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

Використання великих мовних моделей на мобільній платформі для генерації медіа

Текстова частина до курсової роботи за спеціальністю
«Комп'ютерні науки» - 122

Керівник курсової роботи

к.т.н. ст. викл. Франків О. О.

_____

(*підпис*)

"____" _____ 2024 року

Виконала студентка КН-4

Іваненко В. А.

"____" _____ 2024 року

Київ - 2024

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

ЗАТВЕРДЖУЮ
Зав.кафедри інформатики,
доцент, к.ф-м.н.
_____ С. С. Гороховський (підпис)
„____”_____2024 р.

ІНДИВІДУАЛЬНЕ ЗАВДАННЯ
на курсову роботу
студентки Іваненко Валерії Анатоліївни факультету інформатики 4-го курсу
ТЕМА: Використання великих мовних моделей на мобільній платформі для генерації медіа

Зміст ТЧ до курсової роботи: Індивідуальне завдання

Вступ

Частина 1: Теоретичні відомості про великі мовні моделі

Частина 2: Генерація медіа на мобільні платформі за допомогою великих мовних моделей

Частина 3: Реалізація розширення клавіатури з використанням великої мовної моделі

Висновки

Список літератури

Дата видачі „___” _____ 2024 р. Франків О.О.
_____(підпис)

Завдання отримала _____

(підпис)

Календарний план виконання роботи:

Тема: **Використання великих мовних моделей на мобільній платформі для генерації медіа**

No	Назва етапу	Термін виконання	Примітка
1	Отримання завдання на курсову роботу		
2	Пошук тематичної літератури		
3	Огляд літератури за темою роботи		
4	Проведення дослідження		
5	Аналіз отриманих результатів		
6	Імплементація розширення клавіатури		
7	Написання текстової частини курсової роботи		

8	Здача курсової роботи		

Студентка Іваненко В.А.

Керівник Франків О. О.

“____” _____

ЗМІСТ

Анотація

У роботі досліджуються особливості інтеграції великих мовних моделей (LLM) на мобільній платформі iOS, з акцентом на генерації медіа, в середовищі з обмеженими ресурсами. Дослідження проводиться за рахунок розробки розширення клавіатури iOS, яке дозволяє генерувати текст на пристрої за допомогою великої мовної моделі, підвищуючи у такий спосіб конфіденційність, зменшуючи затримки та забезпечуючи офлайн-функціональність.

Ключові слова: LLM, iOS, генерація медіа, розширення клавіатури

Вступ

Стрімка еволюція штучного інтелекту (ШІ) призвела до трансформаційних змін у різних галузях, а великі мовні моделі (Large language models, LMMs), такі як GPT-3 (OpenAI) та її нащадки [1], LaMDA (Google AI) [2], та інші показують безпрецедентні можливості в обробці природної мови (NLP). Ці моделі продемонстрували неабияку майстерність у виконанні таких завдань, як генерація текстів, переклад та саммаризація (узагальнення), що робить їх безцінними інструментами в багатьох галузях. Однак типове розгортання цих моделей через хмарні API створює значні проблеми, включаючи затримку, проблеми з конфіденційністю та залежність від підключення до Інтернету. [3]

Проте, використання великої мовної моделі безпосередньо на пристрої теж не позбавлене викликів, головний з яких - обмежені ресурси, особливо, якщо йдеться про мобільну платформу. Тим часом, кількість користувачів мобільних пристроїв постійно зростає і очікується, що до 2029 року число користувачів смартфонів досягне 6.38 мільярдів, що на 30% більше, ніж сьогодні (4.88 млрд). [4] Природно, що попит на ефективні та безпечні додатки з використанням штучного інтелекту також збільшується. Так, прогнозується, що сектор додатків зі штучним інтелектом зросте з $2,5 млрд у 2022 році до $38,5 млрд у 2028 році в міру того, як технологія розвиватиметься і все більше компаній переходитимуть на процеси зі штучним інтелектом. [5]

Виходячи з даної інформації про потреби користувачів, у цій дипломній роботі досліджується підхід інтеграції LLM безпосередньо на мобільній платформі, зокрема через розробку розширення клавіатури на iOS, для забезпечення конфіденційності.

Наукове та практичне значення роботи полягає у важливості дослідження підходів до ефективного використання мобільним додатком ресурсів пристрою під час роботи з великою мовною моделлю, особливо враховуючи додаткові обмеження, яких має дотримуватись розширення клавіатури на iOS.

Основною метою цього дослідження є проведення дослідження способів ефективного використання даних пристрою під час роботи з великою мовною моделлю на основі розробки розширення клавіатури на базі операційної системи iOS. Конкретні завдання включають:

1. Огляд теоретичних основ і поточного стану LLM та їх застосування.
2. Виявлення та вирішення проблем, пов'язаних з роботою з LLM на пристрої.
3. Розробка та реалізація розширення клавіатури на iOS, яке використовує LLM на пристрої для генерації тексту.

Курсова робота складається з трьох розділів.

У першому розділі описаний огляд LLM як таких, технологій, що лежать в основі, а також можливостей, що вони несуть. Окрім того, зазначені переваги та обмеження використання LLM..

У другому розділі досліджуються виклики та переваги роботи з LLM на пристрої порівняно з хмарними рішеннями. Розглядаються обмеження, що несе з собою робота з розширенням клавіатури, а також описані технології, які можуть бути використані для подолання викликів.

У третьому розділі детально описано розробку та реалізацію розширення клавіатури на платформі iOS.

Розділ 1. Теоретичні відомості про великі мовні моделі

1.1 Великі мовні моделі та технології, що лежать в їх основі

Великі мовні моделі (LMM) є важливою частиною в галузі обробки природної мови (NLP), що дозволяє комп'ютерам розуміти і генерувати текст, подібний до людського, з неабиякою майстерністю. Ці моделі використовують передові методи машинного навчання, зокрема архітектури глибокого навчання, для обробки та генерації тексту в безпрецедентних масштабах.

В основі LLM лежать нейронні мережі, зокрема архітектури на основі трансформерів. Трансформери, запропоновані Васвані та ін. (2017), зробили революцію в NLP, дозволивши моделям фіксувати довгострокові залежності в тексті за допомогою механізмів самоуваги. [6] Ця архітектура лежить в основі багатьох сучасних LMM, включаючи серію GPT (Generative Pre-trained Transformer) від OpenAI, BERT (Bidirectional Encoder Representations from Transformers) від Google і RoBERTa (Robustly optimized BERT approach) від Facebook. [7]

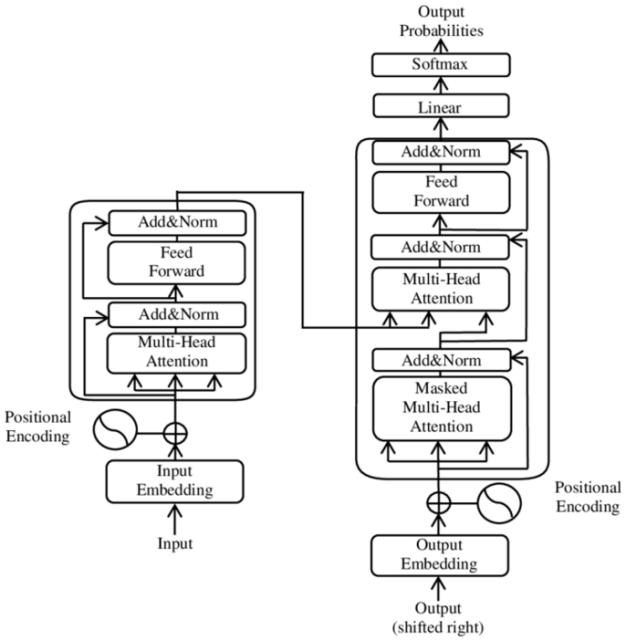

Рисунок 1.1.1 Основні складові трансформера

Успіх LLM можна пояснити кількома ключовими технологічними досягненнями:

- Трансформери: трансформер слугує основою для LLM, дозволяючи ефективно обробляти великі обсяги текстових даних. Механізм самонавчання дозволяє моделі зважувати важливість кожного слова в контексті всієї вхідної послідовності, фіксуючи складні лінгвістичні закономірності та залежності.
- Попереднє навчання і точне налаштування: LLM зазвичай попередньо навчаються на великих обсягах текстових даних за допомогою методів навчання без нагляду. Під час попереднього навчання модель вчиться передбачати наступне слово в послідовності з урахуванням контексту, ефективно використовуючи статистичні властивості природної мови. Точне налаштування додатково адаптує попередньо навчену модель до конкретних подальших завдань, таких як генерація тексту, переклад або узагальнення, шляхом оновлення параметрів моделі на основі мічених даних для конкретного завдання. [8]
- Масштаб і ефективність роботи з даними: останні досягнення в апаратній інфраструктурі, зокрема графічних процесорів (GPU) і тензорних процесорів (TPU), уможливили навчання дедалі більших і складніших LLM. Тепер можна ефективно навчати моделі з мільярдами і навіть трильйонами параметрів, що призводить до значного покращення якості та різноманітності генерації текстів. [9]
- Трансферне навчання: LLM демонструють потужні можливості трансферного навчання, коли знання, отримані під час попереднього навчання на великому корпусі текстів, можуть бути перенесені для ефективного виконання широкого спектру наступних завдань з мінімальними навчальними даними. Ця парадигма трансферного навчання демократизувала доступ до найсучасніших можливостей

NLP, дозволяючи розробникам досягати вражаючих результатів з мінімальними зусиллями. [10]

Таким чином, LLM є кульмінацією досягнень у галузі глибокого навчання, архітектури трансформерів та масштабованої обчислювальної інфраструктури. Ці моделі революціонізували завдання NLP і стали незамінними інструментами в різних сферах, від генерації контенту і перекладу мов до відповідей на питання і аналізу настроїв.

1.2 Дослідження можливостей великих мовних моделей

Великі мовні моделі продемонстрували чудові результати під час виконання різноманітних задач обробки природної мови. У цьому розділі розглядаються різноманітні можливості застосування LMM, висвітлюється їхня продуктивність, конкретні випадки використання та кількісні досягнення.

Генерація тексту

Один з найвідоміших прикладів застосування LLM - генерування тексту. Такі моделі, як OpenAI GPT-3, що має 175 мільярдів параметрів, можуть генерувати зв'язний і контекстуально релевантний текст на основі заданого запиту.. Ця здатність використовується у творчому письмі, автоматизованому створенні контенту і навіть у діалогових системах (чат-ботах). Наприклад, GPT-3 може створювати статті, оповідання та вірші, які часто неможливо відрізнити від написаних людиною. [9]

Переклад

LLM також значно вдосконалили машинний переклад. Модель BERT від Google з її двонаправленим розумінням тексту підвищила точність перекладу завдяки кращому врахуванню контексту. Згідно з дослідженнями, використання BERT у перекладацьких системах Google призвело до значного зростання показників BLEU (показник якості

перекладу), що демонструє покращення до 3-4 балів у деяких мовних парах. [7]

Саммаризація тексту

Автоматичне узагальнення тексту - ще одна сфера, де LLMs досягають успіху. Такі моделі, як BART (Bidirectional and Auto-Regressive Transformers - двонаправлені та авторегресивні трансформатори) від Facebook, розроблені спеціально для створення стислого переказу довгих документів. BART досягла найсучасніших результатів у різних тестах на узагальнення, включно з набором даних CNN/Daily Mail, з оцінкою ROUGE-L 44,16, що значно вище, ніж у попередніх моделей. ROUGE-L - це метрика, яка використовується для оцінювання якості реферування тексту та інших завдань обробки природної мови. Вона розшифровується як "Recall-Oriented Understudy for Gisting Evaluation" і фокусується на найдовшій спільній підпослідовності (LCS) між згенерованою саммаризацією тексту та еталонною. ROUGE-L широко використовується в NLP для вимірювання того, наскільки добре машинне узагальнення відображає важливий зміст оригінального тексту. [11]

Відповіді на запитання

LLM успішно застосовуються для завдань на відповіді на запитання, коли модель читає уривок тексту і відповідає на запитання до нього. GPT-3 від OpenAI продемонстрував вражаючу продуктивність у таких бенчмарках, як SQuAD (Стенфордський набір даних для відповідей на запитання), досягнувши показника точного збігу на рівні 86,4%. Ця здатність LLM використовується при розробці інтелектуальних віртуальних помічників і ботів для обслуговування клієнтів. [9]

Сентимент-аналіз

Галузь аналізу настроїв - завдання визначення настрою, вираженого в тексті, - також виграє від LLM. Такі моделі, як RoBERTa,

продемонстрували чудову продуктивність у завданнях аналізу настроїв, досягнувши показника F1 94,6% на бенчмарку GLUE. [12]

Етичний та суспільний вплив

Хоча LLMs відкривають численні можливості, вони також підіймають етичні та соціальні питання. Такі питання, як упереджені результати, дезінформація та можливість зловживання, потребують ретельного розгляду. OpenAI впровадив заходи безпеки в GPT-3, але для всебічного вирішення цих проблем необхідні подальші дослідження.

Підсумовуючи, можна сказати, що LLM зробили революцію в різних завданнях NLP - від генерації тексту і машинного перекладу до відповідей на запитання та аналізу настроїв. Їхня здатність розуміти і генерувати текст, подібний до людського, відкриває численні можливості, але також вимагає відповідального та етичного застосування для зменшення потенційних ризиків.

1.3 Переваги та недоліки великих мовних моделей

Великі мовні моделі трансформували не тільки NLP, але й весь світ, завдяки цьому. Однак вони мають як значні переваги, так і суттєві недоліки. У цьому розділі ми детально розглянемо ці аспекти.

Переваги

- Універсальність та узагальнення: однією з основних переваг LLM є їхня універсальність. Вони можуть виконувати різні завдання з мінімальним налаштуванням. Наприклад, GPT-3 може виконувати такі завдання, як написання есе, розв'язування математичних задач і генерація фрагментів коду.
- Навчання з кількох спроб та з нуля: LLMs можуть навчатися з кількох спроб і з нуля, що означає, що вони можуть розуміти і виконувати завдання з дуже малою кількістю конкретних прикладів

або навіть без них під час навчання. Ця здатність зменшує потребу у великих маркованих наборах даних для кожного нового завдання, що робить LLM високоефективними для найрізноманітніших застосувань.

- Масштабованість: продуктивність LLM має тенденцію до покращення зі збільшенням масштабу. Дослідження показали, що збільшення кількості параметрів, як правило, призводить до кращої продуктивності для широкого кола завдань. Наприклад, 175 мільярдів параметрів GPT-3 дають змогу генерувати більш зв'язний і контекстуально точний текст порівняно з меншими моделями. Тобто потенційно можна натренувати неймовірно велику модель, через це масштабованість є перевагою, проте варто враховувати, що кількість ресурсів, задіяна для цього, буде недоліком.
- Покращена взаємодія між людиною та комп'ютером: LLMs покращили взаємодію між людиною та комп'ютером завдяки вдосконаленим чат-ботам і віртуальним асистентам. Ці системи можуть розуміти і відповідати на запити користувачів більш природно і точно, покращуючи користувацький досвід у сфері обслуговування клієнтів, персональних асистентів та інтерактивних додатків. [13]

Недоліки

- Обчислювальні та екологічні витрати: тренування LLMs вимагає значних обчислювальних ресурсів та енергії. Наприклад, підготовка GPT-3 спожила приблизно 1287 МВт-год електроенергії, що має значний вплив на навколишнє середовище. [14] Висока вартість обчислень також обмежує доступ до LLM для організацій з мешною кількістю ресурсів.

- Питання упередженості та справедливості: LLM можуть виявляти і навіть посилювати упередження, присутні в навчальних даних. Наприклад, вони можуть генерувати упереджений або образливий контент, що відображає суспільні стереотипи. Усунення цих упереджень має вирішальне значення для справедливого та етичного розгортання ШІ. [15]
- Залежність від великих масивів даних: для навчання LLM потрібні величезні обсяги даних, що може бути обмеженням у тих галузях, де таких даних мало або їх важко отримати. Крім того, якість навчальних даних суттєво впливає на продуктивність та надійність моделі.
- Ризик дезінформації: LLM можуть генерувати текст, який є правдоподібним, але фактично неправильним, що створює ризик дезінформації. Цей недолік вимагає ретельного моніторингу та перевірки, особливо в додатках, пов'язаних із поширенням інформації. [16]

Підсумовуючи, можна сказати, що хоча LLM пропонують потужні можливості і сприяли значному прогресу в NLP, вони також пов'язані з істотними проблемами, які необхідно вирішувати. Баланс між їхніми перевагами та недоліками має вирішальне значення для відповідального та ефективного використання.

Розділ 2. Генерація медіа на мобільні платформі за допомогою великих мовних моделей

2.1 Проблеми хмарного доступу до великих мовних моделей

Хоча хмарний доступ до великих мовних моделей уможливив широке використання цих потужних інструментів, він також пов'язаний із низкою суттєвих проблем. У цьому розділі розглядаються ключові проблеми, пов'язані з хмарним доступом до великих мовних моделей, зокрема затримка, проблеми конфіденційності, вартість і залежність від підключення до Інтернету.

Затримка

Однією з головних проблем використання LLM за допомогою API є затримка. Коли користувач надсилає запит до LLM, розміщеного на віддаленому сервері, запит має пройти через Інтернет, бути обробленим сервером, а потім відповідь має повернутися назад до користувача. Такий обмін даними в обидва боки призводить до затримок. Для додатків, що вимагають відповіді в реальному часі або майже в реальному часі, таких як інтерактивні чат-боти або віртуальні асистенти, ця затримка може суттєво вплинути на користувацький досвід.

Наприклад, дослідження показали, що навіть незначні затримки в часі відгуку можуть призвести до зниження задоволеності користувачів. Дослідження Google показало, що додаткові 100 мілісекунд затримки в результатах пошуку призводять до помітного зниження залученості користувачів. [17] У критично важливих сферах, таких як охорона здоров'я або фінансові послуги, де вчасна реакція має вирішальне значення, затримка є суттєвим недоліком.

Проблеми конфіденційності

Хмарні LLM вимагають, щоб дані надсилалися через Інтернет на віддалені сервери для обробки. Це викликає значні занепокоєння щодо

конфіденційності, оскільки конфіденційна інформація може передаватися та зберігатися на зовнішніх серверах. Користувачі та організації повинні довіряти постачальникам послуг, що другі безпечно оброблятимуть їхні дані та дотримуватимуться відповідних правил конфіденційності.

Випадки витоку даних і несанкціонованого доступу до інформації, що зберігаються в хмарі, посилюють це занепокоєння. Наприклад, звіт компанії IBM Security показав, що середня вартість витоку даних у 2020 році становила 3,86 мільйона доларів США, причому основними причинами були скомпрометовані облікові дані та неправильна конфігурація хмарних сховищ [18]. Конфіденційність даних наразі має першорядне значення, через це використання LLM за допомогою API може становити значні ризики.

Вартість

Доступ до LLM через хмарні сервіси може бути дорогим, особливо при масштабному розгортанні або постійному використанні. Хмарні провайдери зазвичай стягують плату на основі обсягу спожитих обчислювальних ресурсів, включаючи процесор, графічний процесор, пам'ять і сховище. Для моделей з мільярдами та трильйонгами параметрів обчислювальні вимоги є значними, що призводить до високих операційних витрат.

Наприклад, робота з GPT-3 може коштувати кілька центів за запит, залежно від складності та довжини вхідних і вихідних даних. Для підприємств з великими обсягами запитів ці витрати можуть швидко зростати. В опитуванні, проведеному компанією Flexera, 61% підприємств визначили управління витратами на хмарні сервіси як важливу задачу [19], що вказує на популярність хмарних застосунків серед підприємств.

Залежність від підключення до Інтернету

API потребують постійного підключення до Інтернету для функціонування. Ця залежність може бути суттєвим обмеженням у

середовищах з ненадійним або обмеженим доступом до Інтернету. Користувачі у віддалених або слаборозвинених регіонах, або ті, хто має тимчасові проблеми з підключенням, можуть вважати хмарні рішення непрактичними або непридатними для використання.

Більше того, залежність від підключення до Інтернету створює потенційну вразливість до перебоїв у наданні послуг. Перебої в роботі провайдерів хмарних послуг можуть призвести до того, що LLM стануть недоступними, порушуючи роботу служб та операцій, які від них залежать. Наприклад, серйозний збій в роботі AWS у грудні 2021 року вплинув на численні сервіси по всьому світу, підкресливши ризики залежності від хмарної інфраструктури. [20]

Масштабованість і розподіл ресурсів

Масштабованість є ще однією проблемою для хмарних LLM. Хоча хмарні платформи пропонують масштабовані ресурси, ефективне управління та розподіл цих ресурсів може бути складним. Великі моделі потребують значних ресурсів графічного процесора та пам'яті, а їх масштабування для задоволення мінливого попиту без надмірних витрат вимагає ретельного планування та управління.

Дослідження показали, що неправильний розподіл ресурсів може призвести або до їх недовикористання, коли ресурси витрачаються даремно, або до надмірного використання, коли продуктивність погіршується через боротьбу за ресурси. [21] Ефективне масштабування LLM у хмарі передбачає збалансування вартості, продуктивності та доступності ресурсів, що може бути складним завданням для організацій, які не мають спеціалізованого досвіду.

Підсумовуючи, можна сказати, що хоча хмарний доступ до LLM пропонує гнучкість і потужні можливості, він також створює значні виклики, зокрема питання затримки, проблеми з конфіденційністю, велику вартість, залежність від підключення до Інтернету та проблеми

масштабованості. Вирішення цих проблем має вирішальне значення для максимізації переваг і мінімізації ризиків, пов'язаних з розгортанням хмарних LLM.

2.2 Переваги роботи з великими мовними моделями на пристрої

Розгортання великих мовних моделей безпосередньо на пристроях, на відміну від доступу до них через хмару, має кілька значних переваг. У цьому підрозділі описано основні з них, зокрема зменшення часу відповіді, підвищення рівня конфіденційності, економічна ефективність, автономна функціональність і кращий контроль користувача.

Зменшений час затримки

Однією з найпомітніших переваг LLM на пристрої є зменшення часу затримок. Завдяки локальній обробці даних на пристрої відпадає потреба у зворотному зв'язку з віддаленим сервером. Це призводить до швидшого часу відгуку, що має вирішальне значення для додатків, які працюють у режимі реального часу. Наприклад, локальна обробка може зменшити затримку до мілісекунд порівняно з потенційними сотнями мілісекунд і більше, необхідними для хмарних рішень.

Підвищена конфіденційність

LLM на пристрої значно підвищують рівень конфіденційності, гарантуючи, що конфіденційні дані залишаються на пристрої користувача. Такий підхід мінімізує ризик витоку даних і несанкціонованого доступу, пов'язаний з передачею даних на віддалені сервери та з них.

Дослідження компанії KPMG показало, що 87% споживачів вважають, що конфіденційність даних є фундаментальним правом людини, а 74% з більшою ймовірністю довірятимуть організаціям, які демонструють тверду прихильність до захисту їхніх даних. [22] Обробка

на пристрої добре узгоджується з цими очікуваннями щодо конфіденційності.

Економічна ефективність

Хоча початкове розгортання LLM на пристроях може передбачати вищі витрати через потребу в більш потужному обладнанні, поточні операційні витрати можуть бути значно нижчими порівняно з хмарними рішеннями. За користування хмарними сервісами не потрібно платити, і організації можуть уникнути витрат, пов'язаних з передачею та зберіганням даних на віддалених серверах.

Згідно з опитуванням компанії Flexera, 30% корпоративних витрат на хмарні сервіси витрачаються даремно через неефективність і непотрібні ресурси. [19] Обробка на пристрої може допомогти зменшити ці витрати за рахунок більш ефективного використання наявних апаратних можливостей.

Офлайн-функціональність

Однією з унікальних переваг LLM на пристроях є їхня здатність функціонувати без активного підключення до Інтернету. Це особливо корисно в середовищах з поганим або ненадійним зв'язком, наприклад, у віддалених районах, під час подорожей або в ситуаціях, коли постійний доступ до Інтернету неможливий.

Офлайн-функціональність гарантує, що критичні програми залишатимуться працездатними незалежно від стану мережі, забезпечуючи більш надійну та стабільну роботу користувачів. Наприклад, локальна обробка дозволяє мобільним пристроям виконувати такі завдання, як переклад і транскрипція, навіть в автономному режимі, що підвищує їхню корисність.

Кращий користувацький контроль і кастомізація

LLM на пристрої пропонують користувачам і розробникам більше можливостей для контролю та налаштування. Керуючи моделями

локально, розробники можуть тонко налаштовувати та оптимізувати моделі для конкретних застосувань і вподобань користувачів, не покладаючись на зовнішні сервіси.

Такий рівень контролю забезпечує більш персоналізований користувацький досвід і можливість впроваджувати кастомні функції, які можуть бути недоступні в типових хмарних рішеннях. Крім того, це дозволяє організаціям відповідати конкретним регуляторним вимогам, забезпечуючи повну прозорість і можливість кастомізації процесів обробки даних.

Енергоефективність та вплив на довкілля

Хоча підготовка LLMs є енергоємною, висновок на пристрої може бути більш енергоефективним, ніж використання хмарних сервісів, особливо якщо врахувати енергію, необхідну для передачі даних і віддаленої обробки. Зменшуючи залежність від центрів обробки даних, які є значними споживачами електроенергії та джерелами викидів вуглецю, обробка на пристрої може допомогти зменшити загальний вплив на навколишнє середовище.

Дослідження Strubell та ін. [23] показало, що центри обробки даних відповідають приблизно за 1% світового попиту на електроенергію, причому значна частина припадає на робочі навантаження ШІ. Перекладаючи більше завдань з обробки даних на пристрої кінцевих користувачів, можна пом'якшити вплив операцій зі штучним інтелектом на навколишнє середовище.

Підсумовуючи, робота з LLM на пристроях має низку переконливих переваг, серед яких зменшений час затримки, підвищена конфіденційність, економічна ефективність, офлайн-функціональність, кращий контроль користувача та зменшення впливу на навколишнє середовище. Ці переваги роблять обробку на пристрої привабливою альтернативою хмарним

рішенням, особливо для додатків, що вимагають високої продуктивності, конфіденційності та надійності.

2.3 Недоліки використання великих мовних моделей на пристроях

Хоча розгортання LLM на пристрої пропонує численні переваги, воно також пов'язане з деякими суттєвими проблемами. У цьому розділі розглянуто ключові проблеми, пов'язані з використанням великих мовних моделей на пристрої, зокрема апаратні обмеження, енергоспоживання, оновлення моделей, вимоги до сховища та ризики для безпеки.

Обчислювальна потужність

LLM є обчислювально інтенсивними і потребують значної обчислювальної потужності для ефективної роботи. Більшість споживчих пристроїв, таких як смартфони та планшети, мають обмежені обчислювальні ресурси порівняно з потужними хмарними серверами. Це обмеження може вплинути на продуктивність LLM на пристроях, що призводить до уповільнення часу відгуку і зниження точності виконання складних завдань.

Наприклад, хоча чіп Apple A15 Bionic в останніх моделях iPhone є потужним, він все ще не досягнув можливостей високопродуктивних графічних процесорів, що використовуються в центрах обробки даних, таких як Nvidia GeForce RTX 4090, який може забезпечити продуктивність до 100 терафлопс. [24, 25]

Обмеження пам'яті

Для ефективної роботи LLM вимагають значних обсягів оперативної пам'яті. Наприклад, GPT-3, що має 175 мільярдів параметрів, потребує сотні гігабайт пам'яті, що перевищує можливості більшості споживчих пристроїв. Навіть менші моделі, такі як BERT з 340 мільйонами

параметрів, можуть зазнавати труднощів при роботі в межах пам'яті типових смартфонів, які можуть мати лише 4-12 ГБ оперативної пам'яті.

Енергоспоживання

Запуск LLM на пристрої може призвести до значного споживання енергії, що впливає на час роботи акумулятора та терморегуляцію. Високі обчислювальні навантаження можуть швидко розряджати батарею мобільних пристроїв і спричиняти перегрів, що може погіршити продуктивність пристрою та якість роботи смартфону.

Часті оновлення

LLM потребують регулярних оновлень для покращення їхньої продуктивності та усунення вразливостей безпеки. Керування цими оновленнями на пристрої може бути складним, оскільки вимагає від користувачів частого завантаження великих файлів моделей. Цей процес може бути громіздким, особливо в регіонах з обмеженим доступом до Інтернету або для користувачів з обмеженнями на використання даних мобільного Інтернету.

Великі розміри моделей

LLM, особливо ті, що містять мільярди параметрів, потребують значного простору для зберігання. Наприклад, повна модель GPT-3 потребує сотні гігабайт пам'яті. Зберігання таких великих моделей на споживчих пристроях може бути недоцільним, зважаючи на обмежений обсяг пам'яті більшості смартфонів і планшетів.

Відсутність централізованого контролю

Хмарні рішення пропонують централізований контроль над політиками безпеки та оновленнями, що полегшує впровадження послідовних і комплексних заходів безпеки. Рішенням для пристроїв бракує такої централізації, що може призвести до фрагментарних і непослідовних практик безпеки на різних пристроях і в різних користувачів.

Проблеми оптимізації

Розробка застосунків LLM на пристрої вимагає значної оптимізації, щоб вписати модель в обмеження обчислювальних ресурсів і пам'яті пристрою. Цей процес може бути складним і трудомістким, вимагаючи спеціальних знань як про архітектуру моделі, так і про цільове обладнання.

Крос-платформна сумісність

Забезпечення безперебійної роботи LLM-додатків на пристроях на різних платформах (iOS, Android тощо) і пристроях (смартфони, планшети тощо) додає ще один рівень складності. Розробники повинні враховувати відмінності в апаратних можливостях, операційних системах і конфігураціях пристроїв, що може ускладнити процес розробки.

Підсумовуючи, можна сказати, що хоча LLM на пристрої пропонують суттєві переваги, вони також створюють значні проблеми, включаючи апаратні обмеження, споживання енергії, складність оновлення моделей, вимоги до зберігання даних та складність розробки.

2.4 Обмеження характерні для розширення клавіатури на iOS

Реалізація розширення клавіатури в iOS для використання великих мовних моделейна пристроях пов'язана з унікальними проблемами та обмеженнями. У цьому розділі розглядаються специфічні обмеження, пов'язані з розширеннями клавіатури iOS, зокрема обмеження, пов'язані з пісочницею та безпекою, обмеженнями пам'яті та продуктивності, проблемами взаємодії з користувачем, а також дотриманням правил App Store.

Пісочниця додатків

Програми iOS, включно з розширеннями клавіатури, працюють в ізольованому середовищі. Цей захід безпеки обмежує доступ розширення до системних ресурсів і даних користувача, ізолюючи його від інших

програм. Хоча це забезпечує безпеку та конфіденційність, це також обмежує функції, які може виконувати розширення клавіатури.

Наприклад, розширення клавіатури мають обмежений доступ до мережевих служб і не можуть вільно взаємодіяти з іншими програмами або службами на пристрої. Це обмеження може ускладнити виконання таких завдань, як оновлення моделі або завантаження додаткових даних, необхідних для LLM.

Обмежені дозволи

Розширення клавіатури на iOS мають обмежені дозволи порівняно з повноцінними програмами. Вони не можуть отримати доступ до конфіденційних даних користувача або виконувати фонові завдання, що виходять за рамки їх безпосередньої функціональності. Це обмеження забезпечує конфіденційність користувача, але також обмежує здатність розширення виконувати складні операції, які вимагають безперервної фонової обробки.

Обмеження пам'яті

Apple накладає суворі обмеження на використання пам'яті для розширень, включно з клавіатурними розширеннями. Ці обмеження гарантують, що розширення не споживають надмірні системні ресурси, які можуть погіршити продуктивність всього пристрою. Однак LLM займають багато пам'яті, і вписати велику модель у ці обмеження може нетривіальною задачею..

Наприклад, розширення клавіатури зазвичай працюють з максимальним лімітом пам'яті 30-50 МБ, тоді як навіть найменші з LLM, такі як DistilBERT, потребують сотні мегабайт для ефективної роботи. [26]

Вплив на продуктивність

Запуск LLM в розширенні клавіатури може суттєво вплинути на продуктивність пристрою. Високі обчислювальні вимоги LLM можуть уповільнювати інші процеси та швидше розряджати акумулятор пристрою.

Вкрай важливо, щоб розширення залишалося швидким і не заважало користувачеві користуватися пристроєм в цілому.

Вимоги до конфіденційності

Правила App Store від Apple накладають суворі вимоги до розширень клавіатури щодо конфіденційності. Вони повинні гарантувати, що будь-які зібрані дані є мінімальними, використовуються належним чином і безпечно обробляються. Будь-яка функція, яка може бути сприйнята як інвазивна або непотрібна, може призвести до відмови під час розгляду програми.

Обмеження вмісту

Розширення клавіатури також повинні відповідати обмеженням на вміст, накладеним App Store. Це включає в себе забезпечення того, щоб вміст, який генерується LLM, не порушував жодних вказівок, пов'язаних з образливим, шкідливим або неприйнятним вмістом. Впровадження надійних механізмів фільтрації контенту є необхідним для дотримання цих обмежень.

Налагодження та тестування

Розробка та тестування розширень клавіатури може бути складнішою у порівнянні з повноцінними програмами. Ізольоване середовище та інтеграція у фреймворк клавіатури можуть обмежувати видимість і доступність інструментів налагодження.

Підсумовуючи, впровадження розширення клавіатури на iOS для використання LLM на пристрої передбачає врахування різних обмежень, пов'язаних з пісочницею та безпекою, обмеженнями пам'яті та продуктивності, дотриманням рекомендацій App Store та обмеженнями можливостей для розробки. Вирішення цих проблем вимагає ретельного планування, оптимізації та дотримання специфічних для платформи рекомендацій, щоб створити функціональне і зручне для користувача розширення.

2.5 Технології для роботи з великими мовними моделями на iOS

Реалізація великих мовних моделей на iOS передбачає використання різних технологій та інструментів для забезпечення ефективної та продуктивної роботи. У цьому розділі розглядаються ключові технології, зокрема утиліта llm.cpp, популярний інструмент для розгортання LLM на пристроях з обмеженими ресурсами, репозиторій LLM.swift, доступні на GitHub, а також техніки оптимізації самої моделі, такі як квантування, обрізання та дистиляція знань.

Огляд llama.cpp

llama.cpp - це легка в плані ваги бібліотека, призначена для запуску великих мовних моделей на пристроях з обмеженими ресурсами, включаючи мобільні пристрої. Вона зосереджена на оптимізації використання моделі для забезпечення ефективної роботи при обмежених обчислювальних потужностях та пам'яті.

Ключові особливості:

- Легка та ефективна, підходить для мобільних та периферійних пристроїв.
- Підтримує різні архітектури LLM, включаючи GPT і BERT.
- Написана на C++ для високої продуктивності та низького використання пам'яті.

llm.cpp забезпечує відмінне рішення для розгортання LLM на пристроях iOS, оскільки вона розроблена для мінімізації використання ресурсів, зберігаючи при цьому точність і швидкість реакції моделі. Це робить її особливо корисною для таких додатків, як розширення клавіатури, де продуктивність та ефективність використання ресурсів є критично важливими. [27]

LLM.swift

LLM.swift - це проста і зручна бібліотека, яка дозволяє вам легко взаємодіяти з великими мовними моделями локально для macOS, iOS, watchOS, tvOS і visionOS. Ця бібліотека розроблена, щоб бути легкою та продуктивною, що робить її придатною для різних платформ Apple. [28]

Ключові особливості:

- Простий та інтуїтивно зрозумілий API для роботи з LLM.
- Підтримує декілька платформ Apple, включаючи iOS, macOS, watchOS, tvOS та visionOS.
- Надає можливість налаштувати параметр `maxTokenCount` для балансування продуктивності та використання пам'яті.

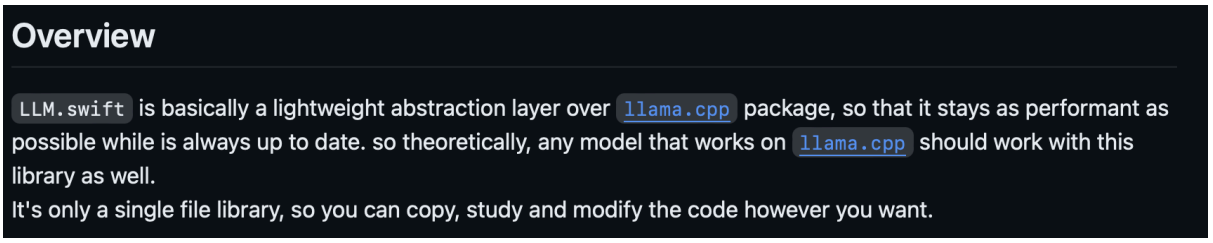

Рисунок 2.5.1 Скріншот з README файла бібліотеки LLM.swift

Методи оптимізації моделей

Квантування

Квантування (Quantization) - це метод, який використовується для зменшення розміру та обчислювальних вимог LLM шляхом представлення ваг та активацій з меншою точністю (наприклад, 8-бітними цілими числами замість 32-бітних чисел з плаваючою комою). Цей підхід може значно зменшити використання пам'яті та покращити швидкість

виведення, що робить його придатним для розгортання на пристрої. [29]

Model	Measure	F16	Q2_K	Q3_K_M	Q4_K_S	Q5_K_S	Q6_K
7B	perplexity	5.9066	6.7764	6.1503	6.0215	5.9419	5.9110
7B	file size	13.0G	2.67G	3.06G	3.56G	4.33G	5.15G
7B	ms/tok @ 4th, M2 Max	116	56	69	50	70	75
7B	ms/tok @ 8th, M2 Max	111	36	36	36	44	51
7B	ms/tok @ 4th, RTX-4080	60	15.5	17.0	15.5	16.7	18.3
7B	ms/tok @ 4th, Ryzen	214	57	61	68	81	93
13B	perplexity	5.2543	5.8545	5.4498	5.3404	5.2785	5.2568
13B	file size	25.0G	5.13G	5.88G	6.80G	8.36G	9.95G
13B	ms/tok @ 4th, M2 Max	216	103	148	95	132	142
13B	ms/tok @ 8th, M2 Max	213	67	77	68	81	95
13B	ms/tok @ 4th, RTX-4080	-	25.3	29.3	26.2	28.6	30.0
13B	ms/tok @ 4th, Ryzen	414	109	118	130	156	180

Рисунок 2.5.2 Результати квантування моделей можуть бути доволі вражаючими та зменшувати розмір моделі в 4 рази

Методи квантування підтримуються різними фреймворками, включаючи TensorFlow Lite та PyTorch Mobile, що дозволяє розробникам ефективно оптимізувати свої моделі для мобільних пристроїв.

Обрізка

Обрізка передбачає видалення менш важливих ваг і зв'язків у нейронній мережі для зменшення її розміру та складності. Ця техніка допомагає підтримувати продуктивність моделі, зменшуючи при цьому обчислювальне навантаження, що полегшує запуск LLM на пристроях з обмеженими ресурсами.

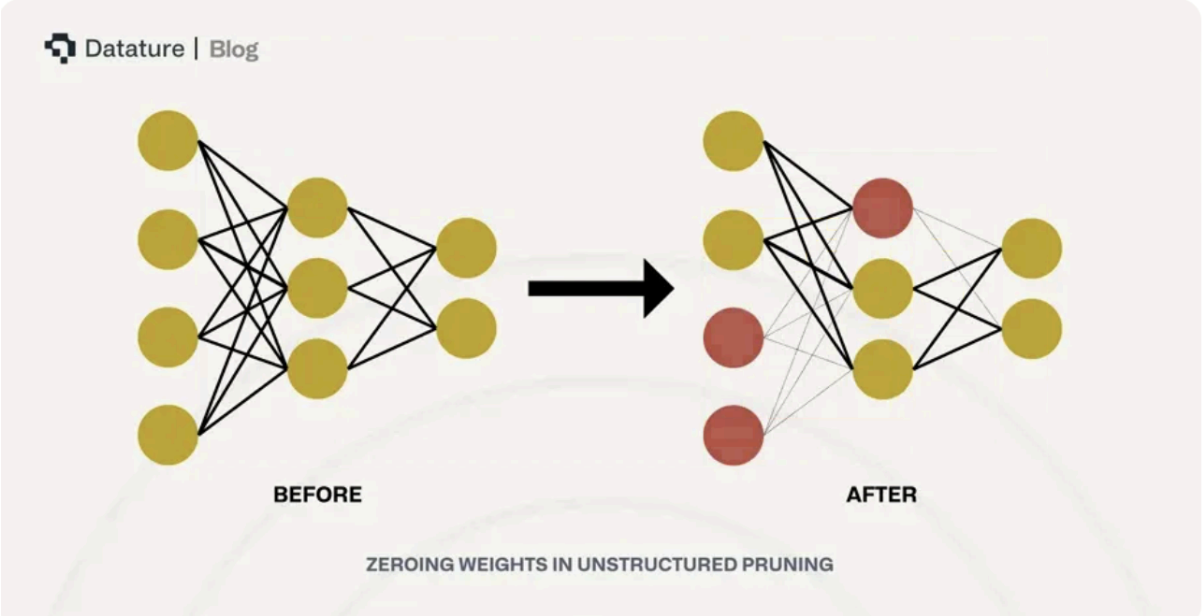

Рисунок 2.5.3 Візуалізація того, як обнуляються ваги під час неструктурованої обрізки

Обрізання можна поєднувати з іншими методами оптимізації, такими як квантування, для подальшого підвищення ефективності моделей на пристроях. [30]

Дистиляція знань

Дистиляція знань - це процес, коли менша, ефективніша модель (учень) навчається повторювати поведінку більшої, складнішої моделі (вчителя). В результаті виходить модель, яка зберігає більшу частину точності більшої моделі, але зі значно меншими розмірами та обчислювальними вимогами. [31]

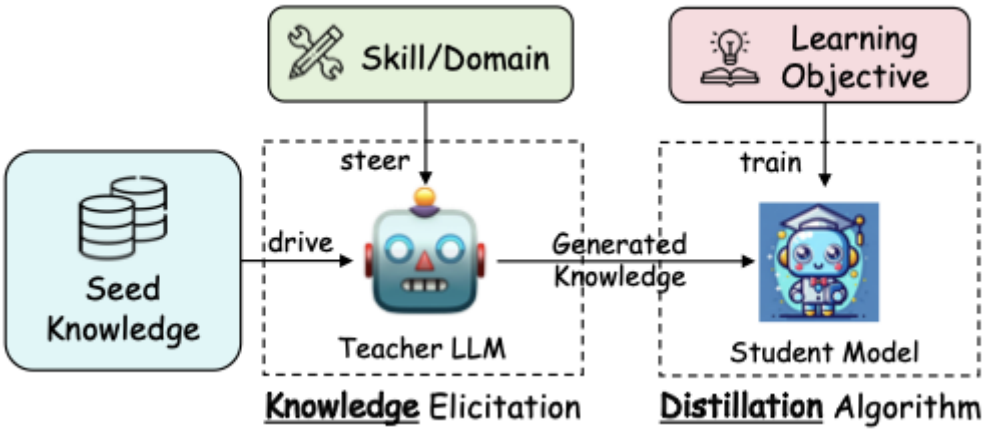

Рисунок 2.5.4 Ілюстрація загального конвеєра для дистиляції знань від великої мовної моделі до моделі-учня

DistilBERT є яскравим прикладом моделі, створеної шляхом дистиляції знань, спеціально оптимізованої для мобільних і периферійних пристроїв.

Таким чином, робота з LLM на iOS передбачає використання різних технологій, зокрема спеціалізованих бібліотек, таких як LLM.swift і llama.cpp, а також методів оптимізації, таких як квантування, обрізання та дистиляція знань. Ці інструменти та методи дозволяють ефективно розгортати LLM на мобільних пристроях, долаючи притаманні їм ресурсні обмеження та забезпечуючи високопродуктивні, адаптивні додатки.

Розділ 3. Реалізація розширення клавіатури з використанням великої мовної моделі

3.1 Функціональні вимоги до розширення клавіатури

Розширення клавіатури з використанням LLM спершу має дотримуватись наступних функціональних вимог для використання на пристрої:

- Аналіз тексту: розширення має аналізувати текст, що вводиться у текстовому полі.
- Аналізувати виділений текст: розширення також має аналізувати текст, який наразі виділено у текстовому полі.
- Генерація тексту на основі аналізу: натиснувши відповідну кнопку, розширення має надіслати проаналізований текст (або повний вміст поля введення, або виділений текст) до LLM і отримати відповідь. Ця відповідь може бути використана для створення різних творчих форматів тексту або доповнення ваших думок на основі контексту.
- Вставка згенерованої відповіді: згенерований текст має бути вставлений в поле вводу для зручності користувача.

3.2 Налаштування користувацького розширення клавіатури

Цей розділ присвячено елементам інтерфейсу користувача (UI) і функціям користувацької розкладки клавіатури, що ми розробили.

Для того, щоб розробити розширення клавіатури в UIKit нам необхідно в нашому проєкті створити окремий таргет, як це називається в XCode, який має назву Custom Keyboard Extension.

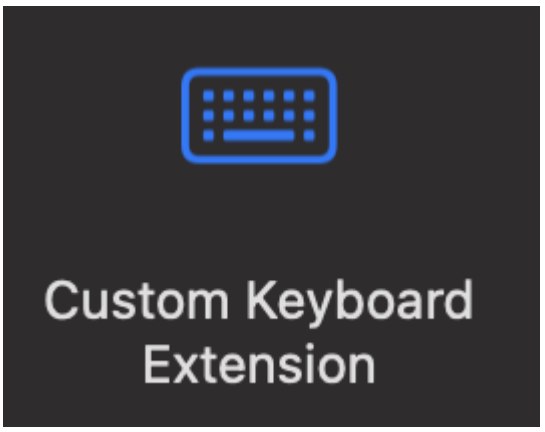

Рисунок 3.2.1 Вигляд в списку шаблонів для таргетів

При натиску, створюється окрема папка в проєкті, що містить власний Info.plist, а також клас, що налсідує UIInputViewController, в якому буде головний код розширення клавіатури. Наслідування UIInputViewController надає доступ до функціональних можливостей, необхідних для користувацького розширення клавіатури. [32]

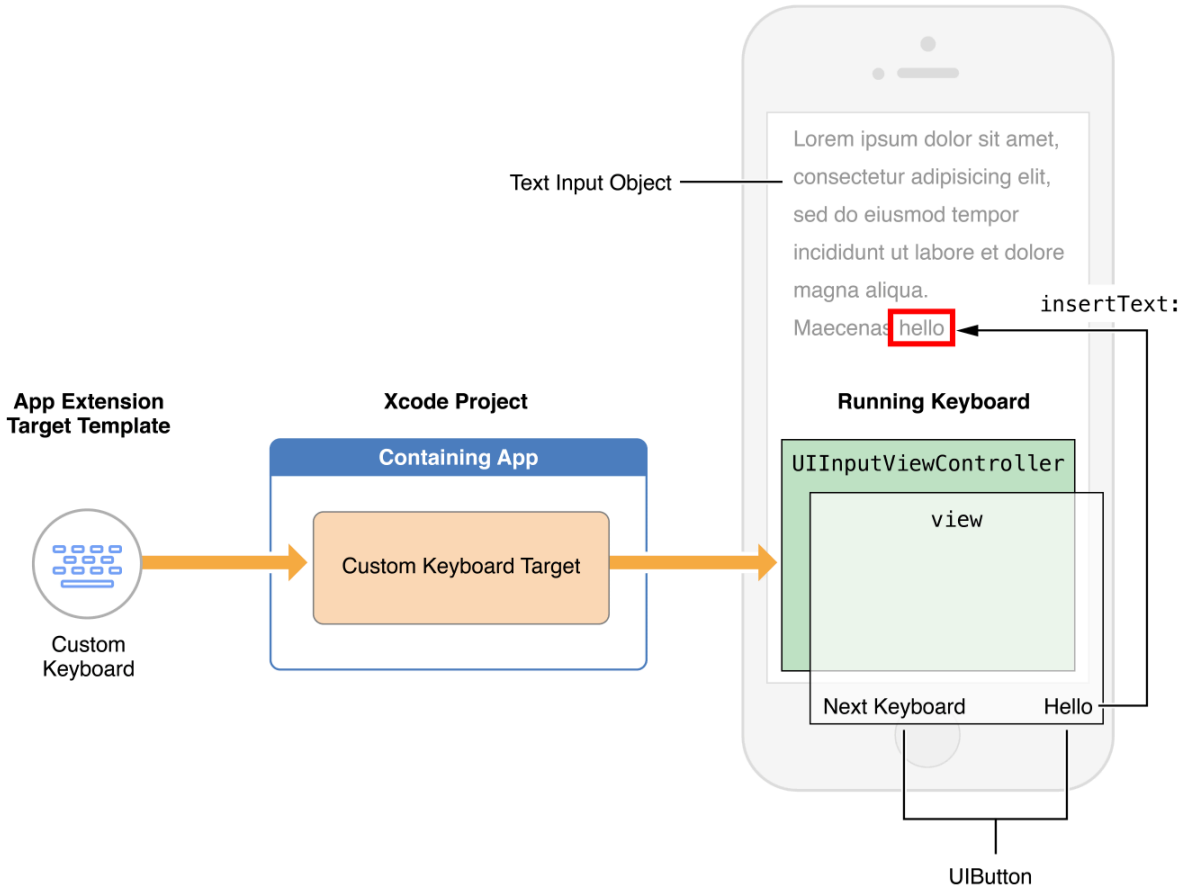

Рисунок 3.2.2 Базова структура кастомної клавіатури

Також в класі ми створюємо користувацьке представлення клавіатури (customKeyboardView) для того, щоб воно слугувало основою для всього інтерфейсу клавіатури.

```
private let customKeyboardView = UIInputView(frame: .zero, inputViewStyle: .keyboard)
```

Метод viewDidLoad перевизначається для налаштування користувацького вигляду клавіатури, коли те завантажилось.

```
override func viewDidLoad() {
    super.viewDidLoad()
    setupCustomKeyboard()
    setupProgressView()
}
```

Компоненти дизайну інтерфейсу користувача (UI):

1. *Кнопки:* створено дві кнопки з відповідними назвами: "Надіслати текст" та "Надіслати виділений текст".

   ```
   let buttonTitles = ["Send Text",
                       "Send Highlited Text"]
   ```

2. *Індикатор виконання:* індикатор виконання (`UIActivityIndicatorView`) включено, але спочатку приховано (`isHidden = true`). Він буде показаний під час обробки запиту моделлю, про що ми поговоримо далі.

   ```
   private var progressView: UIActivityIndicatorView!
   ```

3. *Взаємодія кнопок:* кожна кнопка пов'язана з певною дією за допомогою методів-селекторів:
   a. `textButtonTapped` - надіслати запит, що містить весь текст з поля вводу.
   b. `highlitedTextButtonTapped` - надіслати запит, що містить лише виділений текст.

У наступному підрозділі буде розглянуто, інтеграцію з LLM, та як ці кнопки взаємодіють з нею для аналізу та генерації тексту.

3.3 Інтеграція бібліотеки LLM

Після створення інтерфейсу користувача у попередньому розділі, давайте заглибимося у технічні аспекти інтеграції бібліотеки великої мовної моделі (LLM) для генерації медіа на пристрої.

1. Клас Assistant:

- У коді визначено клас `Assistant`, який успадковується від наданого класу бібліотеки LLM. Цей клас виступає посередником між інтерфейсом користувача та моделлю LLM.
- Конструктор класу `Assistant` ініціалізує LLM-модель наступним чином:
    - Завантаження моделі з вказаного файлу ресурсів з використанням URL, наданого `Bundle.main.url(forResource:)`.
    - Встановлення системного запиту за допомогою попередньо визначеної константи (`Const.systemPrompt`). Цей запит допомагає керувати відповідями LLM.
    - Вказівка максимальної кількості токенів (`Const.maxTokenCount`) для обмеження довжини вхідних послідовностей, що обробляються бібліотекою LLM.

```
class Assistant: LLM {
    convenience init() {
        let url = Bundle.main.url(forResource: Const.modelName, withExtension: Const.modelExtension)!
        let systemPrompt = Const.systemPrompt
        self.init(from: url, template: .llama(systemPrompt), maxTokenCount: Const.maxTokenCount)
    }
}
```

2. Функції видобування тексту:

- Визначено дві функції для отримання тексту для обробки LLM:
    - `getInputFieldText()`: ця функція використовує властивість `textDocumentProxy`, успадковану від

`UIInputViewController`, щоб отримати доступ до всього текстового вмісту, введеного у текстовому полі.

- `getHighlightedText()`: Ця функція також використовує `textDocumentProxy`, але повертає лише поточний виділений текст у полі введення.

3. Робочий процес обробки тексту:

- Методи `textButtonTapped` та `highlitedTextButtonTapped` (згадані у розділі інтерфейсу користувача) відповідають за ініціювання робочого процесу обробки LLM на основі натиснутої кнопки:
  - Ці методи спочатку перевіряють, чи вже завантажено екземпляр `Assistant`. Якщо ні, викликається функція `loadAssistant` для ініціалізації LLM-моделі.
  - Виділений текст (все поле введення або виділений текст) витягується за допомогою відповідної функції (`getInputFieldText` або `getHighlightedText`).
  - Викликається функція `handleTextExtractionAndRequest`, яка передає текст на вхід.

4. Функція обробки (`handleTextExtractionAndRequest`): ця функція обробляє відправку витягнутого тексту на обробку LLM:

- Спочатку вона показує індикатор виконання, викликаючи `showProgressViewAndHideButtons` для забезпечення візуального зворотного зв'язку з користувачем.
- Потім функція викликає `sendRequest`, щоб ініціювати зв'язок з моделлю LLM, передаючи витягнутий текст на обробку.

5. Відправлення запитів (`sendRequest`): ця функція отримує витягнутий текст і передає його на обробку:

- Попередньо обробляє текст за допомогою функції `assistant.preProcess`, що надається бібліотекою LLM. Цей крок

може включати такі завдання, як очищення або форматування тексту для кращого розуміння LLM.

- Попередньо оброблений текст надсилається для генерації LLM за допомогою функції `assistant.getCompletion`. Це асинхронна операція, тобто код продовжує виконання під час очікування відповіді LLM.

6. Отримання та обробка відповідей: після того, як LLM завершить обробку, відповідь буде отримано та збережено у змінній.

7. Інтеграція відповіді (`printResponse`):

- Функція `printResponse` отримує отриману відповідь LLM та інтегрує її в інтерфейс користувача:
- Якщо обробляється весь текст поля вводу, функція спочатку очищає існуючий вміст за допомогою `textDocumentProxy.deleteBackward`.
- Згенерована відповідь від LLM потім вставляється в текстове поле за допомогою `textDocumentProxy.insertText`.
- Нарешті, індикатор виконання ховається, а кнопки знову стають видимими за допомогою `hideProgressViewAndShowButtons`.

```
func printResponse(response: String) {
    if let inputFieldText = textDocumentProxy.documentContextBeforeInput {
        for _ in inputFieldText {
            textDocumentProxy.deleteBackward()
        }
    }
    textDocumentProxy.insertText(response)
    hideProgressViewAndShowButtons()
}
```

Цей підрозділ детально описує, як код взаємодіє з бібліотекою LLM, щоб використовувати її можливості для генерації медіа на пристрої. Елементи інтерфейсу користувача розглянуті раніше запускають цей

робочий процес, дозволяючи користувачам взаємодіяти з бібліотекою LLM і отримувати згенерований текст на основі введених ними даних.

3.4 Принцип роботи готового розширення клавіатури

Оскільки розширення клавіатури є окремим додатком, користувачеві потрібно буде увімкнути його в налаштуваннях iOS:

1. Користувачеві слід перейти до програми "Налаштування" на своєму пристрої.
2. Далі слід знайти пункт "Загальні" і натиснути на нього.
3. У "Загальних" налаштуваннях потрібно знайти розділ "Клавіатура" і натиснути на нього.
4. У розділі "Клавіатури" користувач повинен знайти розширення власної клавіатури LLM за назвою TesDiploma t і натиснути на нього.
5. Нарешті, користувач увімкне клавіатуру, переключивши перемикач поруч з назвою клавіатури.

Увімкнувши клавіатуру, користувач може відкрити будь-яку програму, що вимагає введення тексту, наприклад, програму обміну повідомленнями, поштовий клієнт або платформу соціальних мереж.

Коли користувач натискає на текстове поле, де він хоче ввести текст, за замовчуванням з'являється стандартна клавіатура iOS. Він вводить свій запит і після цього, щоб переключитися на наше кастомне розширення клавіатури LLM, користувач повинен натиснути і утримувати іконку глобуса або емодзі на стандартній клавіатурі. З'явиться список доступних розширень клавіатури.

Потім користувач повинен вибрати клавіатуру, названу TestDiploma.

Користувач побачить кнопки, пов'язані з надсиланням тексту для обробки LLM:

- "Надіслати текст"Ця кнопка проаналізує весь текст, який користувач ввів у поле, і надішле його на обробку до LLM. LLM може згенерувати відповідь, щоб завершити думку користувача, запропонувати творчі формати тексту на основі його змісту або надати резюме.
- "Надіслати виділений текст":** Якщо користувач виділив певну частину тексту, ця кнопка надішле лише виділений текст на обробку LLM. Відповідь від LLM може бути пов'язана з виділеним вмістом, пропонуючи пояснення або альтернативні формулювання.

Після того, як користувач набрав свій текст і вибрав бажаний варіант обробки LLM (натиснувши відповідну кнопку), розширення клавіатури відобразить індикатор прогресу, щоб показати, що LLM працює над запитом.

Після того, як LLM завершить обробку, розширення клавіатури інтегрує згенеровану відповідь назад у текстове поле, замінивши оригінальний текст.

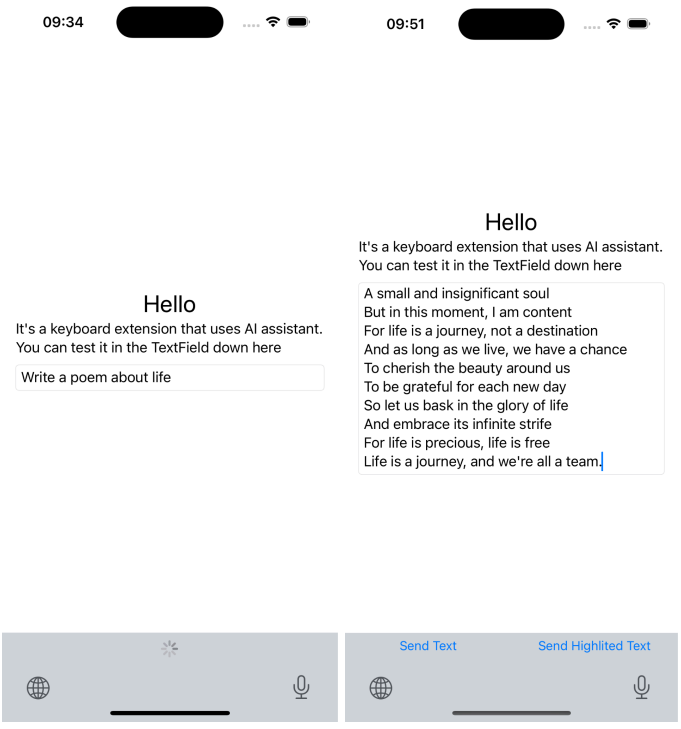

Рисунок 3.4.1 Демонстрація роботи розширення клавіатури

Висновки та аналіз можливостей для подальшого розвитку

У цій дипломній роботі досліджено потенціал використання великих мовних моделей на мобільних платформах для створення медіа на пристроях. Розширення для клавіатури iOS було розроблено як доказ концепції, щоб продемонструвати здійсненність і зручність такого підходу для користувачів.

Розроблене розширення клавіатури успішно інтегрує LLM для аналізу та генерації тексту на пристрої. Інтерфейс користувача надає функціональні можливості для надсилання на обробку LLM або весь текст поля введення, або тільки виділений текст. Інтеграція LLM дозволяє виконувати обробку в реальному часі та уникати залежності від хмарних API, що покращує конфіденційність та швидкість реагування.

Поточну реалізацію зосереджено на базовій обробці. Подальший розвиток може бути спрямований на вивчення більш досконалих методів обробки тексту та творчих форматів генерації тексту на основі можливостей LLM. Оптимізація коду для ефективного використання пам'яті має важливе значення для безперебійної роботи користувача.

Можливості для подальшого розвитку

- Розширення клавіатури може бути розширено для обробки різних текстових форматів, окрім простого тексту, таких як електронні листи, повідомлення або фрагменти коду. Можливості LLM можна використовувати для таких завдань, як узагальнення тексту, аналіз настроїв або завершення коду.
- Контекстно-орієнтована обробка: LLM може бути інтегрована з контекстною інформацією з навколишніх додатків або попередніх взаємодій користувача. Це дозволить отримати більш релевантні та персоналізовані відповіді від LLM, що підвищить його загальну корисність.

- Багатомовна підтримка: розширення клавіатури для підтримки декількох мов збільшить сферу його застосування і задовольнить ширшу базу користувачів. Можливості LLM повинні бути адаптовані для ефективної обробки різних мов.
- Додаткові функції: Інтерфейс користувача можна було б ще більше вдосконалити, включивши в нього інші елементи інтерфейсу, наприклад випадаючий список зі стилями мовлення, в які користувач хотів би переписати текст.

Загалом, розроблене кастомне розширення клавіатури слугує цінною відправною точкою для вивчення можливостей обробки LLM на пристрої для мобільної генерації тексту. Усуваючи обмеження і використовуючи виявлені можливості для подальшого розвитку, ця концепція має значні перспективи для трансформації способу взаємодії користувачів з текстом і створення контенту на своїх мобільних пристроях.

Список використаної літератури

1. GPT-3 powers the next generation of apps [Електронний ресурс]: https://openai.com/index/gpt-3-apps/
2. LaMDA: our breakthrough conversation technology [Електронний ресурс]: https://blog.google/technology/ai/lamda/
3. Large Language Models and APIs: A Story of Friendship and Distrust [Електронний ресурс]: https://equixly.com/blog/2024/02/13/llms-and-apis/
4. How Many Smartphones Are In The World? (2024) [Електронний ресурс]: https://www.bankmycell.com/blog/how-many-phones-are-in-the-world
5. AI App Revenue and Usage Statistics (2024) [Електронний ресурс]: https://www.businessofapps.com/data/ai-app-market/
6. Attention Is All You Need [Електронний ресурс]: https://arxiv.org/abs/1706.03762
7. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding [Електронний ресурс]: https://arxiv.org/abs/1810.04805
8. Universal Language Model Fine-tuning for Text Classification [Електронний ресурс]: https://arxiv.org/abs/1801.06146
9. Language Models are Few-Shot Learners [Електронний ресурс]: https://arxiv.org/abs/2005.14165
10. Dynamic data sampler for cross-language transfer learning in large language models [Електронний ресурс]: https://arxiv.org/abs/2405.10626
11. BART: Denoising Sequence-to-Sequence Pre-training for Natural Language Generation, Translation, and Comprehension [Електронний

ресурс]: https://arxiv.org/abs/1910.13461

12. RoBERTa: A Robustly Optimized BERT Pretraining Approach [Електронний ресурс]: https://arxiv.org/abs/1907.11692
13. Evaluating Quality of Chatbots and Intelligent Conversational Agents [Електронний ресурс]: https://arxiv.org/abs/1704.04579
14. Carbon Emissions and Large Neural Network Training [Електронний ресурс]: https://arxiv.org/abs/2104.10350
15. On the Dangers of Stochastic Parrots: Can Language Models Be Too Big? [Електронний ресурс]: https://www.researchgate.net/publication/349754361_On_the_Dangers_of_Stochastic_Parrots_Can_Language_Models_Be_Too_Big
16. Defending Against Neural Fake News [Електронний ресурс]: https://arxiv.org/abs/1905.12616
17. Speed Matters [Електронний ресурс]: https://research.google/blog/speed-matters/
18. Cost of a Data Breach Report [Електронний ресурс]: https://www.ibm.com/security/digital-assets/cost-data-breach-report/1Cost%20of%20a%20Data%20Breach%20Report%202020.pdf
19. Flexera Releases 2021 State of the Cloud Report [Електронний ресурс]: https://www.flexera.com/about-us/press-center/flexera-releases-2021-state-of-the-cloud-report
20. Prolonged AWS outage takes down a big chunk of the internet [Електронний ресурс]:

https://www.theverge.com/2020/11/25/21719396/amazon-web-services-aws-outage-down-internet

21. Resources Allocation in Cloud Computing: A Survey [Електронний ресурс]: https://www.researchgate.net/publication/338151834_Resources_Allocation_in_Cloud_Computing_A_Survey
22. KPMG Data Privacy Survey [Електронний ресурс]: https://kpmg.com/uk/en/home/misc/search.html?sp_p=any&q=Data%20privacy&SES=71937064846023162367512105139422 8&sort=_score&page=3&sp_c=9
23. Energy and Policy Considerations for Deep Learning in NLP [Електронний ресурс]: https://arxiv.org/abs/1906.02243
24. All-new iPad Air with advanced A14 Bionic chip available to order starting today [Електронний ресурс]: https://www.apple.com/ua/iphone/
25. Explore the Future of Visual Computing for Professionals [Електронний ресурс]: https://www.nvidia.com/en-eu/
26. App Extension Programming Guide - Custom Keyboard [Електронний ресурс]: https://developer.apple.com/library/archive/documentation/General/Conceptual/ExtensibilityPG/CustomKeyboard.html
27. llama.cpp [Електронний ресурс]: https://github.com/ggerganov/llama.cpp
28. LLM.swift [Електронний ресурс]: https://github.com/eastriverlee/LLM.swift
29. Quantization [Електронний ресурс]: https://huggingface.co/docs/optimum/concept_guides/quantization

30. A Comprehensive Guide to Neural Network Model Pruning [Електронний ресурс]: https://www.datature.io/blog/a-comprehensive-guide-to-neural-network-model-pruning
31. A Survey on Knowledge Distillation of Large Language Models [Електронний ресурс]: https://arxiv.org/pdf/2402.13116
32. UIInputViewController [Електронний ресурс]: https://developer.apple.com/documentation/uikit/uiinputviewcontroller