

Міністерство освіти і науки України
НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ «КИЄВО-МОГИЛЯНСЬКА АКАДЕМІЯ»
Кафедра інформатики факультету інформатики



Кваліфікаційна робота

освітній ступінь - магістр

на тему: **Застосування великих мовних моделей до розробки стратегій в настільних іграх**

Виконав: студент 2-го року навчання
освітньої програми «Інженерія
програмного забезпечення»,
спеціальності 121 Інженерія
програмного забезпечення

Найда Андрій Олегович

Керівник магістерської роботи: д.ф.м.н.,
доц. Ігнатенко О. П.

Рецензент: д.т.н., доц. Глибовець А.М.

Кваліфікаційна робота захищена
з оцінкою _____
Секретар ЕК _____
« ____ » _____ 20 ____ р.

Київ 2024

Міністерство освіти і науки України
НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ «КИЄВО-МОГИЛЯНСЬКА АКАДЕМІЯ»
Кафедра інформатики факультету інформатики

ЗАТВЕРДЖУЮ
Зав. кафедри інформатики
к.ф.-м.н., доц. Гороховський С.С

(підпис)
“ ____ ” _____ 2023 р.

ІНДИВІДУАЛЬНЕ ЗАВДАННЯ
на магістерську роботу

студенту 2 р.н. магістерської програми Інженерія Програмного Забезпечення
Найді Андрію Олеговичу
Дослідити Використання настільних ігор для дослідження успішності великих
мовних моделей

Зміст текстової частини до магістерської роботи:

Зміст

Анотація

Вступ

- 1 Теоретичні засади тестів продуктивності мовних моделей.
- 2 Опис тесту продуктивності на базі настільної гри, структура програми.
- 3 Застосування гри BraveRats як бенчмарк для різних LLM проти оптимальної стратегії гри.

Висновки по роботі та рекомендації для подальших досліджень

Список літератури

Додатки

Дата видачі “ ____ ” _____ 2023 р.

Керівник

Ігнатенко О. П., доктор фізико-математичних наук, доцент _____
(підпис)

Завдання отримав

Найда А. О. _____
(підпис)

Тема: Застосування великих мовних моделей до розробки стратегій в настільних іграх

Календарний план виконання роботи:

№ п/п	Назва етапу дипломного проекту (роботи)	Термін виконання етапу	Примітка
1.	Отримання завдання на дипломну роботу	16.10.2023	
2.	Огляд технічної літератури за темою роботи	17.11.2023	
3.	Опис рішення, структур даних, генерація правил гри	20.12.2023	
4.	Побудова дерева рішень та виокремлення виграшних стратегій	21.01.2024	
5.	Огляд доступних до використання LLM та впровадження коректного уніфікованого запиту	26.01.2024	
6.	Огляд можливостей тюнігу ChatGpt, Bard AI,	01.02.2024	
7.	Побудова інфраструктури вимірювання ефективності запитів вирішення гри, створення таблиць порівняння	15.03.2024	
8.	Аналіз отриманих результатів з керівником, написання доповіді та попередній захист магістерської роботи	28.05.2024	
9.	Коригування роботи за результатами попереднього захисту	31.05.2024	
10.	Захист магістерської роботи (проекту)		

Студент Найда А. О.

Керівник Ігнатенко О. П.

ЗМІСТ

ВСТУП.....	4
РОЗДІЛ 1: Аналіз предметної області. Постановка завдання магістерської роботи.....	6
1.1 Аналіз застосувань LLM до ігор та контексту правил середовища....	6
1.2 Аналіз існуючих тестів продуктивності LLM.....	10
1.3 Постановка задачі.....	12
РОЗДІЛ 2. ТЕОРЕТИЧНІ ВІДОМОСТІ.....	14
2.1 Поняття великих мовних моделей.....	14
2.2 Поняття рішення гри та домінуючої стратегії.....	15
РОЗДІЛ 3. ОПИС РІШЕННЯ, СТРУКТУРИ ПРОГРАМИ ТА ГЕНЕРАЦІЇ ДЕРЕВА СТРАТЕГІЙ.....	16
3.1 Структура коду.....	17
РОЗДІЛ 4. ВИСНОВКИ.....	30
СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ.....	32

ВСТУП

В останні роки великі мовні моделі, такі як GPT (Generative Pre-trained Transformer), стали об'єктом інтенсивного дослідження та застосування в різних галузях, включаючи обробку природної мови (NLP), автоматичне підсумовування тексту, машинний переклад, створення контенту та багато іншого. Незважаючи на базовий механізм статистичного передбачення слів в контексті запитання, моделі часто мають здатність вирішувати математичні задачі, писати код, відповідати на нові на запитання з використанням загальних відомостей на яких була навчена модель. Однак, багато аспектів їхньої ефективності та достовірність відповідей ще залишаються під питанням.

Ця робота присвячена аналізу використання великих мовних моделей в настільних іграх та тестах продуктивності великих мовних моделей на прикладі настільних ігор. Вона важлива з кількох причин. По-перше, настільні ігри є відмінним джерелом великого обсягу текстових даних, які можуть бути використані для навчання та оцінки мовних моделей. По-друге, дослідження ефективності цих моделей на конкретних завданнях гри може принести нові уявлення про їхні можливості та обмеження.

Метою цієї роботи є ретельний аналіз продуктивності великих мовних моделей, а також вивчення їхнього використання в контексті настільних ігор. Крім того, метою є визначення оптимальних параметрів та опису запитів для досягнення найкращих результатів у завданнях, пов'язаних з ігровим середовищем. Робота також ставить за мету створити власну систему оцінювання моделі в порівнянні до повного рішення гри.

Предметом дослідження є аналіз контекстного вікна та тестування продуктивності великих мовних моделей на прикладі настільних ігор. Робота буде фокусуватися на вивченні того, як мовні моделі будують стратегію, їхню здатність прогнозувати подальший хід гравця, а також на їхню точність та ефективність у грі.

РОЗДІЛ 1: Аналіз предметної області. Постановка завдання магістерської роботи

1.1 Аналіз застосувань ВММ до ігор та контексту правил середовища

Великі мовні моделі є популярним наразі рішенням багатьох відкритих питань, та залучаються до симуляцій відповідей людей в багатьох сферах.

Так в контексті вирішень ігор можна виокремити залучення ВММ до військових ігор. В статті IQT Labs[1] досліджується інтеграція великих мовних моделей в екосистему військових ігор, підкреслюється їхня здатність автоматизувати агентів та покращувати якість існуючих рішень іграх без дискретного вибору. В статті надається як теоретична основа, так і практична реалізацію на Python для демонстрації цієї інтеграції. Не менш важливою перевагою ВММ в контексті військових ігор є те, що вони дозволяють гравцям, як ШІ, так і людям, робити творчі іноваційні ходи, судячи їх з контексту загальної логіки і правил світу. Теоретично, ідеальна якісна симуляція військової гри на основі ВММ обмежується лише тим, які ходи є правдоподібними, а не чіткими правилами ходів, які автор симуляції військової гри мав час (або творчий потенціал), щоб написати.

Також існують дослідження імплементації ВММ в розробці ігор, використання їх як агентів для не ігрових персонажей та для заміни ролі майстра гри в настільно рольових іграх. Так автори "Великі мовні моделі та ігри: Огляд і дорожня карта"[2] стверджують що ВММ можуть ефективно функціонувати в певних типах ігор, перетворюючи свої типові мовні виводи у відповідні до гри дії. Це передбачає відображення послідовностей токенів або виходів природної мови безпосередньо на вхідні дані гри. Великі мовні моделі показали себе перспективними в

покрокових стратегічних іграх, таких як шахи, де ходи можуть бути закодовані у вигляді послідовностей токенів, а також у більш складних умовах, таких як текстові пригодницькі ігри, де сюжет гри може керувати реакціями моделі. Хоча ВММ досягають успіху в структурованих середовищах, проблеми виникають у динамічних умовах або іграх, що вимагають високого рівня стратегічного мислення або візуальної інтерпретації. Наприклад, хоча ВММ можуть працювати в рамках структурованих правил настільних ігор або текстових пригод, адаптація цих моделей до стратегічних ігор у реальному часі або візуально складних середовищах, таких як шутери від першої особи, вимагає більш складної обробки даних і навчання моделям. Окрім ігор, ВММ також використовуються для покращення ігрового досвіду за допомогою не ігрових персонажей. Ці персонажі можуть брати участь у реалістичних діалогах і адаптувати свою поведінку залежно від контексту гри, значно збагачуючи взаємодію гравців і загальну глибину оповіді гри.

Серед досліджених джерел є приклад альтернативного оцівання великих мовних моделей за допомогою простих ігор на відгадування зашифрованих слів, таких як "Wordl". У статті "Використання ігор на відгадування слів для оцінки інтелекту великих мовних моделей"[3] представлено інноваційні фреймворки для оцінки інтелекту великих мовних моделей (ВММ) за допомогою ігор на відгадування слів. Основними фреймворками, що обговорюються, є DEEP (Dual Expression Evaluation Program) та SpyGame. DEEP оцінює ВММ, вимагаючи від них описувати слова у двох режимах: агресивному та консервативному. В агресивному режимі ВММ повинні надавати чіткі та детальні описи, тоді як у консервативному режимі вони прагнуть створити неоднозначні описи, які не дають змоги легко вгадати слово. Цей метод перевіряє здатність ВММ генерувати як точні, так і завуальовані описи. SpyGame - це

складніша багатоагентна інтерактивна система, заснована на грі "Хто шпигун". Вона оцінює ВММ, пропонуючи їм грати ролі в конкурентній грі на відгадування слів, де вони повинні використовувати стратегічне мислення та лінгвістичні навички. У статті повідомляється, що ці фреймворки успішно розрізняють ВММ з відкритим і закритим кодом, підкреслюючи їхні різні сильні сторони в розумінні контексту, точності опису і створенні неоднозначності. Дослідження робить значний внесок у розуміння можливостей ВММ у розумінні природної мови та стратегічного ігрового процесу, пропонуючи більш нюансований підхід до оцінки штучного загального інтелекту в реальних умовах.

Стаття "Мовні агенти з навчанням з підкріпленням для стратегічної гри у грі-перевертня"[4] представляє фреймворк, який поєднує великі мовні моделі з навчанням з підкріпленням (reinforced learning) для розробки стратегічних мовних агентів. Ці агенти призначені для гри в "Перевертня" - гру на соціальну дедукцію, що вимагає складних навичок прийняття рішень і комунікації. Генеруючи різноманітні варіанти дій за допомогою ВММі обираючи найкращу дію за допомогою політики, навченої навчанням з підкріпленням, агенти ефективно пом'якшують внутрішні упередження, характерні для чистих агентів, що базуються на ВММ. Результати масштабних експериментів показують, що ці агенти не тільки працюють краще, ніж існуючі агенти на основі ВММ, але й досягають людського рівня в стратегічному геймплеї.

У статті "Дослідження великих мовних моделей для комунікативних ігор: Емпіричне дослідження гри Перевертень"[5] також досліджується використання великих мовних моделей у грі на соціальну дедукцію "Перевертень", проте зосереджуючись на тому, як ці моделі можуть покращити ігрові стратегії за допомогою комунікації. У ній представлено підхід без налаштування, де ВММ використовуються у своїх базових

формах, покладаючись на їхні можливості обробки природної мови для інтерпретації та розробки стратегій в ігровому контексті. Дослідження демонструє, що ВММ можуть ефективно брати участь у складних комунікаційних іграх, використовуючи свою здатність аналізувати минулі взаємодії та адаптуватися до динаміки гри, що розвивається, в стратегічній грі без необхідності додаткового навчання або налаштування моделі.

Стаття “Повторювані ігри з великими мовними моделями”[5] досліджує низку базових проблем з теорії ігор і оцінює моделі на їх базі. Так в дослідженні використана низка проблем що мають рівновагу по нешу або рівновагу неша в змішаних стратегіях. Такі проблеми подаються моделі в вигляді пояснення результатів матриць і на повторюваних дослідах збирається статистика успішності вибору рішень моделі. Таке використання моделей має обмежений передаваний контекст і не розглядає проблематику далекоглядного стратегічного планування, проте досліджує можливість моделі обрати найкраще рішення серед запропонованих в рамках однораундових ігор. В даній роботі були використані метрики усередненого результату моделі поділеного на величину, що можна отримати за оптимального рішення на низці проблем.

Game family	Llama 2	Claude 2	davinci-002	davinci-003	GPT-4
Second best	0.486	0.735	0.473	0.692	0.763
Biased	0.632	0.794	0.629	0.761	0.798
Cyclic	0.634	0.749	0.638	0.793	0.806
Unfair	0.641	0.812	0.683	0.833	0.836
PD Family	0.731	0.838	0.807	0.841	0.871
Win-win	0.915	0.878	0.988	0.972	0.992
Overall	0.697	0.814	0.730	0.839	0.854

Рисунок 1.1 Порівняння ВММ

Де з результатів дослідження робляться висновки про оцінку кожної моделі. Автори роблять висновок, що хоча великі мовні моделі, такі як GPT-4, добре виконують індивідуальні завдання в теоретико-ігровому середовищі, їхня координація в соціальних іграх, таких як «Дилема в'язня» та «Битва статей», є неоптимальною. Це підкреслює необхідність кращої координації в соціальних контекстах. Дослідження пропонує впроваджувати підказки, які заохочують соціальну аргументацію для покращення взаємодії. Дослідження також підкреслює потенціал для подальших досліджень складних ігор та взаємодії людини і ВММ, пропонуючи підхід з точки зору поведінкових наук для кращого розуміння ВММ.

1.2 Аналіз існуючих тестів продуктивності LLM

На сьогоднішній день існує велика кількість бенчмарків які досліджують великі мовні моделі, проте їхня різноманітність і кількість вказує на необхідність тестування моделей в різних умовах та для різних задач. Так з популярних зараз бенчмарків виділяють AI2 Reasoning Challenge, HellaSwag, MMLU, TruthfulQA, WinoGrande, GSM8K, SuperGLUE та HumanEval.

AI2 Reasoning Challenge (ARC) - це бенчмарк для оцінки здатності мовних моделей до розуміння та мислення. Він створений з метою перевірки здатності моделей вирішувати завдання різної складності на базі наукових питань та відповідей. Основними характеристиками є датасет, що складається з 7787 наукових питань з чотирма варіантами відповідей. Питання охоплюють широкий спектр наукових дисциплін і можуть бути різної складності, від 3-го до 9-го класу. Питання в ARC здебільшого

мають такі категорії: фактичні, просторові, експериментальні та алгебраїчні. Це робить бенчмарк різноманітним і комплексним, що підштовхує розробників штучного інтелекту на покращення навичок міжгалузевих зв'язків, а не лише витягування фактів з існуючих речень в тренувальній базі.

HellaSwag (скорочено від Harder Endings, Longer contexts, and Low-shot Activities for Situations with Adversarial Generations) - це бенчмарк, спрямований на тестування здатності мовних моделей до розуміння загального здорового глузду. Він є наступником бенчмарку SWAG (Situations With Adversarial Generations) та складається з серії вправ на завершення речень. Датасет HellaSwag містить ряд завдань, кожне з яких складається з підпису до відео як початкового контексту та чотирьох можливих закінчень, з яких лише одне є правильним. Датасет HellaSwag був створений за допомогою алгоритму адверсарного фільтрування, який підвищує складність завдань шляхом генерації обманливих неправильних відповідей, що містять слова та фрази, що стосуються контексту, але суперечать загальним знанням про світ. Такі завдання гарно ілюструють можливість моделей давати достовірну відповідь, але завдання може бути важкими для моделей, які не мають широкого контекстуального розуміння або досвіду в обробці природної мови.

Massive Multitask Language Understanding (MMLU) - це широкий бенчмарк, призначений для вимірювання здатності мовних моделей до розуміння мови та їхню здатність розв'язувати проблеми з використанням знань, які вони отримали під час навчання. Цей бенчмарк розроблений з метою викликати моделі на їхні здатності розуміти мову в широкому спектрі завдань, що охоплюють якісний, так і кількісний аналіз. Датасет MMLU складається з 15,908 питань, які поділені на 57 завдань, зроблених

з різних онлайн-джерел. Ці завдання охоплюють широкий спектр тем, включаючи STEM (наука, технології, інженерія і математика), гуманітарні науки (мовні мистецтва, історія, соціологія, виконавські та образотворчі мистецтва) та інші. Це допомагає тестувати моделі на широкому спектрі завдань та тем, що робить бенчмарк більш універсальним. Проте деякі дослідження показують наявність помилок у датасеті, що може ускладнити оцінку результатів моделей.

GSM8K (Grade School Math 8K) - це бенчмарк, призначений для вимірювання здатності мовних моделей до розв'язання математичних завдань на рівні початкової школи. Цей бенчмарк складається з набору близько 8,500 математичних задач, які були створені людьми та підібрані таким чином, щоб вони були доступні для розв'язання для учнів початкової школи. Завдання в GSM8K вимагають від моделей виконання послідовності дій, щоб знайти правильну відповідь. Складність задач збільшується з кількістю кроків, необхідних для їх розв'язання. Вправи в GSM8K охоплюють різні теми математики, від базових арифметичних операцій до простих геометричних задач.: Відповіді на задачі зберігаються у вигляді природної мови, що дозволяє виміряти здатність моделей не лише до розуміння математичних концепцій, але й до генерації відповідей у вигляді тексту. В датасеті обмежена різноманітність тем, оскільки бенчмарк фокусується переважно на базових арифметичних операціях. Проте саме вміння моделей вирішувати нетривіальні задачі послугувало натхненням для цієї роботи.

1.3 Постановка задачі

З огляду на існуючі рішення та актуальність визначеної проблеми було визначено наступну постановку задачі.

Основне завдання - створити середовище, що симулює стратегічну настільну гру з неочевидними вигравними комбінаціями, впровадити якісний контекст середовища на кожному етапі гри та сформулювати завдання для великої мовної моделі для отримання якісного результату. Реалізація повинна включати зручну можливість імплементації будь-якого API великих мовних моделей та оцінювання їх проти незмінного еталону. Кожна модель має спиратися на початковий контекст правил, які мають бути сформульовані чітко і зрозуміло. Середовище має зазначати про стратегічну важливість будування рішень, спонукати виграти цілісну гру, а не лише раунд в ній. Проект має містити можливість спрощення гри та можливість генерації рішень для кожного обмеженого варіанту гри.

РОЗДІЛ 2. ТЕОРЕТИЧНІ ВІДОМОСТІ

З огляду на поставлену задачу, основні теоретичні відомості, що необхідні для ознайомлення з роботою пов'язані з роботою великих мовних моделей та поняттям вирішення гри в теорії ігор.

2.1 Поняття великих мовних моделей

Великі мовні моделі (ВММ) працюють на основі фреймворку, відомого як трансформатори, який був представлений у статті "Увага - це все, що вам потрібно"[7] у 2017 році. В основі їх архітектури лежать трансформатори та нейронні мережа прямого поширення. ВММ переважно будуються з використанням трансформаторної архітектури, яка спирається на механізм, відомий як самоуважність. Ця архітектура призначена для паралельної обробки послідовностей даних (наприклад, тексту), що є відходом від попередніх моделей обробки послідовностей, які обробляли дані крок за кроком (наприклад, RNN і LSTM). Механізм самоуважності дозволяє моделі зважувати важливість різних слів у реченні, незалежно від їхньої позиції. Для кожного слова модель обчислює, скільки уваги (або "фокусу") вона повинна приділити кожному іншому слову в реченні, створюючи оцінку уваги. Ці бали визначають, як кожне слово впливає на інші, що дозволяє моделі ефективно вловлювати контекст і зв'язки між словами. Трансформатори складаються з шарів, кожен з яких містить кілька "голів" у механізмах уваги. Кожна голова дивиться на різні частини речення, що дозволяє моделі розуміти різні аспекти тексту одночасно. Такий підхід до уваги забезпечує багатше представлення структури та семантики речення. Кожен шар трансформатора також включає в себе нейронні мережі прямого поширення, які застосовують подальші перетворення до даних. Ці мережі однакові для різних позицій, але мають різні параметри від шару до шару. Після навчання на широкому

наборі даних, LLM можуть добре узагальнювати дані для широкого спектру завдань. Однак їх можна додатково доопрацювати на конкретних завданнях або наборах даних, щоб покращити ефективність роботи з певними типами даних або проблем.

2.2 Поняття рішення гри та домінуючої стратегії

Термін "рішення гри" в теорії ігор означає результат(и) або стратегію(і), які раціональні гравці обрали б у грі, припускаючи, що всі гравці діють у власних інтересах і є повністю раціональними. Це поняття є фундаментальним у теорії ігор, яка вивчає стратегічні взаємодії, де результат для кожного учасника залежить від дій інших. Домінуюча стратегія - це стратегія, яка призводить до найбільшого виграшу для гравця, незалежно від того, яких стратегій дотримуються інші гравці. Якщо гравець має домінуючу стратегію, він обиратиме її в кожному можливому сценарії. Ігрове рішення з використанням домінантних стратегій є простим, оскільки воно передбачає передбачувані та стабільні результати.

РОЗДІЛ 3. ОПИС РІШЕННЯ, СТРУКТУРИ ПРОГРАМИ ТА ГЕНЕРАЦІЇ ДЕРЕВА СТРАТЕГІЙ

Для постановки експерименту було обрано настільну гру “Відважні щури”. “Відважні щури” – це стратегічна настільна гра для двох гравців, з одночасним сліпим вибором ходів. Кожен гравець має по 8 унікальних карт і кожен раунд в таємниці від іншого гравця обирає одну з них. Мета гри – здобути чотири перемоги у раундах, які виграються шляхом порівняння сил карт і застосування правил карт до поточного раунду. Гра складається з кількох раундів. На початку кожного раунду обидва гравці вибирають по одній карті зі своїх рук та кладуть її обличчям вниз перед собою. Після того, як обидва гравці вибрали свої карти, вони одночасно відкривають їх. Визначається переможець раунду залежно від значення карт та їх спеціальних властивостей. В грі присутні такі картки:

0: Музикант: Перемагає всі сили, крім 5 (Чарівник), поточний раунд грається в нічию.

1: Принцеса: Якщо ваш опонент грає 7 (принц), гра негайно закінчується, і ви виграєте, незалежно від кількості перемог в раундах

2: Шпигун: у наступному раунді ваш суперник відкриває свою карту, перш ніж ви виберете свою. Якщо обидва гравці вибирають 2 (Шпигун) в одному раунді, їхні сили анулюють один одного.

3: Вбивця: Перемагає найменша сила за винятком гри проти 7 (Принц).

4: Посол: якщо ви виграєте цією карткою, це зараховується як 2 перемоги. Цей ефект дійсний, навіть якщо раунд затримано.

5: Чарівник: Анулює особливу силу карти вашого опонента. це просто означає, що старша карта виграє раунд.

6: Капітан: карта у вашому наступному раунді отримує +2 сили.

7: Принц: Ви виграєте раунд. Ця карта завжди виграє в раунді, за винятком проти 1 (Принцеса) і 0 (Музикант).

Таким чином, BraveRats є динамічною грою, що поєднує прості правила з глибокою стратегічною взаємодією, де інтуїція та вміння передбачати дії супротивника є ключовими для перемоги.

3.1 Структура коду

Для функціонування системи необхідний імпорт низки бібліотек та модулів, які служать різним цілям: `random` для генерації випадкових чисел, `openai` і `OpenAI` для доступу до функціоналу OpenAI API, `time` для роботи з часом, `pandas` під аліасом `pd` для аналізу та обробки даних у табличній формі, `re` для використання регулярних виразів у текстових операціях, та `os` для взаємодії з операційною системою та її компонентами, зокрема файловою системою. Ці імпорти забезпечують необхідні інструменти для розгорнутого програмування в різних сферах обробки даних, маніпуляцій з текстом, і взаємодії з ОС.

Константні параметри включають системний запит, який детально описує правила гри, властивості карт, їхні сили та спеціальні здібності, які впливають на хід гри. Також присутня `result_matrix`, що містить результати раундів для кожної можливої пари карт, символізуючи перемогу, поразку, нічию чи утримання. Крім того, визначені `win_strategy_list` та `win_strategy_list_6`, що містять стратегії для оптимізації результатів гри, де кожен список вказує, які карти слід вибирати для максимізації шансів на перемогу.

Функція `format_card` призначена для форматування інформації про карту в грі. Вона приймає один параметр `card_1`, який спочатку перетворюється з текстового формату в ціле число. У функції визначено список `card_to_number_mapping`, що містить назви карт відповідно до їх індексів. Функція повертає рядок, який вказує на номер та назву карти, використовуючи індекс для пошуку у списку. Це дозволяє легко

ідентифікувати карту за номером у контексті гри.

Функція `format_user_message` структурує інформаційне повідомлення для гравця, використовуючи дані про історію гри, доступні карти та особливу карту шпигуна. Вона ініціюється з пустим рядком `game_format` та далі заповнюється деталями кожного раунду з історії гри. Для кожного раунду вона додає номер раунду, карти, які були в грі, та переможця цього раунду, використовуючи умовні конструкції для визначення, чи переміг гравець, чи було нічию, чи переміг опонент. Якщо у попередньому раунді була зіграна карта шпигуна, функція додає інформацію про карту опонента у поточному раунді. На завершення, вона виводить список доступних карт, які можна використати для гри, та закликає гравця вибрати найкращу карту для наступного ходу, наголошуючи, що відповідь має бути у формі номеру сили карти.

Функція `call_gemini` призначена для взаємодії з генеративною моделлю AI від Google. Вона спочатку імпортує необхідні бібліотеки та конфігурує API, використовуючи конкретний ключ API. Потім функція створює об'єкт моделі, налаштовуючи параметри генерації відповідей, такі як температура, `top_p`, `top_k`, і максимальна кількість токенів для виводу. Після цього ініціюється сесія чату з моделлю, і через неї відправляється повідомлення користувача. Відповідь моделі отримується і повертається як текст, що дозволяє інтегрувати розуміння та обробку природної мови в будь-яку програму, яка використовує цю функцію.

Функція `call_open_ai` використовується для здійснення запиту до API OpenAI, зокрема до моделі GPT-4 Turbo. Після імпорту модуля `openai` і класу `OpenAI`, створюється клієнт з використанням специфічного API ключа. Далі, здійснюється запит до моделі, де задаються параметри, такі як модель, максимальна кількість токенів відповіді та історія повідомлень, включаючи системні інструкції та повідомлення користувача. Результатом є текст відповіді, отриманий з першої вибраної відповіді моделі, який

потім повертається як результат функції. Це дозволяє інтегрувати високоефективні можливості генерації тексту GPT-4 в програми, що використовують природню мову для взаємодії з користувачами.

Клас Game містить кілька методів, кожен з яких відіграє ключову роль у механіці гри. Ось детальний опис кожного з цих методів:

- `__init__`: Ініціалізує екземпляр класу і викликає метод `reset` для встановлення початкових умов гри, таких як обнулення рахунків і встановлення доступних карт.
- `reset`: Встановлює початкові значення для гравців, такі як рахунок, доступні карти та інші змінні, що контролюють стан гри. Також ініціалізує стратегії перемоги в залежності від обмеження карт (`cards_limit`).
- `limit_game`: Змінює доступні карти та умови гри в залежності від параметра `cards_limit`. Це включає вилучення певних карт з гри та зміну необхідної кількості перемог для завершення гри.
- `get_win_strategy`: Визначає стратегію перемоги для гравця на основі поточного стану гри та історії ігор, використовуючи вже встановлені стратегії відповідно до `cards_limit`.
- `get_predefined_player`: Повертає вибір картки для гравця на основі заздалегідь визначеного порядку ходів. Це допомагає симулювати гру з попередньо встановленим набором ходів.
- `get_player_choice`: Вибирає карту випадково для гравця на основі доступних карт, що може використовуватися для симуляції випадкової гри.
- `get_gpt_choice`: Використовує зовнішні AI сервіси (такі як OpenAI або Gemini), щоб отримати раціональний вибір картки на основі поточного стану гри, включаючи доступні картки та інші параметри.

- `spy_table_choice`: Вибирає найкращий можливий хід для гравця, враховуючи картку, вибрану суперником. Цей метод використовує спеціальні здібності картки "шпигун", які дозволяють передбачити ходи супротивника.
- `play_round`: Виконує один раунд гри, визначаючи переможця на основі сил карток і спеціальних правил для карт, таких як асасин, генерал і інші. Цей метод також враховує бонуси та особливості карт, що впливають на результат раунду.
- `get_player_by_name`: Вибирає відповідну функцію для генерації ходу на основі імені гравця, що дозволяє гнучко взаємодіяти з різними стратегіями гравців або алгоритмами управління грою залежно від їх ролі або налаштувань.
- `play_game`: Організовує і контролює всю гру, керуючи раундами, вибором карт, і визначаючи кінцевий результат. Цей метод враховує статус шпигунських карт та інші спеціальні здібності, здійснює перевірку на можливі помилки та обробляє логіку перемоги або нічиї. Цей метод також забезпечує виведення історії гри, рахунків і стану збереження раундів на нічию.
- `play_game_on_list`: Це спеціалізований метод для гри за визначеним списком ходів, що дозволяє тестувати певні стратегії або розігравати сценарії за заздалегідь заданими ходами. Він використовує ту саму основну логіку, що й `play_game`, але замість динамічного визначення ходів, ходи вже заздалегідь визначені в списку, що вимагає меншої інтеракції з гравцями та зосередження на виконанні алгоритмічно заданих дій.

Функція `compare_tuples` призначена для порівняння двох кортежів: `win_comb` та `first_elements`. Вона перевіряє, чи один кортеж є частиною іншого або збігається з ним повністю. Якщо довжина `first_elements` менша або дорівнює довжині `win_comb`, функція перевіряє, чи збігаються початкові елементи `first_elements` з відповідною частиною `win_comb`. Якщо `first_elements` довший, то вона порівнює `win_comb` з відповідною частиною `first_elements` і перевіряє, чи довжина `first_elements` на одиницю більша за довжину `win_comb`. Це дозволяє визначити, чи є `first_elements` розширенням `win_comb` на один додатковий елемент.

Клас `Game_Solution` є частиною системи для аналізу і генерації стратегій перемоги у грі на основі комбінацій карт.

Конструктор класу ініціалізує декілька ключових атрибутів. Він створює словник `dict_states` для збереження станів гри, `combination_stat_composed` для збереження статистики комбінацій, і `numbers_available` як список доступних номерів карт. Якщо вказано обмеження карт (`card_limit`), викликається метод `limit_game` для видалення певних карт. Після цього генеруються всі можливі пари карт, які зберігаються у стеку `stack` для подальшої обробки.

`limit_game` видаляє певні карти з доступних на основі заданого ліміту. Кожен рівень ліміту визначає, які карти будуть вилучені з гри — це робиться поетапно для шпигуна, амбасадора, генерала, мага та музиканта.

`generate_dict` генерує словник станів гри, використовуючи симуляцію гри для кожного стеку комбінацій карт. Він викликає `reset` та `play_game_on_list` на екземплярі `Game` для кожної комбінації карт. Якщо результат гри дозволяє додаткові ходи (наприклад, коли гра не завершена), створюються нові комбінації і додаються до стеку. У іншому випадку, поточна комбінація і її результат зберігаються у `dict_states`.

`win_strategies` аналізує виграшні стратегії, використовуючи перmutації доступних карт. Він розраховує статистику для кожної

комбінації карт, включаючи кількість перемог, поразок і нічиїх. Статистика акумулюється для кожного сегмента комбінацій, а кінцеві відсотки виграшів розраховуються та виводяться для стратегій, які показують високий відсоток успіху.

`show_win_strategies` виводить стратегії перемоги, які перевищують заданий відсоток успіху. Це дозволяє ідентифікувати найбільш ефективні комбінації карт для використання в реальній грі.

Клас `Game_Solution` служить потужним інструментом для аналізу і розробки стратегій гри, що може бути використаний для тренувань або для покращення автоматизованих гравців у складних ігрових симуляції

Для реалізації даних правил було створено клас гри `Game`, в якому зберігаються усі важливі параметри ходу гри.

Для зручності реалізації було створено клас гри який може зберігати поточний стан гри та опрацьовувати вибори передані як параметр при ініціалізації.

Екземпляр класу ініціалізується трьома параметрами, перший з яких є унікальною назвою гри, що дозволяє зберегти її перебіг до словнику за переглянути перебіг кожної гри окремо, два інші відповідають за імена гравців. При ініціалізації імена гравців обробляються і якщо в них є ключові назви агентів такі як `'win'`, `'gemini'`, `'gpt'` - будуть використані відповідні функції агентів що будуть грати в гру. Так з реалізованих на даний момент агентів є `gpt`, `gemini`, високо ймовірно виграшна стратегія та випадковий агент. Для подальшого аналізу буде використовуватись варіант гри між одним з варіантів великих мовних моделей та високоефективної стратегії.

Для порівняння агентів великих мовних моделей потрібно встановити еталон гри, проти якого вони будуть вимірюватись. В концепції такої

роботи еталоном гри мало стати рішення гри - така послідовність ходів за якої будь-який вибір іншого гравця призводить до програшу або нічиєї. Загалом за приблизними оцінками гра “Відважні щури” має дещо менше 1 625 702 400 можливих перебігів гри, тому було вирішено для прорахунку рішення гри скоротити гру та вилучити з початкових колод карту “шпигуна”. Для такої обмеженої колоди був побудований словник всіх можливих комбінацій ходів та їх вирішення. Надалі ці ходи були скомпоновані по ходам окремого гравця для обробки статистики. Таким чином було отримано 10 найкращих стратегій з наступними результатами:

Стратегія	Перемоги	Програші	Нічії	Успішність
(0, 7, 1, 4, 6, 5, 3)	355	61	995	0.9567
(3, 0, 7, 1, 4, 5, 6)	717	87	1293	0.9585
(3, 0, 7, 1, 6, 5, 4)	538	60	1084	0.9643
(3, 7, 0, 1, 4, 5, 6)	518	79	1363	0.9596
(3, 7, 0, 1, 6, 5, 4)	419	55	1186	0.9668
(7, 0, 1, 4, 6, 5, 3)	402	56	1184	0.9658
(7, 0, 3, 1, 5, 4, 6)	723	93	1341	0.9568
(7, 3, 0, 1, 6, 5, 4)	407	61	1082	0.960
(7, 3, 0, 1, 4, 5, 6)	555	77	1148	0.9567

Таблиця 3.1 Статистично виграшні стратегії для гри на семи картках
Клас гри на початку ініціалізації випадковим чином обирає одну згенерованих переможних стратегій відповідно до кількості карток та грає за нею, при визначенні одного з гравців як еталон.

Для подання правильного контексту гри, та правил гри “Відважні щури” було сформульоване наступне уніфіковане повідомлення, яке подається на кожну взаємодію з великою мовною моделлю.

""

You are an expert strategic tabletop game player.

Game is called "brave rats".

Strictly follow the rules:

- At the beginning of the game You will have eight available cards.
- Each game round you have to give a card from one that is left available and not played by YOU already.
- Game will last until 4 wins or no cards left to choose from.
- During the round you have to choose EXACTLY one card to win a round and optimize a winning strategy.
- Each card has a STRENGTH and a special power that determine who wins the round.
- The card with the HIGHEST STRENGTH wins, but some special powers can overrule this.
- The special powers are mandatory and always count.
- In the case of a draw (the strength of both cards is equal), the round is put on HOLD. The winner of the next round also wins the previous round that was on HOLD (it may happen that several rounds in a row are on hold) that is dramatically affecting your win rate.

All cards given at the start of the game:

- STRENGTH: 0 CARD: Musician POWER: Wins over all powers, except the 5 (Wizard).
- STRENGTH: 1 CARD: Princess POWER: If your opponent plays the 7 (Prince), the game ends immediately and you win, regardless of the number of victories. However, the 0 (Musician) still puts the round on hold.
- STRENGTH: 2 CARD: Spy POWER: Next round, your opponent reveals his card before you choose yours. If both players choose the 2 (Spy) the same round, their powers nullify each other.
- STRENGTH: 3 CARD: Assassin POWER: The lowest strength wins The lowest strength wins, except against a 7

(Prince).

- STRENGTH: 4 CARD: Ambassador POWER: If you win with this card, it counts as 2 victories. This effect is valid even if the round is on hold. If the player who played the 5 wins the round following a hold, it's worth 3 victories.

- STRENGTH: 5 CARD: Wizard POWER: Nullifies the special power of your opponent's card. which simply means that the highest card wins the round. Ex: 5 wins 4, 7 wins 0

- STRENGTH: 6 CARD: General POWER: The card in your next round gets +2 strength. The 6's power affects the next round. If the opponent plays the 5 (Wizard) in the next round, it does not nullify the 6 - (General) power.

- STRENGTH: 7 CARD: Prince POWER: You win the round. This card always wins the round except against the 1 (Princess) and the 0 (Musician). It doesn't matter if the 3 (Assassin) has turned the tables, the 7 (Prince) beats the 3 (Assassin). However, the 0 (Musician) still puts the round on hold. Warning: if the opponent plays the 1 (Princess) against your 7 (Prince), he doesn't just win the round, he wins the game!

You will be given current game statistics in the format:

GAME HISTORY:

ROUND 1, YOU: STRENGTH - (CARD NAME), OPPONENT: STRENGTH - (CARD NAME). {YOU/OPPONENT} WINS or HOLD

think step by step how to win the whole game and not a single round.

ALWAYS REPLY WITH THE card number/STRENGTH, never and under any circumstances don't output any other text except card number that left.

""

Рисунок 3.1 Описание системного запроса

В подібному повідомленні зазначені контекст гри, а саме параметри модифікації поточного раунда, кількість виграшних очок кожного з гравців, їх попередні вибори та доступні на момент передання повідомлення карти. Відповідь ВММ перевіряється на достовірність, зводиться до цифрового варіанту обраної карти та передається в систему. Якщо обрана карта не може бути зіграна, повідомлення модифікується зазначаючи помилку і надсилається рекурсивним викликом функції вибору. В ході реалізації для проведення експериментів були інтегровані і протестовані три великі мовні моделі, які мають API та доступні до інтеграції на даний момент. Було протестовано chat-GPT різних версій, Gemini AI та LLAMA 2.

Кожна окрема гра логується і виводиться в консоль для спостереження за результатом та послідовністю ходів супротивників як зображено на рисунку 3.2

```
Win strategy this game: [3, 7, 0, 1, 4, 5, 6]
Game Started: 16
TURN: 1
Player: gpt-3.5-turbo Request: Your ONLY Available cards
left: CARD 0 - musician, CARD 1 - princess, CARD 3 -
assassin, CARD 4 - ambassador, CARD 5 - wizard, CARD 6 -
general, CARD 7 - prince
Your turn to give the best strategic card from one of the
available. Remember to answer only as card
strength/number
gpt-3.5-turbo answer: 1 . CARD 1 - princess
Win strategy : [3, 7, 0, 1, 4, 5, 6] this turn played: 3
TURN: 1 Player: gpt-3.5-turbo CARD 1 - princess
Player: win 3 : CARD 3 - assassin Result: 2
TURN: 1 Scores A:0 Y:1 Hold A:0 Hold Y:0
TURN: 2
Player: gpt-3.5-turbo Request: GAME HISTORY:
ROUND 1, YOUR's CARD 1 - princess, OPPONENT's CARD 3 -
```

```

assassin. OPPONENT WIN
Your ONLY Available cards left: CARD 0 - musician, CARD 3 -
assassin, CARD 4 - ambassador, CARD 5 - wizard, CARD 6 -
general, CARD 7 - prince
Your turn to give the best strategic card from one of the
available. Remember to answer only as card
strength/number
gpt-3.5-turbo answer: 6 . CARD 6 - general
Win strategy : [3, 7, 0, 1, 4, 5, 6] this turn played: 7
TURN: 2 Player: gpt-3.5-turbo CARD 6 - general
Player: win 7 : CARD 7 - prince Result: 1
TURN: 2 Scores A:1 Y:1 Hold A:0 Hold Y:0
TURN: 3
Player: gpt-3.5-turbo Request: GAME HISTORY:
ROUND 1, YOUR's CARD 1 - princess, OPPONENT's CARD 3 -
assassin. OPPONENT WIN
ROUND 2, YOUR's CARD 6 - general, OPPONENT's CARD 7 -
prince. YOU WIN
Your ONLY Available cards left: CARD 0 - musician, CARD 3 -
assassin, CARD 4 - ambassador, CARD 5 - wizard, CARD 7 -
prince
Your turn to give the best strategic card from one of the
available. Remember to answer only as card
strength/number
gpt-3.5-turbo answer: 4 . CARD 4 - ambassador
Win strategy : [3, 7, 0, 1, 4, 5, 6] this turn played: 0
TURN: 3 Player: gpt-3.5-turbo CARD 4 - ambassador
Player: win 0 : CARD 0 - musician Result: 0
TURN: 3 Scores A:1 Y:1 Hold A:1 Hold Y:1
TURN: 4
Player: gpt-3.5-turbo Request: GAME HISTORY:
ROUND 1, YOUR's CARD 1 - princess, OPPONENT's CARD 3 -
assassin. OPPONENT WIN
ROUND 2, YOUR's CARD 6 - general, OPPONENT's CARD 7 -
prince. YOU WIN
ROUND 3, YOUR's CARD 4 - ambassador, OPPONENT's CARD 0 -

```

```

musician. HOLD
Your ONLY Available cards left: CARD 0 - musician, CARD 3 -
assassin, CARD 5 - wizard, CARD 7 - prince
Your turn to give the best strategic card from one of the
available. Remember to answer only as card
strength/number
gpt-3.5-turbo answer: 7 . CARD 7 - prince
Win strategy : [3, 7, 0, 1, 4, 5, 6] this turn played: 1
TURN: 4 Player: gpt-3.5-turbo CARD 7 - prince Player:
win 1 : CARD 1 - princess Result: 4
TURN: 4 Scores A:11 Y:1 Hold A:1 Hold Y:1
Game Ended: 16 Result: 1

```

Рисунок 3.2 Приклад логування гри

В ході експериментів для семи карт було зібрано наступні дані:

Модель	Виграші	Програші	Нічії	Частка не програшу
gpt-3.5-turbo	48	11	41	0.89
gpt-4-turbo	3	29	68	0.71
gemini-1.5	13	15	72	0.85

Таблиця 3.2 Результати гри з сімома картками

Такі результати були отримані для гри на 7 карток, де виграшні стратегії мають статистичні недоліки та показують близько 20.8 % виграшів проти переможних стратегій. На цих даних можемо побачити очевидну перевагу gpt-3.5-turbo. В ході дослідження контексту моделей було виявлено, що ця модель не мала знань про гру “Відважні щури” поза контекстом повідомлення в прикладному програмному інтерфейсі, коли дві інші тестовані моделі можуть неточно роз’яснити правила гри без попереднього їх описання.

Для більш коректної генерації вигравних стратегій та дослідження точності моделі, були проведені експерименти з вилученням додаткових карт для зменшення кількості можливих комбінацій. Подібні експерименти були проведені для шести та п'яти карт.

Модель	Виграші	Програші	Нічії	Частка не програшу
gpt-3.5-turbo	0	94	6	0.06
gemini-1.5	13	14	73	0.86

Таблиця 3.3 Результати гри з шістьма картками

Для шістьох карток модель gemini-1.5 показала серйозно значущіший результат аніж gpt-3.5-turbo. За вигравних стратегій було обрано статистично наймовірніші комбінації виграшу для шестикартового варіанту.

Стратегія	Виграші	Програші	Нічия	Відсоток не програшу
(7, 0, 1, 6, 5, 3)	41	1	0	97
(7, 0, 3, 6, 5, 1)	78	2	0	97,5
(7, 0, 6, 5, 3, 1)	82	3	0	96,4
(7, 3, 0, 6, 5, 1)	63	2	0	96,9
(7, 6, 5, 0, 3, 1)	85	2	0	97,7
(7, 6, 0, 5, 3, 1)	121	5	0	96

Таблиця 3.4 Успішність стратегій виграшу для гри з шести карт

У всіх цих стратегій є спільний недолік з використанням картки принца в першому раунді. Такий хід дає змогу супротивнику обрати карту принцеси та закінчити гру в першому раунді. Передаючи логовані останні зіграні ігри в повідомленні до великої мовної моделі, вона могла відслідкувати патерн початку гри з картки номер сім та відштовхуватись від вже відомої

стратегії супротивника, частіше використовуючи перевагу на першому ході.

Модель	Виграші	Програші	Нічії	Частка не програшу
gpt-3.5-turbo	0	98	2	0.02
gemini-1.5	1	99	0	0.01

Таблиця 3.5 Результати гри з п'ятьма картками

В грі з п'ятьма картками ситуація з використанням сімки як першої картки в виграшних стратегіях зберігається, проте при генеруванні ігор проти мовних моделей жодна з використаних ВММ не виявила патерну та не показала значущих результатів.

РОЗДІЛ 4. ВИСНОВКИ

В рамках даної наукової роботи було поставлено мету дослідити застосування великих мовних моделей до розробки стратегій в настільних іграх, дослідити можливість використання розуміння правил настільних ігор та можливість вибудовувати стратегію в їх контексті як тест продуктивності для мовних моделей та побудувати середовище для цього. Для досягнення цих цілей було сформульовано дослідницьке завдання, що включили в себе огляд сучасної літератури, дослідження вибудовування коректних запитів, створення середовища симуляції гри, інтегрування прикладних програмних інтерфейсів сучасних мовних моделей, ознайомлення з їх налаштуванням та дослідження способів інтеграції мовних моделей в ігрове середовище. Дослідження було проведено з використанням широкого спектру джерел, включаючи електронні ресурси, друковану літературу, матеріали конференцій, а також програмні коди та бібліотеки. Такий підхід дозволив ефективно збирати актуальну інформацію, необхідну для глибокого аналізу та успішного досягнення цілей дослідження.

У результаті проведених досліджень була розроблена програмна система для оцінювання можливостей мовних моделей до розробки стратегій в межах карткової гри з одночасними ходами “Відважні щури”. Така система дозволяє уніфіковано надавати матеріал правил гри та отримувати уніфіковану відповідь від великих мовних моделей та обробляти їх відповіді в симуляції, надавати моделям зворотній зв’язок з інформацією ходу гри та акумулювати результати ігор для статистичної обробки та оцінки моделі на різних рівнях об’єму контексту правил. Розроблені методи та система можуть бути успішно використані в реальних умовах оцінювання моделей та сприяти розвитку навчання великих мовних моделей у напрямку розуміння стратегічної глибини настільних ігор та

вибудовування універсального агента-гравця для різноманітних платформ з симуляціями ігор. Враховуючи всі отримані дані, можна зробити висновок, що дослідження відкриває значні можливості для його подальшого розвитку та удосконалення. Дане системне рішення надає зручну інфраструктуру для збереження результатів експериментів, тестування моделей, дослідження поведінки моделей та має можливість отримання чіткого обґрунтування кожного ходу гри, для легшого розуміння і навчальних цілей. Підхід реалізації даної системи також легко супроводжується та розвивається і забезпечує гнучкість внесення змін в систему та можливість імплементування інших ігор.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

- [1] Hogan D. P., Brennen A. Open-Ended Wargames with Large Language Models. arxiv.org. URL: <https://arxiv.org/pdf/2404.11446v1>.
- [2] Gallotta R., Todd G. Large Language Models and Games: A Survey and Roadmap. arXiv.org e-Print archive. URL: <https://arxiv.org/html/2402.18659v1>.
- [3] Tian Liang, Zhiwei He. Leveraging Word Guessing Games to Assess the Intelligence of Large Language Models. arXiv.org. URL: <https://arxiv.org/abs/2310.20499>.
- [4] Yuzhuang Xu, Shuo Wang. Exploring Large Language Models for Communication Games: An Empirical Study on Werewolf. arXiv.org. URL: <https://arxiv.org/abs/2309.04658>.
- [5] Zelai Xu, Chao Yu. Language Agents with Reinforcement Learning for Strategic Play in the Werewolf Game. arXiv.org. URL: <https://arxiv.org/abs/2310.18940>.
- [6] Elif Akata, Lion Schulz. Playing repeated games with Large Language Models. arXiv.org. URL: <https://arxiv.org/abs/2305.16867>.
- [7] Think you have Solved Question Answering? Try ARC, the AI2 Reasoning Challenge [Електронний ресурс] / [P. Clark, I. Cowhey, O. Etzioni та ін.]. – 2018. – Режим доступу до ресурсу: <https://arxiv.org/abs/1803.05457>.