Методичні рекомендації
до виконання практичної роботи №2

"Основи промпт-інженерії для моделей машинного навчання"

Мета роботи
- Освоїти базові принципи токенізації тексту
- Навчитися працювати з API GitHub Models
- Розвинути навички створення ефективних промптів

Обладнання
- Комп'ютер з доступом до інтернету
- Встановлений Python 3.x
- Jupyter Notebook
- GitHub аккаунт

Порядок виконання роботи
1. **Підготовка середовища:**
- Форкніть репозиторій
- Налаштуйте GitHub Codespaces
- Створіть файл `.env` з необхідними ключами
2. **Виконання базових вправ:**
- Запустіть всі комірки наданого ноутбука
- Проаналізуйте результати токенізації
- Протестуйте з'єднання з API
3. **Індивідуальне завдання:**
Виконайте завдання згідно вашого варіанту з наведеного нижче списку.
Результати роботи збережіть у форматі Jupyter Notebook.

Вимоги до звіту
1. Титульна сторінка
2. Мета роботи
3. Хід виконання:
   - Скріншоти результатів базових вправ
   - Лістинг створених промптів
   - Аналіз токенізації
   - Результати виконання індивідуального завдання
4. Висновки
5. Завантаження результатів роботи у форматі Jupyter Notebook у Moodle з обов’язковим індивідуальним захистом.

Критерії оцінювання

- Виконання базових вправ
- Створення ефективних промптів
- Аналіз токенізації
- Виконання індивідуального завдання
- Якість оформлення звіту
- Завантаження результатів роботи у Moodle та успішний індивідуальний захист.

Контрольні запитання

1. Що таке токенізація і яке її призначення?
2. Які основні принципи створення ефективних промптів?
3. Як системний контекст впливає на відповіді моделі?
4. Які переваги та недоліки різних підходів до промпт-інженерії?
5. Як оцінити якість згенерованого моделлю коду?

Варіанти індивідуальних завдань

1. Створіть промпт для генерації алгоритмів сортування (bubble, quick, merge sort) з оцінкою складності. Порівняйте токенізацію.
2. Розробіть промпт для оптимізації SQL-запитів через системний контекст DBA-експерта.
3. Створіть промпт для API-специфікації системи моніторингу серверного обладнання.
4. Розробіть промпт для пояснення роботи конвеєра процесора з chain-of-thought підходом.
5. Створіть промпт для аналізу конфігурацій мережевого обладнання на предмет безпеки.
6. Розробіть промпт для генерації тестів периферійних пристроїв з порівнянням форматів.
7. Створіть промпт для оптимізації асемблерного коду з few-shot прикладами.
8. Розробіть промпт для документації драйвера USB-пристрою з різними стилями.
9. Створіть промпт для аналізу дампів пам'яті x86/ARM процесорів.
10. Розробіть промпт для генерації схем на Verilog/VHDL з порівнянням стилів.
11. Створіть промпт для аналізу та оптимізації мережевих протоколів IoT пристроїв.
12. Розробіть промпт для генерації тестів проникнення для вбудованих систем.
13. Створіть промпт для аналізу та рефакторингу коду FPGA-проектів.
14. Розробіть промпт для діагностики та налагодження RTOS систем.
15. Створіть промпт для оптимізації енергоспоживання вбудованих систем.

16. Розробіть промпт для аналізу та покращення надійності кеш-пам'яті.
17. Створіть промпт для генерації драйверів GPIO з різними інтерфейсами.
18. Розробіть промпт для оптимізації паралельних обчислень на CUDA.
19. Створіть промпт для аналізу та покращення QoS в реальному часі.
20. Розробіть промпт для генерації специфікацій SoC з IP-блоками.

Для кожного варіанту необхідно:
1. Створити базовий промпт
2. Додати системний контекст
3. Провести токенізацію варіантів
4. Оцінити якість результатів
5. Запропонувати покращення