Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение высшего образования
«Московский государственный технический университет имени Н.Э. Баумана
(национальный исследовательский университет)»
(МГТУ им. Н.Э. Баумана)

ФАКУЛЬТЕТ Робототехника и комплексная автоматизация (РК)

КАФЕДРА Системы автоматизированного проектирования (РК6)

РАСЧЕТНО-ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

К НАУЧНО-ИССЛЕДОВАТЕЛЬСКОЙ РАБОТЕ

НА ТЕМУ:

«Задача игры в ассоциации: исследование возможностей больших и локальных моделей»

Студент РК6-22М _____ **Гунько Н.М.**
(Подпись, дата) Фамилия И.О.

Руководитель _____ **Витюков Ф.А.**
(Подпись, дата) Фамилия И.О.

2024 г.

Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение
высшего образования
«Московский государственный технический университет имени Н.Э. Баумана
(национальный исследовательский университет)»
(МГТУ им. Н.Э. Баумана)

УТВЕРЖДАЮ
Заведующий кафедрой РК6
А.П. Карпенко

«_____» _____ 2024 г.

З А Д А Н И Е
на выполнение научно-исследовательской работы

по теме: Задача игры в ассоциации: исследование возможностей больших и локальных моделей

Студент группы РК6-22М

Гунько Никита Макарович
(Фамилия, имя, отчество)

Направленность НИР (учебная, исследовательская, практическая, производственная, др.) учебная
Источник тематики (кафедра, предприятие, НИР) кафедра

График выполнения НИР: 25% к 5 нед., 50% к 11 нед., 75% к 14 нед., 100% к 16 нед.

Техническое задание: Исследовать возможности крупных языковых моделей (например, ChatGPT) и локальных минималистичных моделей (LLaMA, Mistral, Gemma) для решения задачи игры в ассоциации; установить и протестировать нейросеть 2txt на задаче ассоциаций с изображениями.

Оформление научно-исследовательской работы:
Расчетно-пояснительная записка на 39 листах формата А4.
Перечень графического (иллюстративного) материала (чертежи, плакаты, слайды и т.п.):

Дата выдачи задания «15» февраля 2024 г.

Руководитель НИР _____ **Витюков Ф.А.**
(Подпись, дата) Фамилия И.О.

Студент _____ **Гунько Н.М.**
(Подпись, дата) Фамилия И.О.

Примечание: Задание оформляется в двух экземплярах: один выдается студенту, второй хранится на кафедре.

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

Игра в ассоциации – это один из ключевых процессов человеческого мышления, основанный на умении находить связи между словами, понятиями или образами. Это упражнение отражает способность мозга анализировать, структурировать и синтезировать информацию, а также играть значимую роль в творческих процессах и когнитивных задачах. Технологический прогресс в области машинного обучения и языковых моделей привёл к созданию систем, которые могут имитировать подобные процессы на уровне текста, что открывает перспективы для использования таких моделей в играх, образовании и интерактивных системах.

Современные крупные языковые модели, такие как ChatGPT, стали новаторскими решениями в области обработки естественного языка. Они демонстрируют невероятную гибкость и возможность решать широкий спектр задач, включая генерацию текста, анализ контекста и даже имитацию человеческих диалогов [1, 2]. Однако такие модели обладают значительным количеством параметров, что делает их ресурсоёмкими как в плане вычислительных мощностей, так и затрат на хранение и обработку данных. Это ставит под вопрос их эффективность в решении узкоспециализированных задач, таких как игра в ассоциации, где требуется простота и скорость отклика без необходимости в огромных вычислительных ресурсах.

С другой стороны, локальные языковые модели, такие как LLaMA, Mistral и Gemma, предлагают альтернативные подходы. Важно понять, насколько такие модели могут справляться с задачей поиска ассоциаций между словами, и выявить наиболее производительные и экономичные решения для дальнейшей разработки специализированных систем.

Цель данной работы – исследовать подходы к созданию языковой модели для игры в ассоциации, а также проанализировать существующие крупные и локальные модели, чтобы понять их эффективность в решении этой

задачи. Для достижения поставленной цели исследования были выделены следующие задачи:

– Исследовать и продемонстрировать, как крупные языковые модели, такие как ChatGPT и аналогичные, справляются с задачей игры в ассоциации.

– Исследовать локальные языковые модели, такие как LLaMA, Mistral, Gemma, с целью выявления наиболее минималистичной и менее затратной по производительности модели для решения задачи.

– Установить и протестировать нейросеть 2txt для проверки её эффективности в задаче ассоциаций с изображениями.

1. Игра в ассоциации

Игра в ассоциации – это когнитивное упражнение, которое используется для тренировки мышления и креативности, а также для выявления взаимосвязей между понятиями. Суть игры заключается в том, чтобы на основе одного слова (или изображения) подобрать другое слово, которое с ним связано на основе логических, тематических или даже субъективных ассоциаций. Игра может проводиться в различных форматах, начиная от простых парных игр, в которых участники поочередно предлагают ассоциативные слова, и заканчивая более сложными соревновательными форматами, где участники стремятся найти как можно больше уникальных связей за ограниченное время.

Ассоциации играют важную роль в познании, потому что они позволяют человеку систематизировать информацию, анализировать данные и извлекать из них скрытые связи. Они также являются неотъемлемой частью творческих процессов, способствуя созданию новых идей и нестандартных решений. Возможность нахождения ассоциаций демонстрирует гибкость и вариативность мышления, что делает задачу разработки моделей для ассоциаций интересной и востребованной в области искусственного интеллекта.

1.1. Принципы и механика игры

Механика игры в ассоциации довольно проста, но в то же время многообразна. Основное правило заключается в следующем: участнику игры предлагают слово, и он должен быстро назвать другое слово, которое вызывает у него ассоциацию с исходным. Это может быть синоним, антоним, тематически связанное понятие или даже слово, имеющее определённую связь через личный опыт или культурный контекст. Например, если исходное слово – «море», то возможные ассоциации могут включать «волны», «пляж», «корабль», «соль» и т.д.

Существуют различные форматы игры. В парных играх участники поочередно называют слова, связанные с предыдущим, пока один из них не сможет быстро придумать ассоциацию или пока не будет найдено взаимное согласие о завершении раунда. В групповых и соревновательных форматах игроки делятся на команды и соревнуются, кто быстрее и точнее сможет предложить наиболее подходящие ассоциации. Вариантов может быть много, включая игры на время, игры на уникальные ассоциации (где важно предложить редкое, но логичное слово) и даже игры, где ассоциации могут быть оценены по степени их креативности.

Важным аспектом игры является скорость и креативность мышления. В отличие от задач, которые можно решать последовательно и обдуманно, ассоциации требуют от игрока способности находить слова на основе интуитивных и часто скрытых связей. Это позволяет быстро выявить уровень гибкости и вариативности мышления, а также продемонстрировать, насколько участник способен мыслить нестандартно.

1.2. Типы связей между словами

В игре в ассоциации возможны различные типы связей между словами, которые можно классифицировать следующим образом:

1. Семантические связи (синонимы и антонимы) – это связи, основанные на значении слов. Например, «радость» и «счастье» являются синонимами, так как обозначают похожие понятия, а «тепло» и «холод» – антонимы, так как противоположны по значению. Такие ассоциации являются наиболее предсказуемыми и легко воспринимаемыми как человеком, так и языковыми моделями.

2. Тематика и категория – слова, которые объединены общей темой или категорийным понятием. Например, слова «кошка», «собака» и «хомяк» объединяются категорией «домашние животные». Эти ассоциации опираются на знание предметной области и часто включают в себя обширные связи с контекстом, в котором эти слова встречаются.

3. Логические и функциональные связи – ассоциации, которые строятся на логических или причинно-следственных связях. Например, «ключ» и «замок», «птица» и «гнездо», «молния» и «гром». В таких парах можно проследить логическую или функциональную связь, основанную на причинно-следственных отношениях или совместном использовании предметов.

4. Фонетические и ритмические связи – ассоциации, основанные на схожести звучания слов или их ритмической структуры. Например, «мышь» и «тышь», «зуб» и «труб». Такие ассоциации могут быть креативными и неочевидными, однако они демонстрируют способности модели находить связи не только на уровне значений, но и на уровне фонетических характеристик.

5. Культурные и личные ассоциации – связи, которые зависят от личного опыта, культурного фона или популярности определённых понятий в обществе. Например, слово «музыка» может ассоциироваться с именами популярных исполнителей, жанрами или даже конкретными песнями. Такие ассоциации требуют более сложного анализа и часто зависят от особенностей культуры и опыта конкретного человека.

Понимание и использование всех этих типов связей является важным для создания моделей, способных эффективно работать в задаче ассоциаций. Большинство языковых моделей способны предсказывать наиболее вероятные семантические связи, но сложнее обрабатывать культурные и личные ассоциации, а также фонетические связи, которые требуют дополнительных знаний и понимания контекста.

1.3. Роль языковых моделей в задаче ассоциаций

Языковые модели хорошо подходят для решения задачи игры в ассоциации благодаря своей способности анализировать контекст и предсказывать вероятные слова на основе огромных объемов данных. Современные модели, такие как GPT-4o, обучены на миллиардах текстов, что позволяет им выявлять связи между словами, которые характерны для

человеческого языка. В основе их работы лежит принцип вероятностного анализа, где модель предсказывает, какое слово с наибольшей вероятностью должно следовать за заданным словом или быть связано с ним. Это позволяет моделям находить такие ассоциации, как «кофе» и «утро», «автомобиль» и «двигатель», поскольку они часто встречаются вместе в текстах.

Однако роль языковых моделей не ограничивается лишь предсказанием семантических и тематических связей. Они также могут выявлять более сложные логические и даже креативные связи, опираясь на контексты, в которых встречались слова. Например, модели могут находить ассоциации между словами «звёзды» и «путешествие», основываясь на литературных контекстах или метафорических выражениях. Это делает языковые модели полезными инструментами для создания интеллектуальных систем, которые могут не только выполнять предсказуемые задачи, но и демонстрировать креативность и гибкость мышления.

Тем не менее, есть определённые ограничения в работе языковых моделей при решении задачи игры в ассоциации. Во-первых, модели могут сталкиваться с проблемами многозначности слов. Например, слово «ключ» может означать и музыкальный инструмент, и инструмент для открывания замков, и важное понятие в определённой сфере знаний. Модель должна понимать контекст, чтобы предсказать правильную ассоциацию. Во-вторых, модели затрудняются с предложением креативных и редких ассоциаций, которые выходят за рамки предсказуемых шаблонов, поскольку их алгоритмы обучены на существующих данных и не всегда способны создавать новые и неожиданные связи.

Таким образом, языковые модели обладают значительным потенциалом для использования в игре в ассоциации, но для достижения максимальной эффективности они требуют улучшений, которые помогут им лучше обрабатывать многозначные, культурные и креативные связи. Современные подходы к разработке таких моделей позволяют внедрять знания о контексте, учитывать особенности языка и использовать методы обучения, направленные

на повышение их гибкости и интуитивного понимания связей между понятиями.

2. Анализ крупной языковой модели

Крупные языковые модели, такие как GPT-4, представляют собой передовые достижения в области обработки естественного языка. Они способны эффективно решать широкий спектр задач, включая генерацию текста, анализ контекста и поиск ассоциаций между словами. Эти модели обучаются на огромных объемах текстовых данных, что позволяет им обнаруживать и воспроизводить сложные взаимосвязи между словами и фразами.

2.1. Описание работы GPT-4

GPT-4 (Generative Pre-trained Transformer 4) является четвёртым поколением в семействе трансформерных моделей, разработанных компанией OpenAI. Она построена на архитектуре трансформера, которая позволяет эффективно обрабатывать последовательности данных и учитывать длинные зависимости между элементами последовательностей [3, 4]. Основным компонентом этой архитектуры является механизм внимания (self-attention), который позволяет модели фокусироваться на важных частях входного текста при генерации ответа.

Модель GPT-4 отличается высокой масштабируемостью и способностью обрабатывать большие объемы текстовых данных. Она обучена на обширных текстовых корпусах, включающих различные типы текстов, от научных статей и новостных материалов до художественных произведений и социальных сетей. Это разнообразие позволяет модели понимать и генерировать текст на различных уровнях сложности и стиля, что делает её универсальным инструментом для обработки естественного языка.

Одной из особенностей GPT-4 является её способность понимать контекст и предсказывать наиболее вероятные слова, исходя из контекстуальных подсказок. Это делает модель эффективной для задач,

связанных с генерацией текста, перевода, анализа настроений и, в том числе, поиска ассоциаций между словами.

2.2. Плюсы и минусы использования модели в своем решении

Одним из ключевых преимуществ GPT-4 является её высокая производительность. Модель обучена на миллиардах параметров, что позволяет ей эффективно обрабатывать сложные задачи и находить скрытые закономерности в текстовых данных [3, 4]. За счёт своей способности учитывать длинные зависимости и анализировать большие объемы информации модель может предсказывать ассоциации между словами с высокой точностью. Это делает её незаменимой для задач, требующих сложного анализа текста, таких как синтез информации, генерация креативного контента и нахождение логических связей.

Производительность модели также подкрепляется её способностью обрабатывать данные параллельно благодаря архитектуре трансформера, что позволяет значительно ускорить обработку текста по сравнению с более традиционными методами. Модель может одновременно анализировать несколько предложений и генерировать ответы, которые соответствуют различным контекстам и условиям задачи.

Несмотря на многочисленные преимущества, GPT-4 имеет и ряд недостатков, связанных с высокими требованиями к вычислительным ресурсам и высокой стоимостью внедрения. Обучение и использование таких крупных моделей требует значительных затрат на оборудование, в частности на мощные графические процессоры (GPU) и системы распределённой обработки данных. Это делает использование модели труднодоступным для небольших компаний и индивидуальных разработчиков, которые не обладают соответствующими ресурсами.

Кроме того, для обеспечения высокой производительности требуется поддержка значительных объемов памяти и хранение огромных массивов

данных, что также приводит к росту эксплуатационных расходов. Эти факторы ограничивают применение GPT-4 в случаях, когда требуется высокая масштабируемость и экономичность решений.

Ещё одной проблемой является безопасность данных. Поскольку модель GPT-4 разработана и поддерживается OpenAI, её использование предполагает обработку данных через удалённые серверы компании. Это означает, что все запросы пользователей передаются через внешние сервера, что может вызвать обеспокоенность относительно конфиденциальности данных, особенно в случае работы с личной или чувствительной информацией. В связи с этим использование GPT-4 может быть небезопасным для приложений, где необходимо строго контролировать доступ к данным и защищать их от возможных утечек.

2.3. Использование в задаче ассоциаций

GPT-4 может быть использована для создания интеллектуальных систем, которые способны находить и предлагать ассоциации на основе заданных терминов. Для оценки эффективности модели в задаче ассоциаций был проведён эксперимент, целью которого являлось проверка способности модели генерировать ассоциативные связи на основе входных терминов. Для взаимодействия с моделью был использован OpenAI API, позволяющий отправлять запросы к модели GPT-4 и получать ответы, обработанные с учётом её вероятностных ассоциаций. В рамках эксперимента модели предлагалось определённое слово, на основе которого она должна была сгенерировать ассоциативный ряд из нескольких слов, связанных с исходным термином.

Программа была реализована на языке программирования Python с использованием библиотеки openai. Ниже представлен полный листинг программы, позволяющий отправлять запросы к модели GPT-4 через OpenAI API.

Листинг 1 – Программа для генерации ассоциаций с использованием модели GPT-4 через OpenAI API

```
import openai

# OpenAI API-key
openai.api_key = "YOUR_API_KEY_HERE"

def get_associations(word, num_responses=5, temperature=0.7, attempts=1):
    """
    Функция для получения ассоциаций от модели GPT-4
    """
    prompt = f"Представь, что ты играешь в ассоциации. Назови {num_responses} слов, связанных с '{word}'."

    try:
        response = openai.Completion.create(
            engine="gpt-4",
            prompt=prompt,
            max_tokens=50,
            temperature=temperature,
            n=attempts,
            stop=None
        )

        associations = [resp.text.strip() for resp in response.choices]
        return associations

    except Exception as e:
        print(f"Произошла ошибка при запросе к OpenAI API: {e}")
        return []

def main():
    print("Добро пожаловать в программу для генерации ассоциаций с использованием модели GPT-4!\n")

    while True:
        # Получаем слово от пользователя
        word = input("Введите слово для ассоциаций (или 'exit' для выхода): ")
        if word.lower() == 'exit':
            print("Программа завершена.")
            break

        # Получаем параметры от пользователя
        try:
            num_responses = int(input("Введите количество ассоциаций (по умолчанию 5): ") or 5)
            temperature = float(input("Введите креативность (от 0 до 1, по умолчанию 0.7): ") or 0.7)
```

```
            attempts = int(input("Введите количество попыток (по умолчанию 1): ")
or 1)
        except ValueError:
            print("Ошибка ввода. Используются параметры по умолчанию.")
            num_responses, temperature, attempts = 5, 0.7, 1

        # Получаем ассоциации от модели
        associations = get_associations(word, num_responses, temperature,
attempts)

        # Выводим результаты
        if associations:
            print(f"\nАссоциации для слова '{word}':")
            for i, assoc in enumerate(associations, 1):
                print(f"Попытка {i}: {assoc}")
        else:
            print("Не удалось получить ассоциации. Попробуйте снова.")

        print("\n" + "-"*40 + "\n")

if __name__ == "__main__":
    main()
```

Программа получает от пользователя слово и параметры для запроса, такие как количество ассоциаций, уровень креативности (temperature), и число попыток. Далее формирует запрос для модели GPT-4 с учетом этих параметров и отправляет его через OpenAI API. В ответ от модели GPT-4 возвращается список ассоциаций, который затем отображается пользователю. Если модель не может предложить ассоциации или возникнет ошибка, программа оповестит об этом и позволит пользователю повторить запрос.

2.4. Проведение экспериментов

Для всесторонней оценки программы было проведено пять сложных тестов, в которых варьировались входные параметры – уровень креативности, количество ассоциаций и число попыток. Это позволило проверить, как модель GPT-4 реагирует на различные условия задачи, а также выявить ее возможности в нахождении ассоциаций разного уровня сложности и оригинальности.

1) Первый тест был направлен на проверку способности модели создавать обширный список ассоциаций для достаточно общего слова “вода”.

В настройках был задан высокий уровень количества ассоциаций (8), умеренный уровень креативности (temperature = 0.7) и одна попытка запроса. В результате модель предложила такие ассоциации, как “река”, “озеро”, “море”, “капля”, “чистота”, “жидкость”, “дождь” и “океан”. Эти ассоциации охватывают как природные объекты, так и физические свойства воды, что свидетельствует о способности модели предоставлять разнообразные ответы в пределах ожидаемого контекста. Средний уровень креативности позволил получить предсказуемые и логичные ассоциации, которые подходят для общеупотребительного понимания концепта “вода”.

2) Во втором тесте использовалось слово “мысль”, а параметр креативности был повышен до 0.9, чтобы проверить, как GPT-4 справляется с более абстрактными и глубокими концептами. Количество ассоциаций было снижено до пяти, так как основной целью являлось качество и оригинальность предложенных ассоциаций. Модель сгенерировала такие ассоциации, как “идея”, “творчество”, “свобода”, “сознание” и “вдохновение”. Эти ассоциации оказались менее очевидными и более метафоричными, что указывает на способность модели переходить от конкретного значения к более абстрактным и философским аспектам понятия “мысль”. Температура 0.9 способствует созданию уникальных связей, что может быть полезно в задачах, где требуется нестандартный подход и оригинальные идеи.

3) Третий тест был направлен на оценку того, как модель справляется с многозначными словами при низком уровне креативности. Для слова “память” была установлена температура 0.4, три попытки и три ассоциации на каждую попытку. Это позволило оценить, как модель работает с многозначностью при более узких рамках. Модель предложила такие ассоциации, как “прошлое”, “опыт”, “мозг”, “воспоминания”, “разум”, “информация" и “архив". Низкий уровень креативности способствовал получению логичных и предсказуемых ответов, а три попытки дали возможность разнообразить ассоциации, добавив слова, относящиеся к биологическим, когнитивным и информационным аспектам "памяти". Это

демонстрирует, что при необходимости модель может фокусироваться на стандартных значениях, создавая надежные и точные связи.

4) Для четвертого теста использовалось слово “ключ", которое обладает множеством значений и ассоциаций. Установив умеренную креативность (temperature = 0.6) и три попытки с четырьмя ассоциациями на каждую, мы стремились изучить, как модель работает с многозначностью в условиях средней креативности. В результате модель предложила такие ассоциации, как “замок”, “дверь”, “музыка”, “решение”, “ответ”, “аккорд”, “нотная строка”, “ключница” и “открытие”. Модель продемонстрировала способность интерпретировать слово “ключ” как в контексте предмета (например, ключ к замку), так и в музыкальном контексте (аккорд, нотная строка). Это подтверждает, что модель может обрабатывать многозначные слова, адаптируя свои ответы в зависимости от значений, которые могут быть релевантными в заданном контексте. Температура 0.6 позволила получить оригинальные и полезные ассоциации для разнообразных значений слова.

5) Пятый тест был проведён для концепта “путешествие”, который часто вызывает ассоциации, связанные как с физическим передвижением, так и с культурными и личными ценностями. Была выбрана высокая креативность (temperature = 0.85), чтобы модель генерировала более разнообразные и креативные ассоциации. Количество попыток было установлено на 2, чтобы можно было получить альтернативные версии ассоциативных рядов. В ответах модель предложила такие ассоциации, как “приключение”, “свобода”, “познание”, “дорога”, “новое”, “опыт”, “исследование”, “культура”, “открытия”, “мир”, “вдохновение” и “поиск”. Эти ассоциации охватывают широкий спектр понятий, связанных с путешествием: от физического (дорога) до культурного и личного опыта (культура, вдохновение, поиск). Высокая креативность позволила получить ответы, которые выходят за рамки простых ассоциаций, указывая на культурные и философские аспекты путешествия. Это демонстрирует способность модели учитывать как общественные, так и личные ценности.

2.5. Общий анализ результатов

Результаты пяти тестов показали, что модель GPT-4 способна успешно адаптироваться к условиям задачи благодаря настройке параметров. При низком уровне креативности и ограниченном количестве ассоциаций модель предлагала более предсказуемые и общепринятые ответы, что полезно при необходимости точных и простых ассоциаций. Высокая креативность позволила получить ассоциации, выходящие за пределы стандартных шаблонов, что особенно ценно для слов с абстрактными или многозначными значениями.

Кроме того, многократные попытки увеличивали разнообразие ассоциаций, позволяя пользователю изучить несколько возможных интерпретаций одного слова. В случае многозначных слов, таких как "ключ", модель успешно интерпретировала слово в разных контекстах, генерируя как физические, так и абстрактные ассоциации.

Таким образом, результаты демонстрируют, что GPT-4 может гибко реагировать на изменения параметров и успешно работать с разными типами ассоциативных задач, предоставляя качественные и разнообразные ответы, адаптированные под конкретные потребности пользователя.

3. Анализ локальных языковых моделей

Локальные языковые модели, такие как LLaMA, Mistral и Gemma, представляют собой компактные альтернативы крупным языковым моделям. Эти модели разработаны с акцентом на экономию ресурсов, что делает их доступными для локального развертывания, включая выполнение на персональных компьютерах и серверах с ограниченными вычислительными мощностями. Рассмотрим особенности их архитектуры, применимость для задачи ассоциаций, а также проведем тестирование этих моделей для оценки их эффективности.

3.1. Особенности LLaMa, Mistral и Gemma

Локальные модели, такие как LLaMA, Mistral и Gemma, предлагают уникальные архитектурные решения и подходы к обучению, которые делают их менее ресурсоёмкими по сравнению с крупными языковыми моделями, такими как GPT-4. Каждая из этих моделей имеет свои особенности, что позволяет оценивать их относительно требований к памяти, скорости обработки данных и применимости в различных задачах.

- Модель LLaMA (Large Language Model Meta AI) была разработана исследовательской группой Meta и ориентирована на эффективность работы на устройствах с ограниченными ресурсами [5]. LLaMA использует архитектуру трансформера с оптимизированной конфигурацией слоев, что позволяет снизить требования к памяти и вычислительным ресурсам без значительного снижения качества ответа. Кроме того, LLaMA использует специализированные методы регуляризации и сокращения параметров, чтобы минимизировать потерю информации при сокращении количества слоев. Обучение LLaMA также включает методы адаптивной оптимизации, которые помогают снизить энергозатраты при сохранении точности предсказаний.

- Mistral – это ещё одна локальная языковая модель, отличающаяся высокой оптимизацией и эффективностью использования памяти. В её архитектуре реализованы улучшенные методы регуляризации и упрощенные слои внимания, что позволяет ускорить обработку текста и снизить нагрузку на центральные и графические процессоры. Mistral специализируется на использовании параллельных вычислений и сегментированного обучения, что даёт преимущество при работе с длинными последовательностями текста [6]. Эта модель также имеет повышенную устойчивость к отсутствию крупных вычислительных ресурсов, благодаря чему её можно использовать на локальных серверах и ПК с относительно низкими характеристиками.

- Модель Gemma является ещё одним примером локальной языковой модели, ориентированной на эффективность и компактность [7].

Особенностью Gemma является использование гибридной архитектуры, которая сочетает элементы трансформера и рекуррентных сетей, что позволяет снизить потребление ресурсов и увеличить скорость генерации текста. Gemma также включает методы адаптивного обучения, позволяющие модели эффективно подстраиваться под различные задачи с минимальными изменениями в структуре. Это делает её более универсальной и подходящей для широкого круга задач, в том числе для игры в ассоциации.

3.2. Требования к ресурсам

Все три модели разработаны с учетом снижения требований к ресурсам по сравнению с крупными языковыми моделями. Однако для их эффективного функционирования необходима минимальная конфигурация ПК или сервера, включающая:

- не менее 8 ГБ оперативной памяти для стабильной работы;
- графический процессор с поддержкой CUDA для ускорения вычислений (рекомендуется для LLaMA и Mistral);
- процессор с многопоточностью (желательно от 4 ядер и выше), так как это улучшает параллельную обработку данных в Mistral.

LLaMA, Mistral и Gemma могут эффективно работать на ПК с вышеуказанными характеристиками, что делает их более доступными для индивидуального использования и менее зависимыми от специализированного оборудования [8].

3.3. Применимость для игры в ассоциации

Использование локальных языковых моделей для задачи ассоциаций требует способности модели выявлять связи между словами, что возможно, если модель обучена на большом количестве разнообразных текстов. Теоретически, LLaMA, Mistral и Gemma могут справляться с этой задачей, так как они используют вероятностные подходы и механизм внимания для анализа контекста, что позволяет им определять, какие слова могут быть логически связаны с другими.

Модели, как правило, обучены на множестве текстов, в которых встречаются тематические, логические и даже культурные ассоциации, и это позволяет им генерировать ассоциативные ряды для заданных слов. Однако важно учитывать, что в отличие от крупных языковых моделей, локальные модели могут иметь ограниченный словарный запас и меньшую степень генеративной способности. Это может снизить точность предсказаний в задачах, где требуются глубокие и абстрактные ассоциации.

Несмотря на эти ограничения, локальные модели, такие как LLaMA, Mistral и Gemma, подходят для создания базовых и средне сложных ассоциативных связей, особенно если модель настроена на выявление стандартных и распространённых связей. Таким образом, локальные модели можно эффективно использовать для задачи ассоциаций, если приоритетом является экономия ресурсов и умеренная точность ассоциативных связей.

Перед запуском моделей необходимо установить Python 3.7 или выше, а также библиотеки torch и transformers, которые поддерживают загрузку и использование языковых моделей. Для ускорения вычислений рекомендуется наличие графического процессора (GPU) с поддержкой CUDA.

3.3.1. Проверка модели LLaMA

Модель LLaMA (Large Language Model Meta AI) была разработана командой Meta AI и доступна через платформу Hugging Face. Мы использовали версию модели LLaMA-7B, где “7B” обозначает количество параметров модели – 7 миллиардов. Это говорит о том, что модель является компактной по сравнению с крупными языковыми моделями, которые могут иметь сотни миллиардов параметров, но при этом достаточно мощной, чтобы справляться с широким спектром задач. Модель LLaMA-7B оптимизирована для локального использования, что делает её привлекательной для приложений с ограниченными вычислительными ресурсами.

LLaMA-7B была выбрана для экспериментов из-за её баланса между производительностью и требованиями к ресурсам. Она может эффективно работать на компьютерах с 16 ГБ оперативной памяти и графическими

процессорами среднего уровня, что позволяет исследователям и разработчикам запускать её локально без необходимости в специализированном оборудовании.

Для использования модели необходимо получить доступ к репозиторию модели на Hugging Face, а также получить токен для аутентификации на платформе. Далее загрузите модель LLaMA-7B и её токенизатор, используя библиотеку transformers. Ниже приведён код для генерации ассоциаций с использованием модели LLaMA-7B.

Листинг 2 – Программа для генерации ассоциаций с использованием модели LLaMa-7B

```
from transformers import AutoModelForCausalLM, AutoTokenizer
from huggingface_hub import login
import time

# Токен Hugging Face
token = "hf_cUEX********************VkbOjtZh"

# Загрузка токенизатора и модели LLaMA
tokenizer = AutoTokenizer.from_pretrained("huggyllama/llama-7b",
use_auth_token=token)
model = AutoModelForCausalLM.from_pretrained("huggyllama/llama-7b",
use_auth_token=token)

def generate_associations_llama(word):
    # Упрощенный и конкретный запрос
    prompt = f"Name 10 words related to '{word}':"
    inputs = tokenizer(prompt, return_tensors="pt")

    # Начало отсчета времени
    start_time = time.time()

    outputs = model.generate(
        inputs['input_ids'],
        max_length=50,  # Уменьшение длины для предотвращения избыточной
генерации
        eos_token_id=tokenizer.eos_token_id
    )

    # Конец отсчета времени
    end_time = time.time()

    response = tokenizer.decode(outputs[0], skip_special_tokens=True)
    response = response.replace(prompt, "").strip()
```

```
    # Обрезаем текст после 10 ассоциаций
    response = "\n".join(response.split("\n")[:10])

    generation_time = end_time - start_time
    return response, generation_time

# Пример использования
word = "summer"
response, generation_time = generate_associations_llama(word)
print("LLaMA Ассоциации для слова:", word)
print(response)
print(f"Время генерации ответа: {generation_time:.2f} секунд")
```

В ходе тестирования модели LLaMA-7B для генерации ассоциаций для слова “summer” были выявлены несколько важных наблюдений, которые помогли оценить её способность решать задачу и выявить некоторые ограничения.

- В одном из результатов модель вывела повторяющееся слово (“summer”) в разных позициях списка, что говорит о трудностях в интерпретации задания. Это может быть вызвано слишком конкретной или неоднозначной формулировкой запроса. Чтобы улучшить качество ответа, мы модифицировали запрос, сделав его более простым и прямым.
- В другом случае модель попыталась интерпретировать запрос как указание на создание более длинного текста, включающего описание концепций лета и зимы, сгенерировав кусочек: “Now, name 10 words related to c 'winter': 1. snow 2. ice 3. Frost” после ассоциаций для слова “summer”. Это произошло потому, что модель пыталась следовать шаблону, создавая текст вместо того, чтобы просто перечислить слова. Чтобы избежать этого, мы добавили ограничения по длине ответа и упростили запрос, что позволило модели сосредоточиться на генерации ассоциаций.
- На генерацию ответа для каждого запроса модель LLaMA-7B затрачивала в среднем около 30-95 секунд, что относительно медленно для задачи ассоциаций. Такая задержка связана с необходимыми вычислениями на

модели с 7 миллиардами параметров. Время генерации также зависело от сложности запроса и длины полученного ответа.

- При работе с простыми словами, такими как “summer” (лето), модель обычно генерировала ассоциации вроде “beach”, “sun”, “sea”, “holiday”. Однако при работе с более абстрактными понятиями (например, “freedom”) результаты оказались ограниченными и менее релевантными. Это связано с ограничениями данных, на которых обучалась модель, и её параметрами, которые не всегда позволяют получать разнообразные ассоциации для сложных понятий.

Эксперименты показали, что модель LLaMA-7B способна генерировать базовые ассоциации для простых понятий, но имеет некоторые ограничения при работе с более сложными задачами, такими как абстрактные ассоциации. Основные сложности включают:

- Повторение слов в ответах;
- Генерацию более широких текстов, чем требовалось;
- Ограниченные возможности по креативным ассоциациям.

В целом, LLaMA-7B показала хорошие результаты для создания стандартных ассоциаций, связанных с конкретными понятиями, но ограниченные возможности для сложных и нестандартных задач. В будущем для улучшения качества ассоциаций и увеличения скорости работы можно попробовать использовать модели с меньшим количеством параметров или дополнительно настроить параметры генерации и текстовые подсказки.

3.3.2. Проверка модели Mistral

Для проведения эксперимента мы использовали модель Mistral-7B-Instruct-v0.3 – это версия модели Mistral-7B, дообученная на инструкциях, что позволяет ей лучше справляться с задачами, требующими понимания контекста инструкций. Модель Mistral-7B разработана командой Mistral AI, а версия “Instruct” дополнительно оптимизирована для взаимодействия с

пользователем, что делает её подходящей для генерации ассоциаций в ответ на конкретные запросы. Данная модель, как и LLaMa, также содержит 7 миллиардов параметров, что даёт ей возможность предоставлять точные и разнообразные ответы, оставаясь при этом сравнительно компактной по сравнению с более крупными моделями.

Модель оптимизирована для экономии памяти и может быть развернута локально на компьютере с достаточной оперативной памятью и графическим процессором средней мощности. Mistral-7B-Instruct также поддерживает расширенный словарь и обладает более продвинутыми возможностями работы с функциями, что позволяет более гибко взаимодействовать с запросами пользователей. Ниже приведён код для генерации ассоциаций с использованием модели Mistral-7B-Instruct-v0.3.

Листинг 3 – Программа для генерации ассоциаций с использованием модели Mistral-7B-Instruct-v0.3

```
from transformers import AutoModelForCausalLM, AutoTokenizer
from huggingface_hub import login
import time

# Токен Hugging Face
token = "hf_cUEX********************VkbOjtZh"

# Загрузка токенизатора и модели Mistral-7B-Instruct-v0.3
tokenizer = AutoTokenizer.from_pretrained("mistralai/Mistral-7B-Instruct-v0.3",
token=token)
model = AutoModelForCausalLM.from_pretrained("mistralai/Mistral-7B-Instruct-
v0.3", token=token)

# Устанавливаем pad_token_id на eos_token_id, если pad_token отсутствует
if tokenizer.pad_token_id is None:
    tokenizer.pad_token_id = tokenizer.eos_token_id

def generate_associations_mistral(word):
    prompt = f"List exactly 10 words associated with '{word}', and stop there."
    inputs = tokenizer(prompt, return_tensors="pt", padding=True)

    # Создаем attention mask
    attention_mask = inputs['attention_mask']

    # Начало отсчета времени
    start_time = time.time()
```

```
    outputs = model.generate(
        inputs['input_ids'],
        attention_mask=attention_mask,  # Передача attention mask
        max_length=150,  # Ограничение длины для предотвращения лишней генерации
        eos_token_id=tokenizer.eos_token_id,
        pad_token_id=tokenizer.pad_token_id  # Установка pad_token_id
    )

    # Конец отсчета времени
    end_time = time.time()

    response = tokenizer.decode(outputs[0], skip_special_tokens=True)
    response = response.replace(prompt, "").strip()

    # Вычисляем время генерации
    generation_time = end_time - start_time

    return response, generation_time

# Пример использования
word = "ocean"
response, generation_time = generate_associations_mistral(word)
print("Mistral Ассоциации для слова:", word)
print(response)
print(f"Время генерации ответа: {generation_time:.2f} секунд")
```

При тестировании модели Mistral-7B-Instruct на задаче генерации ассоциаций были выявлены несколько интересных особенностей, которые помогают понять её возможности и ограничения.

- При запросе ассоциаций для простых слов, таких как “ocean”, модель сгенерировала релевантные ассоциации, включая слова “Waves”, “Beach”, “saltwater”, “coral reef”, “Fish”, и другие. Это показывает, что модель хорошо справляется с простыми задачами ассоциаций для конкретных понятий, предоставляя набор логически связанных слов.
- В некоторых тестах модель переходила к расширению ответа после завершения списка, добавляя описание или комментарии, такие как “Now, let's talk about the ocean's role in climate change”. Это поведение связано с тем, что Mistral-7B-Instruct оптимизирована для инструктивных ответов и, следовательно, может неправильно интерпретировать намерение запроса, добавляя текст, выходящий за рамки простой задачи ассоциаций. Для решения

этой проблемы мы добавили в запрос конкретные инструкции на завершение списка (“and end the list here”), что уменьшило вероятность дополнительных текстов.

- Среднее время генерации ответа для модели Mistral-7B-Instruct составило около 77–110 секунд, что является сравнительно высоким показателем для простой задачи. Это может быть связано с тем, что модель генерирует дополнительные ответы или рассматривает запросы как более сложные задания, требующие расширенного объяснения.

Модель Mistral-7B-Instruct продемонстрировала хорошие результаты в задаче генерации ассоциаций для конкретных и распространённых понятий. Однако её склонность к расширению ответа за рамки ожидаемого списка создала некоторые трудности, особенно для задач, где требуются исключительно краткие и чёткие ответы. В целом:

- Модель надёжно справляется с запросами на генерацию списков слов, связанных с определёнными терминами.
- Mistral-7B-Instruct может интерпретировать запрос как инструкцию на развернутый ответ, добавляя ненужные объяснения.
- Уменьшение max_length, а также добавление явных инструкций на завершение списка помогли уменьшить количество дополнительных текстов.

В дальнейшем можно рассмотреть возможность адаптации модели для более точного выполнения задач ассоциаций либо использовать её для задач, где требуется более развернутая генерация ответов, что соответствует её инструктивной природе.

3.3.3. Проверка модели Gemma

Модель Gemma – это семейство лёгких и современных открытых языковых моделей, разработанных компанией Google на основе тех же исследований и технологий, которые лежат в основе моделей Gemini. Для проведения анализа была выбрана модель Gemma-2-2B, которая является компактной (содержит 2 миллиарда параметров) и предназначена для

текстовых задач. Данная модель представляет собой модель с архитектурой “только декодер” и была разработана для работы на устройствах с ограниченными вычислительными ресурсами, таких как ноутбуки и настольные компьютеры. Это делает её идеальным выбором для приложений, где необходим баланс между производительностью и доступностью.

Gemma-2-2B обладает широкими возможностями в области генерации текста и подходит для выполнения различных задач, таких как: ответы на вопросы; суммаризация текста; обработка запросов на естественном языке и рассуждение.

Данная модель разработана с открытыми весами и доступна как в базовой версии, так и в версии, дообученной на инструкциях (instruction-tuned), что делает её универсальной для различных текстовых задач. Основная задача модели – поддержание высоких стандартов генеративного ИИ с учётом безопасности, поэтому в процессе подготовки данных были применены строгие фильтры для исключения вредоносного контента.

Запуск модели и проверка на задаче ассоциаций были реализованы похожим образом, как и у двух предыдущих языковых моделей. Ниже приведён код для генерации ассоциаций с использованием модели Gemma-2-2B.

Листинг 4 – Программа для генерации ассоциаций с использованием модели Gemma-2-2B

```
from transformers import AutoModelForCausalLM, AutoTokenizer
from huggingface_hub import login
import time

# Токен Hugging Face
token = "hf_cUEX********************VkbOjtZh"

# Загрузка токенизатора и модели Gemma-2-2B
tokenizer = AutoTokenizer.from_pretrained("google/gemma-2-2b", token=token)
model = AutoModelForCausalLM.from_pretrained("google/gemma-2-2b", token=token)

# Устанавливаем pad_token_id на eos_token_id, если pad_token отсутствует
if tokenizer.pad_token_id is None:
    tokenizer.pad_token_id = tokenizer.eos_token_id
```

```
def generate_associations_gemma(word):
    prompt = (
        f"[Question]: List exactly 10 words associated with '{word}' and stop
after the list.\n"
        "[Answer]: "
    )
    inputs = tokenizer(prompt, return_tensors="pt", padding=True)

    # Создаем attention mask
    attention_mask = inputs['attention_mask']

    # Начало отсчета времени
    start_time = time.time()

    outputs = model.generate(
        inputs['input_ids'],
        attention_mask=attention_mask,  # Передача attention mask
        max_length=100,  # Умеренное ограничение длины
        eos_token_id=tokenizer.eos_token_id,
        pad_token_id=tokenizer.pad_token_id  # Установка pad_token_id
    )

    # Конец отсчета времени
    end_time = time.time()

    response = tokenizer.decode(outputs[0], skip_special_tokens=True)
    response = response.replace(prompt, "").strip()

    # Удаляем возможные HTML-теги
    response = response.replace("<strong>", "").replace("</strong>", "")

    # Вычисляем время генерации
    generation_time = end_time - start_time

    return response, generation_time

# Пример использования
word = "forest"
response, generation_time = generate_associations_gemma(word)
print("Gemma Ассоциации для слова:", word)
print(response)
print(f"Время генерации ответа: {generation_time:.2f} секунд")
```

В процессе тестирования модели для задачи ассоциаций были выявлены несколько особенностей и ограничений:

- В некоторых тестах модель генерировала список, в котором повторялись одни и те же слова. Например, при запросе ассоциаций для слова "forest" в

результатах многократно появлялось слово "woodland". Этот повтор может свидетельствовать о том, что модель пытается закончить список, используя наиболее вероятное слово, вместо того чтобы предлагать уникальные ассоциации. Чтобы решить эту проблему, был добавлен постпроцессинг для фильтрации дубликатов.

- В некоторых случаях модель вместо списка генерировала более длинные и рефлексивные ответы, как, например, при запросе для слова "yellow". В этом случае модель выдала введение и начала рассуждение, не ограничиваясь списком ассоциаций. Это поведение можно объяснить её архитектурой, ориентированной на инструкции, и попыткой интерпретировать запрос как задание на развернутый ответ. Для улучшения результатов мы изменили запрос, добавив имитацию формата "чат", чтобы модель ограничилась кратким ответом в форме списка.
- Введение эмуляции чата оказалось эффективным решением для управления длиной и форматом ответа. При подаче запроса в формате «[User]:... [Bot]:...», модель начала лучше воспринимать задачу как запрос на краткий список ассоциаций. Это помогло сократить избыточное содержание и снизить склонность модели к созданию развернутых текстов вместо простого перечня слов.
- В некоторых случаях модель добавляла HTML-разметку (), что указывает на её склонность к структурированным ответам. Это было устранено в процессе постобработки вывода, удаляя такие теги.
- Время генерации для каждого запроса составило в среднем от 20 до 30 секунд, что является умеренным показателем для модели такого размера. Временные задержки варьировались в зависимости от сложности и длины запроса, однако оставались в пределах приемлемого диапазона для задачи генерации ассоциаций.

Модель Gemma-2-2B показала хорошие результаты для генерации ассоциаций, хотя и столкнулась с трудностями, связанными с дублированием

слов и избыточными объяснениями. Основные выводы можно подытожить следующим образом:

- Модель успешно справляется с генерацией списков ассоциаций для простых понятий, особенно после введения формата чата и ограничения на длину вывода.
- Для абстрактных понятий и открытых инструкций модель может склоняться к созданию длинных текстов вместо конкретных ответов.
- Использование постобработки для удаления дубликатов и установка чётких ограничений в запросе помогают улучшить точность и релевантность ответа.

В дальнейшем модель Gemma-2-2B может быть успешно применена для задач, требующих структурированных ответов и списков ассоциаций, особенно если используется дополнительная обработка вывода и чёткие формулировки запросов.

3.4. Общий анализ результатов

Проведённое тестирование локальных языковых моделей LLaMA-7B, Mistral-7B и Gemma-2B продемонстрировало как их преимущества, так и ограничения в решении задачи генерации ассоциаций. Эти модели, разработанные для локального развертывания с относительно небольшими требованиями к ресурсам, отличаются возможностью использования на устройствах с ограниченными вычислительными мощностями, такими как ноутбуки и настольные компьютеры. При этом каждая из моделей показала свою уникальную специфику и характерные особенности, которые влияют на качество и точность генерируемых ассоциативных рядов.

Подводя итог, можно отметить, что все три модели имеют потенциал для использования в задачах генерации ассоциаций, но требуют тщательной настройки запросов и постобработки. Для более конкретных и структурированных задач локальные модели LLaMA-7B, Mistral-7B и Gemma-2B обеспечивают адекватное качество и могут использоваться без облачной

инфраструктуры. Однако для получения более креативных и разнообразных ассоциаций, а также для работы с абстрактными понятиями, потребуется либо дополнительная адаптация моделей, либо выбор более крупных языковых моделей, способных глубже понимать ассоциативные связи.

4. Модель 2txt

Модель 2txt предназначена для быстрой генерации текстового описания изображений. Созданная с использованием SDK от Vercel и технологии Anthropic, она оптимизирована для высокоскоростной генерации текста и упрощает процесс визуальных ассоциаций.

4.1. Установка и запуск модели

Для установки и настройки модели 2txt необходимо выполнить несколько шагов. Прежде всего, требуется доступ к репозиторию проекта, размещённому на платформе GitHub, где содержатся все необходимые для установки файлы и документация.

Шаги установки:

- Для начала необходимо склонировать репозиторий на локальный компьютер, используя команду:

```
git clone https://github.com/ai-ng/2txt.git
```

- В корневой директории проекта создаётся файл .env.local, в котором указывается API-ключ Anthropic. Этот ключ можно получить, зарегистрировавшись на платформе Anthropic и создав его в консоли разработчика. Пример содержимого файла:

```
ANTHROPIC_API_KEY=your-api-key
```

Здесь your-api-key заменяется на фактический API-ключ от Anthropic.

- После настройки файла конфигурации следует установить все необходимые библиотеки и зависимости, используя пакетный менеджер pnpm. Выполните команду:

```
pnpm install
```

Этот процесс установит все зависимости, указанные в проекте, включая Vercel AI SDK, Next.js и библиотеки для работы с API Anthropic.

- После завершения установки зависимостей можно запустить сервер разработки командой:

```
pnpm dev
```

Сервер запустится в режиме разработки, что позволит тестировать работу модели и изменять её параметры в реальном времени (рис. 1).

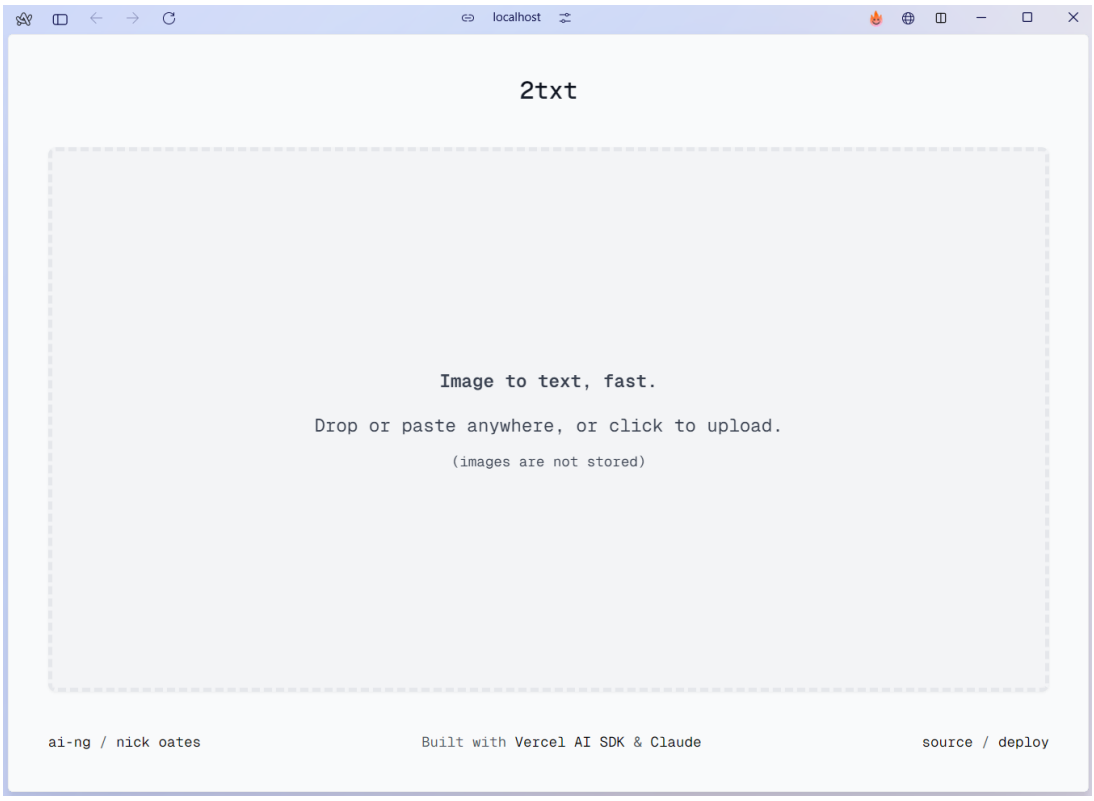

Рисунок 1 – Запущенная и готовая к работе модель 2txt

После завершения установки и настройки модель 2txt готова к использованию. С помощью API Anthropic и оптимизированного окружения на базе Vercel она обеспечивает быструю генерацию текстовых ассоциаций для изображений.

4.2. Тестирования на задаче визуальных ассоциаций

Для оценки эффективности модели 2txt были проведены эксперименты с различными изображениями, целью которых являлось проверка точности и качества генерируемых описательных текстов. В качестве тестового набора данных были выбраны изображения с различными визуальными элементами,

чтобы оценить, насколько точно модель может интерпретировать визуальный контекст и генерировать описание, соответствующее сцене.

В первом тесте для модели было загружено изображение, на котором изображён ленивец, висящий на ветке дерева (рис. 2). Ожидаемый результат описания заключался в том, что модель должна была правильно интерпретировать и описать ленивца и его позу.

Рисунок 2 – Результат работы модели 2txt с картинкой

Результат тестирования показал, что модель 2txt успешно распознала изображение и точно определила его содержимое, описав ленивца на дереве. Описание получилось лаконичным, соответствующим содержанию изображения и подходящим для использования в задачах ассоциаций. Модель корректно идентифицировала объект и представила его в понятной форме, что свидетельствует о её способности обрабатывать простые визуальные сцены.

Во втором тесте было загружено изображение, на котором изображён стакан воды на фоне горного пейзажа. В данном случае целью было проверить

способность модели идентифицировать не только объект на переднем плане (стакан с водой), но и фоновые элементы изображения (горы и озеро).

Рисунок 3 – Результат работы модели 2txt с картинкой

Модель 2txt продемонстрировала способность к детализированному описанию изображения, включая информацию о фоне и переднем плане. В описании были правильно указаны как основной объект изображения – стакан с водой, так и контекст окружающего ландшафта, что свидетельствует о высоком качестве генерации ассоциаций для сложных изображений.

Результаты тестирования показали, что модель 2txt способна создавать точные и релевантные текстовые описания для визуальных объектов, что делает её пригодной для использования в задачах визуальных ассоциаций. Основные выводы по результатам тестирования можно сформулировать следующим образом:

- Модель 2txt надёжно генерирует описания для изображений с простыми и сложными объектами.

- Генерируемые тексты имеют высокую релевантность и соответствуют контексту изображения, что свидетельствует о потенциале модели для задач визуальных ассоциаций.
- Тесты подтверждают, что модель способна успешно обрабатывать как основные объекты на изображении, так и задний план, создавая детализированные описания.

Таким образом, модель 2txt показала себя как эффективное средство для генерации ассоциаций на основе изображений, демонстрируя хорошую точность и релевантность описаний.

ЗАКЛЮЧЕНИЕ

Выполненная работа была направлена на исследование возможностей современных крупных и локальных языковых моделей в решении задачи ассоциаций. Процесс исследования охватил широкий спектр задач и методов, включая анализ работы известных языковых моделей, таких как GPT-4, LLaMA, Mistral, и Gemma, их установку и тестирование на задаче ассоциаций. Также была выполнена практическая реализация и настройка модели 2txt для генерации визуальных ассоциаций с изображениями, что позволило получить более полное представление о возможностях обработки различных типов данных в контексте ассоциативных задач.

Основной результат работы состоит в том, что крупные языковые модели, такие как GPT-4, показали высокую способность находить ассоциативные ряды благодаря глубине анализа и охвату сложных взаимосвязей между словами. Их архитектура и алгоритмы позволяют моделям с высокой степенью точности предсказывать ассоциации даже для абстрактных понятий, что делает их полезными для задач, требующих высокого уровня креативности и понимания контекста. Однако их применение ограничено высоким потреблением вычислительных ресурсов и необходимостью доступа к мощной облачной инфраструктуре, что снижает их доступность для индивидуальных пользователей и небольших проектов.

В ходе анализа локальных языковых моделей, таких как LLaMA, Mistral и Gemma, было установлено, что они обладают достаточным потенциалом для выполнения задач ассоциаций, особенно если цель заключается в создании простых и предсказуемых ассоциативных связей. Эти модели, благодаря своей оптимизированной архитектуре, подходят для локального развертывания и демонстрируют хорошие результаты при меньших ресурсных затратах. В то же время, локальные модели имеют ограничения по объему знаний и креативности, что особенно заметно при работе с абстрактными и многозначными понятиями. Тем не менее, при должной настройке запросов и

параметров, локальные модели могут быть адаптированы для создания базовых ассоциативных рядов, что делает их экономичным и практичным решением для определённых типов задач.

Эксперименты с моделью 2txt для задачи визуальных ассоциаций показали её пригодность для генерирования описаний изображений, что существенно расширяет возможности использования ассоциативного подхода в задачах обработки визуальной информации. Модель продемонстрировала способность распознавать и описывать как основные объекты на изображении, так и задний план, что открывает перспективы её использования в областях, требующих быстрого анализа и описания визуального контекста, таких как мультимедийные системы и образовательные платформы. Работа модели подтвердила возможность успешной интерпретации изображений с достаточно высоким уровнем точности, что делает её полезным инструментом для интеграции в приложения с поддержкой визуальных данных.

Таким образом, в ходе работы были достигнуты поставленные цели, включающие исследование различных моделей для задачи ассоциаций и создание собственного подхода к решению данной задачи. Анализ продемонстрировал преимущества и ограничения как крупных языковых моделей, так и локальных решений, что позволило сформулировать выводы о целесообразности их использования в зависимости от требований задачи. Полученные результаты и выводы создают прочную основу для дальнейших исследований в области минималистичных моделей для ассоциативных задач и разработки эффективных, экономичных решений, ориентированных на конкретные прикладные цели.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Ben Lutkevich. Language modeling // TechTarget [Электронный ресурс]. Режим доступа: https://www.techtarget.com/searchenterpriseai/definition/language-modeling (дата обращения 20.05.2024).

2. Language Models, Explained: How GPT and Other Models Work // Altexsoft [Электронный ресурс]. Режим доступа: https://www.altexsoft.com/blog/language-models-gpt/ (дата обращения 20.05.2024).

3. Fawad Ali. GPT-1 to GPT-4: Each of OpenAI's GPT Models Explained and Compared // MakeUseOf [Электронный ресурс]. Режим доступа: https://www.makeuseof.com/gpt-models-explained-and-compared/ (дата обращения 21.05.2024).

4. T. Brown et al. Language Models are Few-Shot Learners // OpenAI [Электронный ресурс]. Режим доступа: https://arxiv.org/abs/2005.14165 (дата обращения 25.05.2024).

5. Meta AI. LLaMA: Open and Efficient Foundation Language Models // Meta AI Research [Электронный ресурс]. Режим доступа: https://ai.facebook.com/research/publications/llama-open-and-efficient-foundation-language-models/ (дата обращения 22.05.2024).

6. Mistral AI Team. Mistral 7B Model Overview // Hugging Face [Электронный ресурс]. Режим доступа: https://huggingface.co/mistralai/Mistral-7B (дата обращения 23.06.2024).

7. Google Research. Gemini Language Models // Google AI Blog [Электронный ресурс]. Режим доступа: https://ai.googleblog.com/2023/06/gemini-language-models.html (дата обращения 22.05.2024).

8. Hugging Face Transformers Documentation // Hugging Face [Электронный ресурс]. Режим доступа: https://huggingface.co/docs/transformers (дата обращения 23.06.2024).