моделі вправно відображати складні патерни та залежності у вхідних послідовностях, зберігаючи при цьому збалансовану обчислювальну складність.

Введення 64 одиниць у перший щільний шар полегшує врахування нелінійності, дозволяючи виокремлювати особливості та репрезентації більш високого рівня з конкатенації вхідних даних. Крім того, кількість одиниць у другому щільному шарі вводить додаткові шари нелінійних перетворень, збільшуючи здатність моделі розуміти складні закономірності та взаємозв'язки в даних.

Швидкість навчання визначає розмір кроку, з яким модель коригує свої ваги під час навчання. Вибір помірної швидкості навчання, наприклад, 0,0001, забезпечує стабільну і поступову збіжність процесу оптимізації моделі, особливо в сценаріях, що включають складні дані або складну архітектуру моделі.

Для задач бінарної класифікації, таких як прогнозування результатів матчів, бінарні перехресні ентропійні втрати виявляються підходящим вибором. Ця функція втрат вимірює розбіжність між прогнозованими ймовірностями та істинними мітками, стимулюючи модель генерувати точні значення ймовірностей для кожного класу [17].

Adam, добре відомий алгоритм оптимізації, об'єднує сильні сторони AdaGrad і RMSprop. Динамічно адаптуючи швидкість навчання на основі оцінок градієнтного моменту, Adam сприяє ефективній та результативній оптимізації. Його популярність у

моделях глибокого навчання зумовлена сприятливими властивостями збіжності та надійністю [18].

Рання зупинка вступає в гру, щоб передчасно зупинити навчання, якщо продуктивність моделі на валідаційному наборі не покращується протягом певної кількості епох, таким чином запобігаючи надмірному пристосуванню. Зменшити LR на плато коригує швидкість навчання, якщо валідаційні втрати моделі виходять на плато, допомагаючи моделі вдосконалити процес оптимізації і, можливо, уникнути локальних мінімумів.

Вибір 500 епох забезпечує достатню кількість ітерацій, що дозволяє моделі отримувати інформацію з даних, адаптувати вагові коефіцієнти і прагнути до збіжності та оптимальної продуктивності.

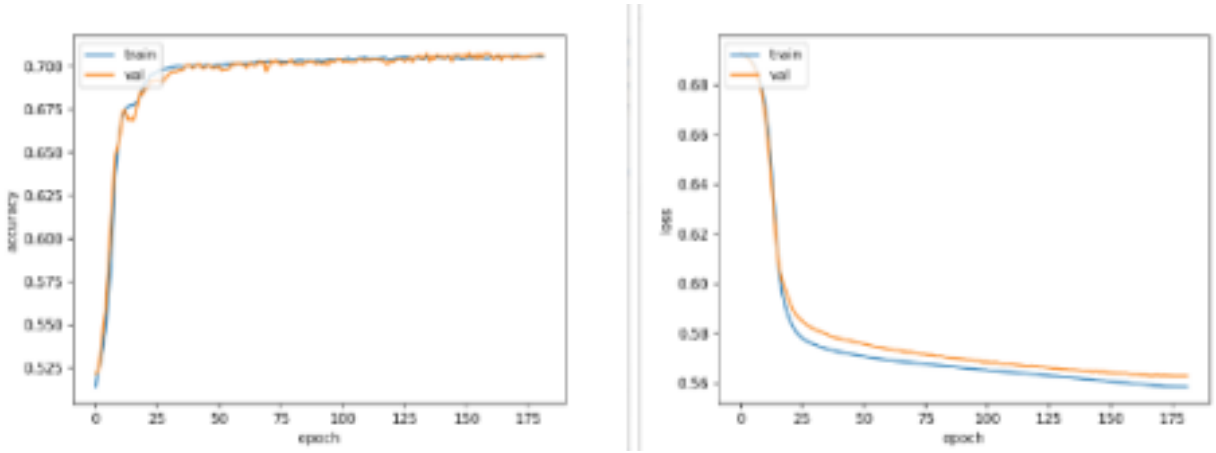


Рис. 3.9. Результати тренувань

Важливе значення має розмір партії, що представляє кількість навчальних вибірок, оброблених перед оновленням вагових коефіцієнтів моделі. Більший розмір пакету, наприклад, 2000, зменшує частоту оновлення ваг, сприяючи отриманню стабільних градієнтів і потенційно прискорюючи навчання, особливо в середовищах з паралельною обробкою даних.

3.4 Висновки до розділу

В даному розділі було описано реалізацію LSTM-моделі та проведено дослідження з практичної точки зору. Результати цього дослідження показують, що модель CBOW з LSTM продемонструвала похвальну точність у 81% у прогнозування вибору здібностей Dota 2. Це підкреслює помітні прогностичні можливості моделі, що свідчить про її потенційну корисність у прогнозуванні майбутніх результатів матчів.

Щоб провести комплексну оцінку ефективності LSTM-моделі та з'ясувати її здатність досягати оптимальних результатів, ми провели порівняльний аналіз з кількома іншими алгоритмами машинного навчання. Набір даних був розділений на навчальний і тестовий набори у співвідношенні 80-20%, що забезпечило рівномірний розподіл виграних і програних матчів. Такий збалансований підхід сприяв неупередженому оцінюванню, уможлививши справедливе порівняння точності прогнозування та здатності до узагальнення серед усіх алгоритмів. Результати надали цінну інформацію про сильні та слабкі сторони кожної моделі, показуючи, чи перевершує модель LSTM інші в точності прогнозування результатів матчів.

Модель LSTM продемонструвала найкращу продуктивність серед протестованих алгоритмів, досягнувши точності 81,65%. Її здатність фіксувати довгострокові залежності та зберігати важливу інформацію з плином часу виявилася корисною для завдання прогнозування. Незважаючи на пристойні результати інших алгоритмів, таких як KNN, Logistic Regression, Gradient Boost, Random Forest та Gaussian Naive Bayes, модель LSTM постійно перевершувала їх, підтверджуючи свою перевагу в точності.

Ці результати підкреслюють цінність використання LSTM-моделей для подібних завдань, вносячи цінний внесок у сферу машинного навчання.

Отже, це дослідження надає багатообіцяючі докази того, що CBOW з LSTM-моделлю може слугувати цінним інструментом для прогнозування вибору здібностей в Dota 2. Тим не менш, необхідні подальші дослідження для повної оцінки ефективності моделі та її практичної застосовності.

РОЗДІЛ 4
СТВОРЕННЯ ПРОГРАМИ ДЛЯ ОПТИМІЗАЦІЇ ПРОФІЛІВ ІГРОВИХ ПЕРСОНАЖІВ ГЕРОЇВ НА ОСНОВІ ПОБУДОВАНОЇ LSTM МОДЕЛІ

4.1 Основні проблеми розробки

4.1.1 Користувацький інтерфейс для безперешкодної взаємодії

Dota 2 Skill Enhancement Tool може похвалитися ретельно розробленим користувацьким інтерфейсом (UI), який надає пріоритет безперешкодному та цікавому досвіду для гравців. Наголошуючи на доступності та зручному дизайні, за допомогою якого гравці можуть використовувати розширені аналітичні можливості моделі LSTM.

Інтуїтивно зрозумілі принципи дизайну

Інтерфейс дотримується інтуїтивно зрозумілих принципів дизайну, гарантуючи, що гравці, незалежно від їхнього ігрового досвіду, можуть легко орієнтуватися в ньому. Чистий і організований макет мінімізує когнітивне навантаження, дозволяючи користувачам зосередитися на стратегічній інформації та рекомендаціях, що надаються інструментом.

Оновлення та візуалізація в режимі реального часу

Щоб підтримувати актуальність у динамічному середовищі Dota 2, інтерфейс легко інтегрує оновлення в режимі реального часу. Гравці можуть в реальному часі спостерігати за змінами у виборі героїв, частоті зустрічних виборів і прогнозному аналізі, що підвищує їхню здатність приймати обґрунтовані рішення під час матчу.

Інтерактивний інтерфейс вибору здібності

Етап вибору героя є критично важливим аспектом Dota 2, і користувальницький інтерфейс інструменту відображає його важливість. Інтерактивний інтерфейс вибору здібності дозволяє користувачам візуалізувати вибір в залежності від обрання своєї команди та суперників у режимі реального часу. Таке візуальне представлення допомагає гравцям зрозуміти динаміку матчу.

Рекомендації щодо вибору здібності

Інтерфейс користувача представляє рекомендації щодо вибору здібностей у чіткій та лаконічній формі. По мірі розвитку матчу і вибору героїв інструмент динамічно оновлює свої рекомендації, надаючи гравцям своєчасну інформацію про оптимальний вибір здібностей. Візуальні підказки та підказки допомагають інтерпретувати ці рекомендації.

Можливості кастомізації

Враховуючи різноманітність стилів гри та вподобань гравців, інтерфейс включає в себе опції налаштування. Користувачі можуть налаштувати параметри відображення, наприклад, відрегулювати рівень деталізації в прогнозному аналізі або вибрати конкретні показники для фокусування. Така гнучкість дозволяє гравцям персоналізувати свій досвід. Інтерфейс має інтуїтивно зрозумілий механізм зворотного зв'язку, який заохочує користувачів надавати свої коментарі щодо рекомендацій інструменту. Цей інтерактивний елемент не лише підвищує залученість користувачів, але й сприяє ітеративному вдосконаленню базових моделей, сприяючи розвитку відносин співпраці між інструментом та його користувачами.

Таким чином, користувальницький інтерфейс Dota 2 Skill Enhancement Tool створений як динамічний портал, орієнтований на користувача, який надає гравцям доступ до передової аналітики та стратегічної інформації. Завдяки продуманому дизайну та інтерактивним функціям інтерфейс покращує загальний ігровий досвід, роблячи процес прийняття стратегічних рішень більш доступним і приємним для гравців усіх рівнів майстерності.

4.1.2 Інтеграції даних у режимі реального часу

Перевага Dota 2 Skill Enhancement Tool полягає у вмілій інтеграції даних у режимі реального часу, що дозволяє користувачам отримувати доступ до актуальної інформації та стратегічного аналізу під час ігрового процесу. У цьому розділі розглядаються

тонкощі інтеграції даних у реальному часі, підкреслюється роль інструменту в забезпеченні гравців динамічним і чуйним компаньйоном під час матчів у Dota 2.

Безперервна передача даних

В основі можливостей інструменту в режимі реального часу лежить його безшовне з'єднання з OpenDota API, що забезпечує безперервний потік даних з поточних матчів Dota 2. Цей механізм безперервної передачі даних гарантує, що інструмент залишається синхронізованим з динамікою ігрового процесу в реальному часі. Отримуючи критично важливу інформацію, таку як вибір героя, рухи гравців та ігрові події, інструмент створює надійну основу для аналізу в реальному часі.

Движок динамічної обробки даних

Після отримання даних у реальному часі інструмент задіює складний механізм обробки даних, призначений для швидкої інтерпретації та категоризації вхідної інформації. Ця можливість динамічної обробки дозволяє інструменту виокремлювати відповідні закономірності, тенденції та дієві ідеї з потоку ігрових даних у реальному часі. Оскільки інструмент адаптується до постійно мінливого ландшафту матчів Dota 2, користувачі отримують вигоду від миттєвого аналізу, який допомагає у прийнятті стратегічних рішень.

Адаптивна візуалізація

Інтеграція даних у режимі реального часу доповнюється зручною системою візуалізації, яка динамічно відображає нові тенденції та зміни в ігрових подіх. Візуальні підказки, такі як графіки, діаграми та статистика в реальному часі, надають гравцям чіткий та інтуїтивно зрозумілий огляд матчу, що розгортається. Така адаптивна візуалізація не тільки покращує користувацький досвід, але й сприяє швидкому розумінню критично важливої інформації в розпалі ігрового процесу.

Оновлення в режимі реального часу

Однією з ключових переваг інтеграції інструменту в режимі реального часу є його здатність надавати користувачам актуальну інформацію про поточні дії гравців, командні стратегії та результати матчів. Оскільки гравці приймають рішення і

виконують маневри в режимі реального часу, інструмент оперативно відображає ці дії, гарантуючи, що користувачі мають найсвіжішу інформацію для прийняття рішень в грі.

Адаптивні інсайти для прийняття стратегічних рішень

Інтегруючи дані в режимі реального часу, інструмент стає адаптивним компаньйоном, який пристосовує свої висновки до мінливого характеру матчів Dota 2. Миттєвий аналіз слугує стратегічним орієнтиром для користувачів, пропонуючи цінну інформацію для вибору героїв, створення предметів та загальної стратегії гри. Ця адаптивна природа дозволяє гравцям приймати обґрунтовані рішення відповідно до обставин, що постійно змінюються в реальному часі матчу.

Таким чином, інтеграція даних в режимі реального часу в Dota 2 Skill Enhancement Tool є наріжним каменем його функціональності. Завдяки безперешкодному включенню даних гри в реальному часі інструмент надає користувачам своєчасну, релевантну та динамічну інформацію, сприяючи збагаченню ігрового досвіду в Dota 2 та формуванню стратегічних рішень.

4.1.3 Інтеграція моделі LSTM

Інтеграція моделей довготривалої короткочасної пам'яті (LSTM) в Dota 2 Skill Enhancement Tool додає ще один рівень складності прогнозування, розширюючи можливості інструменту пропонувати стратегічні рекомендації на основі історичних даних і даних в реальному часі. У цьому розділі пояснюється складна інтеграція LSTM-моделей, окреслюється їхній внесок у прогнозування та підтримку стратегічних рішень.

Розпізнавання історичних патернів

Інтегровані в інструмент LSTM-моделі використовують історичні дані ігрового процесу для розпізнавання складних закономірностей і кореляцій між вибором героїв, стратегіями гравців і результатами матчів. Навчені на великих масивах даних, що охоплюють різні рівні майстерності та стилі гри, ці моделі відображають тонкі взаємозв'язки між різними ігровими змінними. Розпізнавання історичних

закономірностей є основою для прогнозування потенційного успіху певних комбінацій героїв та стратегічних маневрів.

Підтримка прийняття рішень у реальному часі

Оскільки інструмент динамічно інтегрує потоки даних у реальному часі, моделі LSTM постійно адаптують свої прогнози на основі подій, що розгортаються під час живих матчів Dota 2. Цей механізм підтримки прийняття рішень в режимі реального часу дозволяє інструменту надавати користувачам миттєву інформацію про динаміку розвитку поточного матчу. Поєднуючи історичні патерни з поточним ігровим процесом, моделі LSTM сприяють створенню комплексної системи прийняття рішень для користувачів.

Адаптивний механізм навчання

Моделі LSTM, вбудовані в інструмент, демонструють адаптивний механізм навчання, постійно вдосконалюючи свої прогностичні можливості з кожною новою порцією інформації. Ця адаптивність гарантує, що інструмент завжди реагує на нові мета-стратегії, синергію героїв і тенденції гравців. Ітеративний процес навчання дозволяє моделям LSTM розвиватися разом з постійно мінливим ландшафтом ігрового процесу Dota 2, забезпечуючи довговічність та актуальність їхніх прогнозів.

Багатовимірний аналіз

Моделі LSTM виконують багатовимірний аналіз, враховуючи різні фактори, такі як частота зустрічних виборів героїв, історичні ймовірності перемоги і показники ігрового процесу в реальному часі. Такий комплексний підхід дозволяє інструменту генерувати нюанси, що виходять за рамки простих прогнозів виграшу/програшу. Користувачі отримують вигоду від більш глибокого розуміння складної динаміки гри, що дає їм можливість приймати стратегічні рішення, які відповідають конкретним нюансам поточних матчів.

Користувацьке налаштування

Інтеграція LSTM-моделей в інструменті розроблена з орієнтацією на користувача, що дозволяє гравцям налаштовувати та адаптувати фокус моделі відповідно до своїх

уподобань та стилів гри. Користувачі можуть вказати певні параметри, підкреслити певні аспекти аналізу та відрегулювати вагу, яка присвоюється історичним даним порівняно з інформацією в реальному часі. Ця функція кастомізації гарантує, що інструмент відповідає стратегічним уподобанням користувача, сприяючи персоналізованому та адаптивному ігровому досвіду.

Динамічний цикл зворотного зв'язку

Невід'ємним компонентом інтеграції моделі LSTM є створення динамічного циклу зворотного зв'язку. Коли користувачі взаємодіють з інструментом, приймають ігрові рішення та спостерігають за результатами матчів, LSTM-моделі асимілюють цей зворотний зв'язок для подальшого вдосконалення своїх прогнозів.

На закінчення, інтеграція LSTM-моделі в Dota 2 Skill Enhancement Tool являє собою поєднання історичних знань і адаптивності в режимі реального часу. Завдяки складному механізму навчання ці моделі підвищують прогностичні можливості інструменту, надаючи користувачам багатовимірну, керовану користувачем і динамічно реагуючу систему підтримки прийняття стратегічних рішень.

4.2 Створення алгоритмів для програми

4.2.1 Створення основного алгоритму програми

На даному етапі розробки в нас є готова LSTM модель, тепер потрібно створити програму для використання.

У складній сфері Ability Draft у Dota 2 пошук оптимізації здібностей героя в режимі реального часу вимагає складної взаємодії між ігровою динамікою та передовими моделями машинного навчання. У цьому розділі ми створимо та опишемо основні алгоритми програми, висвітливши поетапний процес, який розгортається від початку матчу до кульмінації фази драфту.

Отже, алгоритм для програми:

1. Програма визначає, що матч почався за допомогою алгоритму, який описаний у підрозділі 4.2.2.

   Програма починає свою послідовність з виявлення початку збігу. Цей ключовий момент запускає розгортання наступних кроків у процесі оптимізації в реальному часі.
2. Визначення пулу героїв і доступних здібностей, який описаний у підрозділі 4.2.3.

   Після ініціювання матчу програма динамічно визначає пул доступних героїв і пов'язані з ними здібності. Ця динамічна інвентаризація слугує основою для подальших процесів прийняття рішень.
3. Програма відає дані готові моделі.

   Програма надсилає скомпільовані дані, що містять поточний пул героїв та доступні здібності, попередньо навченим моделям машинного навчання. Ці моделі вміють розпізнавати закономірності та робити обґрунтовані прогнози щодо оптимального вибору здібностей.
4. Модель визначає найкращу здібність для вибору.

   Використовуючи алгоритми, закладені в моделях машинного навчання, система визначає найкращу здатність для кожного героя в режимі реального часу. Цей процес прийняття рішень ґрунтується на здатності моделей аналізувати поточний стан гри та прогнозувати оптимальний вибір.
5. Коли настає черга для вибору користувача програма фіксує вибрану здібність і передає нові дані моделі.

   У процесі гри, коли настає черга користувача обирати здатність, програма фіксує обрану здатність і передає оновлені дані назад моделям машинного навчання. Цей ітеративний цикл гарантує, що моделі постійно отримують інформацію про динаміку розвитку проекту.
6. Якщо найкращий вибір було забрано іншим гравцем програма пропонує нову здібність для вибору.

У випадках, коли на оптимальний вибір вже претендує інший гравець, програма динамічно пропонує користувачеві альтернативні можливості для розгляду. Цей адаптивний підхід гарантує, що користувачі постійно отримують життєздатні варіанти.

7. Кроки 5-6 повторяються до закінчення драфту.
   Кроки 5 і 6 ітеративно розгортаються до кульмінації етапу підготовки проекту. Цей цикл у реальному часі гарантує, що моделі машинного навчання адаптуються до вибору гравців, пропонуючи оптимальні пропозиції протягом усього драфту.

4.2.2 Створення алгоритму визначення матчу

Можливі два варіанта події: користувач вибирає сам коли починати скан екрану для визначення коли починається матч або діє автоматичне зчитування для визначення цього процесу.

Алгоритм для визначення початку:

1. Від значення змінної start_game_setting_auto визначається поведінка, якщо значення false тоді, функція закінчується з вихідними значення true
2. Якщо start_game_setting_auto= True тоді працює цей алгоритм.
3. Загрузка карти супроводжується спінер і словами “Loading”, за допомогою бібліотеки OpenCV, визначається стан загрузки, перевірка діє кожну секунду. Якщо загрузка йде більше певного числа програма призупиняється.
4. Якщо напис відсутній то починається пошук напису “Ability Draft”, який з’являється тільки після загрузки.

4.2.3 Створення алгоритму визначення пулу героїв і доступних здібностей

Щоб визначити пул потрібно дізнатися 10 героїв + 2 сета рандомних скілів. Це можливо виконати за допомогою бібліотеки OpenCV.

Алгоритм для визначення пулу героїв:

1.Визначаєм стартову точку для перебору матриці скілів

2.Вибирає першу здібність у рядку

3.Визначаєм сет здібностей по першому здібності за допомогою даних які збираєм за допомогою API

4. Повторюємо 3 крок поки не закінчать рядки із здібностями

5. Проходимо окремим циклом по значках ультимативних здібностей для знаходження 2 випадкових здібностей.

4.3 Програмна реалізація

4.3.1 Створення середовища розробки та структура програмно забезпечення

Python є винятковим вибором для створення програмного забезпечення для оптимізації профілів ігрових персонажів, особливо в контексті героїв стратегічних ігор у реальному часі, за допомогою алгоритмів машинного навчання. Універсальність Python та великі бібліотеки роблять її найкращою мовою для розробки надійних та ефективних моделей машинного навчання. Читабельність мови та простота використання сприяють пришвидшенню циклів розробки, дозволяючи розробникам безперешкодно впроваджувати та ітерації алгоритмів машинного навчання.

У сфері машинного навчання Python може похвалитися низкою потужних бібліотек та фреймворків, таких як TensorFlow, PyTorch та scikit-learn. Ці бібліотеки надають готові функції та модулі для створення та навчання моделей машинного навчання, що значно прискорює процес розробки. Популярність Python у спільноті машинного навчання забезпечує безліч ресурсів, навчальних посібників та підтримку спільноти, що полегшує розробникам доступ до останніх досягнень та найкращих практик у галузі машинного навчання.

Крім того, Python легко інтегрується з іншими технологіями, які зазвичай використовуються у розробці ігор, забезпечуючи безперебійну взаємодію. Простота коду Python дозволяє швидко створювати прототипи та експериментувати, що має

вирішальне значення при точному налаштуванні моделей для оптимізації профілів ігрових персонажів у реальному часі. Незалежно від того, чи реалізовуєте ви складні нейронні мережі, чи традиційні алгоритми машинного навчання, адаптивність Python та всеосяжна екосистема роблять його ідеальним вибором для створення інтелектуальних та адаптивних систем у динамічному середовищі розробки ігор.

Щоб встановити потрібно перейти на офіційний сайт і вибрати потрібну версію, що зображено на рис.4.1.

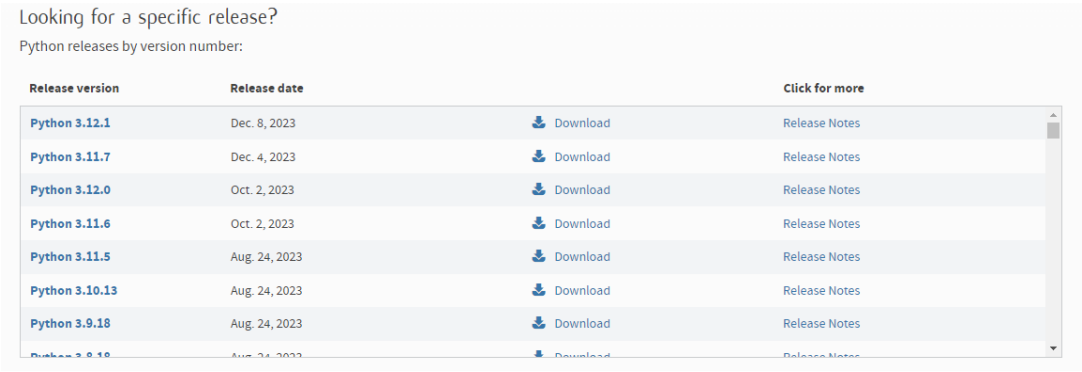


Рис.4.1. Вибір версії Python

Після встановлення створимо папку з ефективно структурою розподілу по модулів по їхніх властивостей, як можна побачити на рисунку 4.2

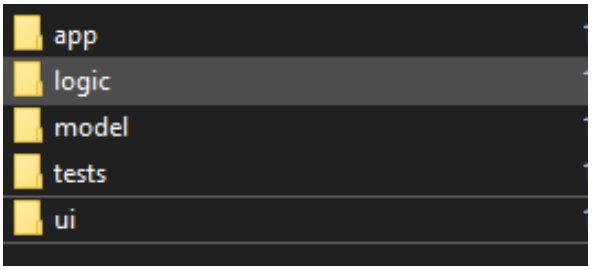


Рис.4.2. Структура кореневої папки

Поділ коду на різні модулі - це хороша практика для підтримки чистої, модульної та організованої кодової бази. Розробляючи систему з користувацьким інтерфейсом (UI),

логікою та компонентами моделі машинного навчання, потрібно структурувати свій код в окремі модулі, щоб покращити зручність супроводу та масштабованість.

UI модуль, спеціально призначений для роботи з користувацьким інтерфейсом. Цей модуль міститиме код, пов'язаний з відображенням інформації користувачеві та збором даних, що вводяться користувачем. Використаємо бібліотеку інтерфейсу користувача, наприклад, PyQt, щоб ефективно створювати графічні інтерфейси користувача. В цьому модулі містяться функції або класи для обробки подій інтерфейсу та оновлення інтерфейсу на основі стану програми.

Рис.4.3. Структура модуля UI

Створимо модуль для управління логікою додатку. Цей модуль повинен відповідати за бізнес-правила, обробку даних та інші функції, не пов'язані з інтерфейсом. Реалізуємо функції або класи для виконання обчислень, управління ігровою логікою та взаємодії з модулем інтерфейсу. Потрібно зробити логічний модуль незалежним від інтерфейсу, щоб його було легше тестувати і повторно використовувати в різних контекстах. Поділ коду на інтерфейсі та логічні модулі покращує розробку програмного забезпечення, сприяючи модульності та організації коду. Такий підхід полегшує незалежну розробку і тестування компонентів користувацького інтерфейсу та базової логіки, що призводить до більш чистої та зручної для підтримки кодової бази. Модульність також дозволяє покращити співпрацю між розробниками, оскільки вони

можуть працювати над різними аспектами програми одночасно. Крім того, розділення інтерфейсу та логіки забезпечує гнучкість для майбутніх оновлень або адаптації, полегшуючи масштабування програми та інтеграцію нових функцій без зміни всієї структури коду. Загалом, такий модульний підхід сприяє повторному використанню, масштабуванню та ефективному обслуговуванню програмних систем.

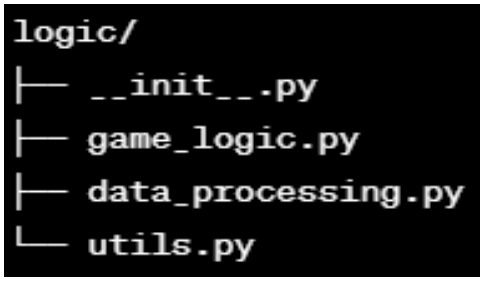


Рис.4.4. Структура модуля logic

Наш додаток передбачає машинне навчання, тому створюємо модуль спеціально для керування навченими моделями. Створимо функції або класи для завантаження попередньо навчених моделей, прогнозування та оновлення інтерфейсу користувача або логіки на основі результатів роботи моделі. Цей модуль повинен інкапсулювати всі взаємодії з бібліотеками та фреймворками машинного навчання.

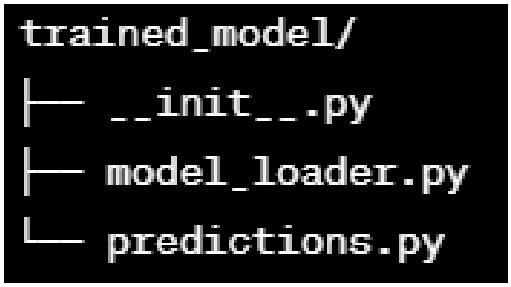


Рис.4.5. Структура модуля trained_model

Створімо головний модуль, який слугуватиме точкою входу для вашого додатку. Імпортуйте та узгодьте функціональність модулів інтерфейсу, логіки та навченої моделі. Цей модуль повинен ініціалізувати інтерфейс, пов'язувати події інтерфейсу з

відповідною логікою та керувати загальним потоком роботи програми. В середині цього модуля буде реалізований main loop, частина патерну цикл подій, що надай змогу очікувати прибуття і виробляти розсилку подій або повідомлень у програмі.

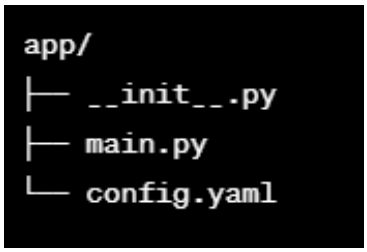


Рис.4.6. Структура модуля app

Перспективним є створення окремого модуля для тестів, щоб забезпечити коректність кожного модуля. Використаєм фреймворк для тестування, pytest, для написання та запуску тестів на функціональність інтерфейсу, логіки та моделі.

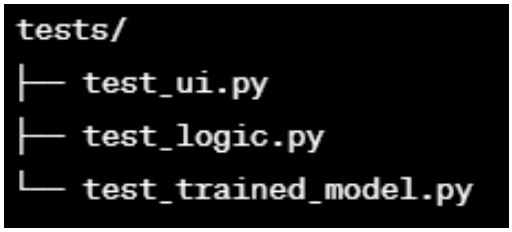


Рис.4.7. Структура модуля tests

4.3.2 Створення системи визначення пулу здібностей

Для того щоб отримати дані про доступні здібності для вибору, без порушення умов використання Dota 2, потрібно створити модуль який буде визначати доступних герої по картинці використовуючи бібліотеку OpenCV.

Лістинг 4.1.

```
model_version = "microsoft/trocr-base-printed"

processor = TrOCRProcessor.from_pretrained(model_version)
```

```
    model = VisionEncoderDecoderModel.from_pretrained(model_version)

    pixel_values = processor(image, return_tensors="pt").pixel_values

    generated_ids = model.generate(pixel_values)

    generated_text = processor.batch_decode(generated_ids,
skip_special_tokens=True)[0]
    print(generated_text)
```

Використанні цієї бібліотеки зумовлена рядом причин. А саме тому що OpenCV - це комплексна бібліотека з відкритим вихідним кодом, спеціально розроблена для задач комп'ютерного зору. Вона надає широкий набір інструментів, функцій та алгоритмів, які призначені для вирішення таких завдань, як аналіз зображень та відео, виявлення об'єктів. Багатофункціональність бібліотеки робить її найкращим вибором для розробників, які працюють над різноманітними проектами комп'ютерного зору.

OpenCV розроблена як кросплатформенна, що дозволяє розробникам без проблем розгортати свої додатки на різних операційних системах. Ця сумісність гарантує, що розроблені рішення можуть працювати на різних платформах без значних модифікацій. Ця гнучкість має вирішальне значення для додатків, орієнтованих на різноманітну базу користувачів. OpenCV має велику і активну спільноту розробників і дослідників. Підтримка спільноти означає, що розробники мають доступ до великої кількості ресурсів, включаючи форуми, навчальні посібники та документацію. Це може бути надзвичайно корисним при вирішенні проблем, вивченні нових функціональних можливостей або пошуку порад щодо найкращих практик. Обширна документація також полегшує реалізацію складних алгоритмів і методів.

Бібліотека реалізовано на мовах C та C++, які відомі своєю продуктивністю. Це дозволяє розробникам створювати високопродуктивні програми для обробки зображень у реальному часі. Крім того, OpenCV надає інтерфейси для Python та інших мов, поєднуючи ефективність мов низького рівня з простотою використання мов високого рівня. OpenCV широко використовується як в академічних колах, так і в промисловості. Про його популярність свідчить використання в різних галузях, включаючи

робототехніку, доповнену реальність, автономні транспортні засоби та аналіз медичних зображень. Вибір OpenCV забезпечує сумісність з галузевими стандартами і практиками, що робить його надійним і перевіреним інструментом в області комп'ютерного зору.

OpenCV активно підтримується і регулярно оновлюється для включення останніх досягнень в області комп'ютерного зору. Це гарантує, що розробники мають доступ до найсучасніших методів. Постійне оновлення має вирішальне значення для використання новітніх технологій та забезпечення довгострокової життєздатності проекту. Отже, оптимальне рішення щодо використання OpenCV полягає в його універсальності, підтримці спільноти, крос-платформенній сумісності, продуктивності, прийнятті в індустрії та постійному розвитку. Ці фактори в сукупності роблять OpenCV надійним вибором для цієї задачі.

4.3.3 Створення системи для визначення обраних здібностей

Важливою частиною програми для оптимізації вибору профіля є інформація про доступність найкращою здібності. Для реалізація цього фунціоналу, також використовується бібліотека OpenCV, основним тригером для визначення того, що здібність була вибрана є ефект який накладається на зображення іконки здібності, це зображено на рис.4.8.

Рис.4.8. Приклад ефекту вибраної здібності

Ця візуальна підказка слугує основним тригером для програми, щоб переконатися, що обрано певну здібність. Використовуючи методи комп'ютерного зору, програма забезпечує надійний механізм відстеження динамічного стану здібностей у режимі

реального часу на етапі драфту. Це не тільки підвищує точність процесу оптимізації профілю, але й демонструє синергію між алгоритмами машинного навчання та комп'ютерним зором, пропонуючи комплексне рішення для точного прийняття рішень у постійно мінливому ландшафті Ability Draft.

4.3.4 Інтеграція готової моделі

Після того, як LSTM-модель буде навчена і оптимізована для прогнозування здібностей героя або інших важливих характеристик, експортуйте модель у серіалізований формат, наприклад, збережену модель Keras або TensorFlow SavedModel.

В інструменті Python завантажем експортовану LSTM-модель за допомогою бібліотеки глибокого навчання, наприклад, TensorFlow або Keras. Це передбачає використання функцій на кшталт load_model для десеріалізації навченої моделі.

Лістинг 4.2.

```
from tensorflow.keras.models import load_model
lstm_model = load_model('trained_model.h5')
```

Підготовляєм вхідні дані для LSTM-моделі відповідно до вимог визначених в розділі 3. Це може включати попередню обробку вхідних даних користувача, перетворення їх у формат, очікуваний LSTM-моделлю, і забезпечення відповідності даних формі вхідних даних моделі.

Лістинг 4.3.

```
predictions = lstm_model.predict(user_input_data)
```

Інструмент призначений для взаємодії в режимі реального часу, тому інтегруєм механізм для безперервної подачі даних в реальному часі в LSTM-модель. Це включає налаштування конвеєрів даних, які надають змогу швидко та точно реагувати на зміни

в поточному пулі здібностей та виборів, як суперників та і союзників, що є важливою характеристикою розробленої моделі.

4.3.5 Створення простого інтерфейсу

Для створення інтерфейсу було використано Pyqt. Було вирішено створити простий інтерфейс який буде найкраще відображати суть програми і не буде заважати насолоджуватися гемплейними моментами в грі. А саме відображення потрібні здібності напроти портрету вашого героя ця задача легко реалізується за допомогою коду в лістингу 4.4.

Лістинг 4.4.

```
class AbilityWidget(QtGui.QWidget):
    def __init__(self, parent=None):
        QtGui.QWidget.__init__(self, parent=parent)
        self.initUI()
    def initUI(self,custom_h,custom_w):
        self.resize(custom_h, custom_w)
        self.center()
        self.setWindowTitle('Browser')
        self.lb = QtGui.QLabel(self)
        pixmap = QtGui.QPixmap("ablities/sven_1.png")
        height_of_label = 100
        self.lb.resize(self.width(), height_of_label)
        self.lb.setPixmap(pixmap.scaled(self.lb.size(),
QtCore.Qt.IgnoreAspectRatio))
        self.show()
    def resizeEvent(self, event):
        self.lb.resize(self.width(), self.lb.height())
        self.lb.setPixmap(self.lb.pixmap().scaled(self.lb.size(),
QtCore.Qt.IgnoreAspectRatio))
        QtGui.QWidget.resizeEvent(self, event)
    def center(self):
        qr = self.frameGeometry()
        cp = QtGui.QDesktopWidget().availableGeometry().center()
        qr.moveCenter(cp)
        self.move(qr.topLeft())
def main():
    app = QtGui.QApplication(sys.argv)
    w = AbilityWidget()
    app.exec_()
```

Такий варіант реалізація даної частини програми є оптимальною, через використання класів, що дає можливість використовувати переваги об'єктно-орієнтованого програмування. В даному прикладі ми наслідуємо клас QtGui.QWidget, який є віджетом. Віджет - це атомом користувацького інтерфейсу, який отримує події від миші, клавіатури та інші події від віконної системи і малює своє представлення на екрані. Кожен віджет має прямокутну форму, і вони відсортовані у Z-подібному порядку. Віджет обмежується своїм батьком і віджетами, що знаходяться перед ним.

Ця частина інтерфейсу є реактивно тобто при зміні найкращого вибору для гравця іконка здібності зміниться на потрібну, як зображено на рис. 4.9.

Рис.4.9. Приклад інтерфейсу з однією рекомендацією

Також реалізований варіант який відображає чотири найкращі вибори цей функціонал, можна настроїти при запуску програми.

Отже, такий варіант інтерфейсу є простий та ефективний, дає змогу побачити потрібну інформацію і не закриває вікно гри при виборі.

4.4 Висновки до розділу

В даному розділі було розглянуті основні проблеми для розробки програмно забезпечення для оптимізації профілів, описані основні алгоритми функціональної системи. Маючи на руках алгоритми, важливим етапом стало в реальне програмне

забезпечення. Складний процес реалізації програмного забезпечення передбачав інтеграцію моделі LSTM у структуру програми.

Розроблено програму яка інтегрує треновану модель. Необхідність дотримання тонкого балансу між обчислювальною ефективністю та точністю моделі підкреслила нюансований характер проблем, з якими ми зіткнулися. Центральним елементом успіху нашої програми є формулювання алгоритмів, які використовують потужність LSTM. Завдання полягало не лише у виборі відповідних функцій, а й у створенні механізмів, які могли б динамічно адаптуватися до постійно мінливого характеру ігрового процесу Dota 2. Програма, наділена здатністю оптимізувати профілі ігрових персонажів, була реалізована завдяки ретельному кодуванню, тестуванню та вдосконаленню.

ВИСНОВКИ

В даній магістерській роботі було розглянуто проблему створення моделі оптимізації профілів ігрових персонажів героїв в реальному часі та створення програмного забезпечення для динамічної роботи систем на основі готової до використання, тренованої моделі. Було досліджено на наявність схожих рішень, як з'ясувалось всі дослідження фокусувалися на визначенні перемоги чи поразки у матчі, що надає відкритий простір для створення потрібного рішення.

Для створення моделі було проведено дослідження для визначення оптимального алгоритму для поставленої задачі за допомогою порівняння відомих моделей та їх результатів. Проаналізовано алгоритми нейромереж: Random Forest, Gradient Boost, Gaussian Naive Bayes, K-Nearest Neighbors, Logistic Regression, Long Short-Term Memory.

Після дослідження стало відомо, що такою моделлю буде LSTM з використання технологій CBOW. Така архітектура моделі дає можливість оцінювати довгострокові відношення між відсотками перемоги та відсотками вибору здібності. Що є ключовим фактором для оптимізації профілів.

Здійснено детальний опис алгоритмів системи при виконанні її головних функцій. На основі цих даних було програмно реалізовано систему, що оптимізує профілі ігрових персонажів героїв в реальному часі на основі створеної моделі машинного навчання. Для доступу до динамічних даних було використано бібліотеку OpenCV, а для створення інтерфейсу PyQt.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Zach Cleghern,Soumendra Lahiri,Osman Ozaltin, and David L Roberts. Predicting future states in dota 2 using value-split models of time series attribute data. In Proceedings of the 12th International Conference on the Foundations of Digital Games, pages 1–10, 2017.
2. Kevin Conley and Daniel Perry. How does he saw me? a recommendation engine for picking heroes in dota 2. Np, nd Web, 7, 2013.
3. Anders Drachen, Matthew Yancey, John Maguire, Derrek Chu, Iris Yuhui Wang, Tobias Mahlmann, Matthias Schubert, and Diego Klabajan. Skill-based differences in spatiotemporal team behaviour in defence of the ancients 2 (dota 2). In 2014 IEEE Games Media Entertainment, pages 1–8. IEEE, 2014.
4. Petra Grutzik, Joe Higgins, and Long Tran. Predicting outcomes of professional dota 2 matches. Technical report, Technical Report. Stanford University, 2017.
5. Juho Hamari and Max Sjoblom. What is esports and why do people watch it? Internet research, 2017.
6. Victoria J Hodge, Sam Michael Devlin, Nicholas John Sephton, Florian Oliver Block, Peter Ivan Cowling, and Anders Drachen. Win prediction in multi-player esports: Live professional match prediction. IEEE Transactions on Games, 2019.
7. Filip Johansson and Jesper Wikstrom. Result prediction by mining replays in dota 2, 2015.
8. Kaushik Kalyanaraman. To win or not to win? a prediction model to determine the outcome of a dota2 match. Technical report, Technical Report. Technical report, University of California San Diego, 2014.
9. Ilya Makarov, Dmitry Savostyanov, Boris Litvyakov, and Dmitry I Ignatov. Predicting winning team and proba- bilistic ratings in “dota 2” and “counter-strike: Global offensive” video games. In International Conference on Analysis of Images, Social Networks and Texts, pages 183–196. Springer, 2017.

10. Aleksandr Semenov, Peter Romov, Sergey Korolev, Daniil Yashkov, and Kirill Neklyudov. Performance of machine learning algorithms in predicting game outcome from drafts in dota 2. In International Conference on Analysis of Images, Social Networks and Texts, pages 26–37. Springer, 2016.
11. Kuangyan Song, Tianyi Zhang, and Chao Ma. Predicting the winning side of dota2. Sl: sn, 2015.
12. Nanzhi Wang, Lin Li, Linlong Xiao, Guocai Yang, and Yue Zhou. Outcome prediction of dota2 using machine learning methods. In Proceedings of 2018 International Conference on Mathematics and Artificial Intelligence, pages 61–67, 2018.
13. Weiqi Wang. Predicting multiplayer online battle arena (moba) game outcome based on hero draft data. PhD thesis, Dublin, National College of Ireland, 2016.
14. Yifan Yang, Tian Qin, and Yu-Heng Lei. Realtime esports match result prediction. arXiv preprint arXiv:1701.03162, 2016.
15. Shitole Ruturaj. Player compability and win prediction in Dota 2 using Graph Neural Network, 2023.
16. Gilmer, Justin et al. “Neural Message Passing for Quantum Chemistry.” ArXiv abs/1704.01212 (2017): n. pag.
17. Cui, Y., Jia, M., Lin, T. Y., Song, Y., & Belongie, S. (2019). Class-balanced loss based on effective number of samples. In Proceedings of the IEEE/CVF conference on computer vision and pattern recognition (pp.9268-9277).
18. Kingma, D. P., & Ba, J. (2014). Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980.
19. Christopher Olah. Understanding LSTM Networks,2015.
20. Breiman, L. Random Forests. Machine Learning 45, 5–32 (2001).
21. Friedman, J. H. (March 1999). "Stochastic Gradient Boosting".
22. Mason, L.; Baxter, J.; Bartlett, P. L.; Frean, Marcus (1999). "Boosting Algorithms as Gradient Descent" (PDF). In S.A. Solla and T.K. Leen and K. Müller (ed.). Advances in Neural Information Processing Systems 12. MIT Press. pp. 512–518.

23. Fix, Evelyn; Hodges, Joseph L. (1951). Discriminatory Analysis. Nonparametric Discrimination: Consistency Properties. USAF School of Aviation Medicine, Randolph Field, Texas. Archived from the original on September 26, 2020.
24. Cover, Thomas M.; Hart, Peter E. (1967). "Nearest neighbor pattern classification" (PDF). IEEE Transactions on Information Theory. 13 (1): 21–27. CiteSeerX 10.1.1.68.2616. doi:10.1109/TIT.1967.1053964. S2CID 5246200.
25. Beyer, Kevin; et al. "When is "nearest neighbor" meaningful?". Database Theory—ICDT'99. 1999: 217–235.
26. Berkson, Joseph (1944). "Application of the Logistic Function to Bio-Assay". Journal of the American Statistical Association. 39 (227): 357–365.
27. Hilbe, Joseph M. (2009). Logistic Regression Models. Chapman & Hall/CRC Press. ISBN 978-1-4200-7575-5.
28. Hosmer, David (2013). Applied logistic regression. Hoboken, New Jersey: Wiley. ISBN 978-0-470-58247-3.
29. McCallum, Andrew. "Graphical Models, Lecture2: Bayesian Network Representation". Archived from the original on 2022-10-09. Retrieved 22 October 2019.
30. Russell, Stuart; Norvig, Peter (2003) [1995]. Artificial Intelligence: A Modern Approach (2nd ed.). Prentice Hall. ISBN 978-0137903955.
31. Hand, D. J.; Yu, K. (2001). "Idiot's Bayes — not so stupid after all?". International Statistical Review. 69 (3): 385–399. doi:10.2307/1403452. ISSN 0306-7734. JSTOR 1403452.
32. Sepp Hochreiter; Jürgen Schmidhuber (1997). "Long short-term memory". Neural Computation. 9 (8): 1735–1780.
33. Hochreiter, Sepp (1991). Untersuchungen zu dynamischen neuronalen Netzen (diploma thesis). Technical University Munich, Institute of Computer Science.
34. Graves, A.; Liwicki, M.; Fernández, S.; Bertolami, R.; Bunke, H.; Schmidhuber, J. (May 2009). "A Novel Connectionist System for Unconstrained Handwriting Recognition". IEEE Transactions on Pattern Analysis and Machine Intelligence. 31 (5): 855–868.

35. Sak, Hasim; Senior, Andrew; Beaufays, Francoise (2014). "Long Short-Term Memory recurrent neural network architectures for large scale acoustic modeling" (PDF). Archived from the original (PDF) on 2018-04-24.
36. Malhotra, Pankaj; Vig, Lovekesh; Shroff, Gautam; Agarwal, Puneet (April 2015). "Long Short Term Memory Networks for Anomaly Detection in Time Series" (PDF). European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning — ESANN 2015. Archived from the original (PDF) on 2020-10-30. Retrieved 2018-02-21.
37. Mikolov, Tomas; et al. (2013). "Efficient Estimation of Word Representations in Vector Space". arXiv:1301.3781 [cs.CL].
38. Mikolov, Tomas; Sutskever, Ilya; Chen, Kai; Corrado, Greg S.; Dean, Jeff (2013). Distributed representations of words and phrases and their compositionality. Advances in Neural Information Processing Systems. arXiv:1310.4546. Bibcode:2013arXiv1310.4546M.
39. Goldberg, Yoav; Levy, Omer (2014). "word2vec Explained: Deriving Mikolov et al.'s Negative-Sampling Word-Embedding Method". arXiv:1402.3722.
40. Adrian Kaehler; Gary Bradski (14 December 2016). Learning OpenCV 3: Computer Vision in C++ with the OpenCV Library. O'Reilly Media. pp. 26ff. ISBN 978-1-4919-3800-3.