

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

**ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ**

**ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ  
«ДОНСКОЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»**

**(ДГТУ)**

**ОТЧЕТ**

**по практической подготовке**

**в рамках реализации проекта «Цифровая кафедра»**

|  |  |
| --- | --- |
| Обучающийся \_\_\_\_\_\_Полуян Юлия Сергеевна\_\_\_\_  (Ф.И.О.) | \_\_\_\_\_\_ВКБ33\_\_\_\_\_\_\_\_\_  (группа) |
| Направление подготовки 10.05.01 Компьютерная безопасность  (шифр, наименование) | Информатика и вычислительная техника  (факультет) |
| Программа обучения \_\_\_Искусственный интеллект и машинное обучение\_\_\_\_\_\_\_\_\_\_\_\_\_\_  (наименование ДПП ПП) | |
| Наименование места практической подготовки \_\_\_\_ФГАНУ НИИ «Спецвузавтоматика»\_\_\_\_\_\_\_\_  (название организации) | |
| Период прохождения практики с21.04.2025 г. по 17.05.2025 г. | |
| Индивидуальное/групповое (*подчеркнуть нужное*) задание:  Telegramм бот ‘RecipeAI’: инструмент для нахождения рецептов\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ | |
| Практические результаты:  Разработан промышленный Telegram-бот RecipeAI, реализующий интеллектуальную систему поиска рецептов с интеграцией NLP-моделей (Sentence Transformers) для семантического анализа запросов и векторного поиска по датасету 50 000+ рецептов, обеспечивающий многоуровневую фильтрацию по ингредиентам и релевантности с учетом диетических ограничений в реальных условиях эксплуатации. | |

|  |  |  |
| --- | --- | --- |
| Обучающийся | \_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_  (дата сдачи отчета) (подпись) | Полуян Ю. С.  (Ф.И.О.) |
|  |  |  |
| Руководитель практической подготовки от ДГТУ | \_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  (оценка) (подпись, дата) | доцент, Криворучко А. В.  (должность, ФИО.) |

ДНЕВНИК ПРОХОЖДЕНИЯ ПРАКТИКИ

|  |  |  |  |
| --- | --- | --- | --- |
| Дата | База практики | Выполняемые работы | Оценка руководителя |
| 21.04.2025 | ФГАНУ НИИ «Спецвузавтоматика» | Знакомство с предприятием, прохождение вводного инструктажа |  |
| 22.04.2025 | ФГАНУ НИИ «Спецвузавтоматика» | Получение индивидуального (группового) задания и постановка задачи. |  |
| 23.04.2025-  28.04.2025 | ФГАНУ НИИ «Спецвузавтоматика» | Аналитический обзор предметной области |  |
| 29.04.2025 – 12.05.2025 | ФГАНУ НИИ «Спецвузавтоматика» | Программная реализация |  |
| 13.05.2025 – 16.05.2025 | ФГАНУ НИИ «Спецвузавтоматика» | Подготовка и оформление отчета по практике |  |
| 17.05.2025 | ФГАНУ НИИ «Спецвузавтоматика» | Сдача итогового отчета |  |

Руководитель практической подготовки от ДГТУ:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Доцент |  |  |  | Криворучко А. В. |
|  |  | *подпись, дата* |  |  |



МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

**ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ**

**ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ  
«ДОНСКОЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»**

**(ДГТУ)**

**ОТЧЕТ**

**по практической подготовке**

**в рамках реализации проекта «Цифровая кафедра»**

|  |  |
| --- | --- |
| Обучающийся \_\_\_\_\_\_Ковалец Илья Андреевич\_\_\_\_  (Ф.И.О.) | \_\_\_\_\_\_ВКБ33\_\_\_\_\_\_\_\_\_  (группа) |
| Направление подготовки 10.05.01 Компьютерная безопасность  (шифр, наименование) | Информатика и вычислительная техника  (факультет) |
| Программа обучения \_\_\_Искусственный интеллект и машинное обучение\_\_\_\_\_\_\_\_\_\_\_\_\_\_  (наименование ДПП ПП) | |
| Наименование места практической подготовки \_\_\_\_\_\_Отдел систем технического контроля\_\_  (название организации) | |
| Период прохождения практики с21.04.2025 г. по 17.05.2025 г. | |
| Индивидуальное/групповое (*подчеркнуть нужное*) задание:  Telegramм бот ‘RecipeAI’: инструмент для нахождения рецептов\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ | |
| Практические результаты:  Разработан промышленный Telegram-бот RecipeAI, реализующий интеллектуальную систему поиска рецептов с интеграцией NLP-моделей (Sentence Transformers) для семантического анализа запросов и векторного поиска по датасету 50 000+ рецептов, обеспечивающий многоуровневую фильтрацию по ингредиентам и релевантности с учетом диетических ограничений в реальных условиях эксплуатации. | |

|  |  |  |
| --- | --- | --- |
| Обучающийся | \_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_  (дата сдачи отчета) (подпись) | Ковалец И.А.  (Ф.И.О.) |
|  |  |  |
| Руководитель практической подготовки от ДГТУ | \_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  (оценка) (подпись, дата) | доцент, Криворучко А. В.  (должность, ФИО.) |

ДНЕВНИК ПРОХОЖДЕНИЯ ПРАКТИКИ

|  |  |  |  |
| --- | --- | --- | --- |
| Дата | База практики | Выполняемые работы | Оценка руководителя |
| 21.04.2025 | Отдел систем технического контроля | Знакомство с предприятием, прохождение вводного инструктажа |  |
| 22.04.2025 | Отдел систем технического контроля | Получение индивидуального (группового) задания и постановка задачи. |  |
| 23.04.2025-  28.04.2025 | Отдел систем технического контроля | Аналитический обзор предметной области |  |
| 29.04.2025 – 12.05.2025 | Отдел систем технического контроля | Программная реализация |  |
| 13.05.2025 – 16.05.2025 | Отдел систем технического контроля | Подготовка и оформление отчета по практике |  |
| 17.05.2025 | Отдел систем технического контроля | Сдача итогового отчета |  |

Руководитель практической подготовки от ДГТУ:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Доцент |  |  |  | Криворучко А. В. |
|  |  | *подпись, дата* |  |  |

**Содержание**

[Введение 6](#_Toc199553792)

[1 Аналитический обзор предметной области 8](#_Toc199553793)

[1.1 Обзор предметной области 8](#_Toc199553794)

[1.2 Постановка задачи 9](#_Toc199553795)

[1.3 Выводы по главе 10](#_Toc199553796)

[2 Анализ моделей машинного обучения 12](#_Toc199553797)

[2.1 Описание датасета List\_of\_Recipes 12](#_Toc199553798)

[2.2 Модель paraphrase-multilingual-MiniLM-L12-v2 14](#_Toc199553799)

[3 Алгоритмическое и программное конструирование 18](#_Toc199553800)

[3.1 Общий алгоритм работы приложения 18](#_Toc199553801)

[3.2 Алгоритм применения обученных моделей 19](#_Toc199553802)

[3.3 Выбор и обоснование инструментальных средств 19](#_Toc199553803)

[3.4 Программная реализация 20](#_Toc199553804)

[4 Демонстрация работы приложения 22](#_Toc199553805)

[Заключение 27](#_Toc199553806)

[Приложение Листинг программы 29](#_Toc199553807)

Введение

В современном мире кулинария занимает особое место, объединяя традиции, творчество и повседневные потребности человека. С развитием технологий и ускорением ритма жизни многие люди сталкиваются с проблемой выбора блюд и поиска подходящих рецептов. Нехватка времени, ограниченный набор продуктов под рукой, пищевые предпочтения и диетические ограничения – всё это усложняет процесс приготовления пищи. В то же время растёт спрос на персонализированные кулинарные решения, которые помогут быстро подобрать рецепт, адаптировать его под конкретные запросы и вдохновить на эксперименты на кухне.

Telegram-бот с обширной базой рецептов предлагает удобное и современное решение этой задачи. Он позволяет пользователям в несколько кликов получать доступ к разнообразным блюдам – от простых повседневных рецептов до сложных кулинарных шедевров. Такой сервис может быть полезен не только для домашнего использования, но и в профессиональной сфере. Например, владельцы кафе и ресторанов могут использовать его для быстрого поиска идей при составлении меню, а фуд-блогеры – для вдохновения и создания нового контента. Сервисы доставки еды могут интегрировать подобные боты в свои платформы, чтобы предлагать клиентам персонализированные рекомендации на основе их предпочтений.

Использование искусственного интеллекта и структурированных датасетов открывает дополнительные возможности для подобных сервисов. Алгоритмы могут анализировать вкусовые предпочтения пользователей, учитывать их диетические ограничения (например, вегетарианство, безглютеновое или низкоуглеводное питание) и предлагать оптимальные варианты блюд. В перспективе такие системы могут быть интегрированы с "умной" кухонной техникой – например, автоматически отправлять рецепты в мультиварку или духовку с поддержкой IoT, что сделает процесс готовки ещё более удобным и технологичным.

Telegram-бот с базой рецептов – это не просто инструмент для поиска кулинарных идей, а многофункциональный помощник, способный адаптироваться под нужды пользователя. Он экономит время, расширяет кулинарные горизонты и способствует более осознанному подходу к питанию, что особенно актуально в условиях современного темпа жизни.

Аналитический обзор предметной области

В рамках данной главы проведем анализ предметной области, выполним постановку задач.

1.1 Обзор предметной области

Современный мир кулинарии стремительно переходит в цифровой формат, и это неудивительно - в эпоху, когда время становится самым ценным ресурсом, люди все чаще ищут удобные способы находить рецепты и планировать свое питание. Статистика показывает впечатляющий рост этого рынка: к 2025 году его объем превысит 3 миллиарда долларов [1]. За этими цифрами стоят реальные изменения в поведении людей - более 70% населения регулярно готовят дома, при этом все больше внимания уделяют здоровому питанию и персонализированному подходу к выбору блюд [2].

Технологический прогресс кардинально изменил наши кулинарные привычки. Голосовые помощники, умные кухонные устройства, мобильные приложения - все это стало неотъемлемой частью современной кухни. Однако, протестировав множество популярных кулинарных сервисов, осознаёшь, что большинство из них предлагают довольно ограниченный функционал. Они плохо понимают естественные запросы пользователей, не умеют искать по имеющимся продуктам, не адаптируют рецепты под нужное количество порций, а их интерфейсы часто неудобны для использования на мобильных устройствах.

Именно эти проблемы вдохновили на создание кулинарного бота в Telegram. Выбор платформы не случаен - Telegram сочетает в себе массовость (более 800 миллионов пользователей), удобство и привычный интерфейс [3]. Бот задуман как интеллектуальный помощник, который не просто выдает рецепты, а действительно понимает потребности пользователя. Он умеет анализировать имеющиеся продукты, адаптировать блюда под конкретные запросы, учитывать диетические ограничения и даже запоминать вкусовые предпочтения, чтобы предлагать персонализированные рекомендации.

Главная идея - создать не просто базу рецептов, а настоящего цифрового шефа, который сделает процесс готовки более простым, приятным и вдохновляющим. Бот должен стать тем самым виртуальным помощником, который всегда подскажет, что вкусного можно приготовить из того, что есть в холодильнике, и как сделать это быстро и вкусно. В эпоху, когда кулинария становится не просто необходимостью, но и формой творчества, такой инструмент может стать незаменимым спутником как для начинающих, так и для опытных кулинаров.

1.2 Постановка задачи

Целью проекта является создание интеллектуальной системы поиска и рекомендации кулинарных рецептов в формате Telegram-бота, сочетающей:

1. **Технологические инновации**:
   * Применение современных NLP-моделей
   * Векторные базы данных для эффективного поиска
   * Персонализированные рекомендательные алгоритмы
2. **Функциональную полноту**:
   * Многоуровневый поиск (по названию, ингредиентам, времени)
   * Разнообразие рецептов при схожести запроса
3. **Пользовательский опыт**:
   * Интуитивный чат-интерфейс
   * Пошаговая визуализация приготовления
   * Возможности социального взаимодействия

Технические требования к системе включают:

1. **Базовый функционал**:
   * Поиск рецептов по параметрам
   * Поддержка отправки рецептов
2. **Расширенные возможности**:
   * Выдача списка рецептов

Для реализации проекта необходимо решить следующие технические задачи:

1. **Алгоритмические**:
   * Разработка гибридной системы поиска (ключевые слова + векторный поиск)
   * Создание алгоритмов адаптации рецептов
   * Реализация рекомендательной системы
2. **Инфраструктурные**:
   * Оптимизация работы с большими объемами данных
   * Обеспечение отказоустойчивости
3. **Интерфейсные**:
   * Проектирование UX для Telegram-бота
   * Реализация интерактивных элементов
   * Адаптация под различные устройства
4. **Аналитические**:
   * Сбор пользовательских данных
   * Механизмы A/B-тестирования
   * Инструменты анализа эффективности

1.3 Выводы по главе

В результате проведенного анализа предметной области можно сделать вывод о значительном инновационном потенциале разрабатываемого кулинарного бота. В текущих рыночных условиях наблюдается устойчивый рост спроса на цифровые кулинарные решения, что открывает обширные перспективы для внедрения и улучшения существующих сервисов. Платформа Telegram, избранная для реализации проекта, обладает высоким потенциалом охвата аудитории, что позволяет уверенно рассчитывать на заинтересованность пользователей.

Современные технологии обработки естественного языка (NLP) предоставляют возможности для реализации интеллектуального поиска, что будет способствовать более точной и быстрой выдаче рекомендаций в ответ на запросы пользователей [4]. Векторные базы данных, используемые в архитектуре проекта, обеспечивают необходимую высокую производительность, а микросервисная архитектура гарантирует масштабируемость системы, что имеет критическое значение для удовлетворения растущих требований пользователей.

Пользовательский опыт играет ключевую роль в успехе кулинарного бота, и интеграция с мессенджером обеспечит мгновенный доступ к услугам, что в свою очередь повысит уровень удовлетворенности пользователей. Персонализация предложений также усилит интерес к продукту, а социальные функции будут способствовать вовлеченности и созданию активного сообщества вокруг приложения.

В целом, кулинарный бот сможет занять устойчивую позицию на рынке цифровых кулинарных решений, при условии комплексного подхода к его реализации, включающего передовые технологии машинного обучения, продуманный пользовательский опыт и эффективную бизнес-модель.

Анализ моделей машинного обучения

Для реализации системы поиска рецептов на основе пользовательского запроса была применена технология семантического поиска с использованием модели машинного обучения. Основными задачами являются поиск подходящего датасета и работа с ним, выбор необходимой модели, поддерживающей преобразование текстовых данных в числовые векторные представления, сохранение их и использование в работе.

**2.1 Описание датасета List\_of\_Recipes**

Для реализации проекта по созданию системы поиска рецептов был использован открытый датасет List\_of\_Recipes, взятый из репозитория на GitHub. Набор данных содержит более 50 000 уникальных рецептов, что делает его подходящим для обучения и тестирования моделей машинного обучения.

Исходные данные представлены в формате .csv и имеют такие столбцы с данными как «name», где указывается название рецепта, «ingredients» - указывается перечень ингредиентов, входящих в рецепт, с указанием количества и формы подготовки, а также «instructions» - текстовое описание процесса приготовления по рецепту и другие вспомогательные столбцы.

Перед началом работы с данными была выполнена предварительная обработка, направленная на улучшение качества данных и повышение точности поиска.

Основные характеристики датасета до обработки:

* Общий объём данных - 121,86 Мбайт,
* Количество строк данных – 64263 строка.

Было произведено удаление повторяющихся записей, чтобы избежать избыточности результатов, обработка от специальных символов столбца с указаниями по приготовлению, а также разбитие на полноценные предложения и стандартизация для последующего использования в семантическом поиске.

Основные характеристики датасета после обработки:

* Общий объём данных - 107,42 Мбайт,
* Количество строк данных – 56801 строка.

Для эффективной работы модели машинного обучения каждое блюдо было преобразовано в единый текстовый документ, учитывающий название, ингредиенты и описание, что позволило увеличить вес ингредиентов в общем тексте, тем самым повысив их значимость при поиске. Для этого потребовалось умножить на целые числа отельные части рецепта.

Остальные сведения про датасет:

* Среднее количество ингредиентов – 8.2 единицы, что соответствует средним по реализации и изображено на рисунке 1,

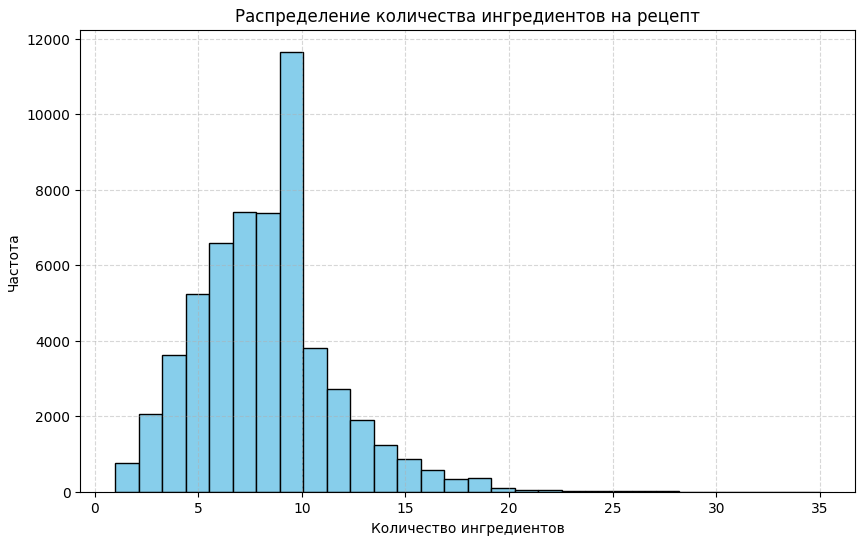


Рисунок 1 – Распределение количества ингредиентов

* Максимальное количество символов в инструкциях – 7494,
* Среднее количество символов в инструкциях – 642, что в сравнении с максимальным количеством символов соответствует, простым – средним по приготовлению рецептам, что отображено на выборке на рисунке 2.

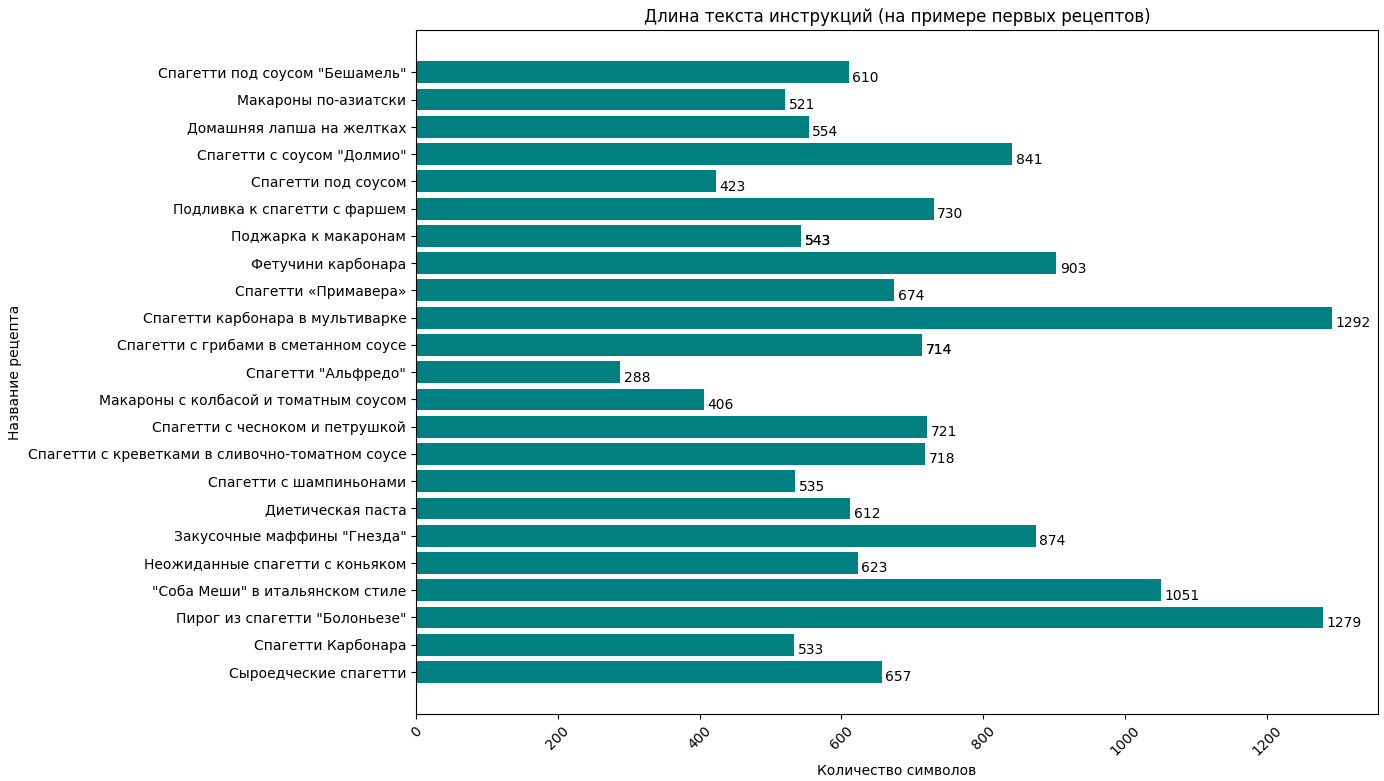


Рисунок 2 – Длина текста инструкций на выборке

Его большой объём (более 50 000 записей), разнообразие ингредиентов и подробные описания позволяют обучить модель машинного обучения и построить точную систему поиска. После предварительной обработки данные пригодны для использования в семантическом поиске с применением преобразования данных. Визуализация данных позволяет глубже понять структуру и особенности набора.

**2.2 Модель paraphrase-multilingual-MiniLM-L12-v2**

Для реализации системы семантического поиска рецептов была выбрана модель paraphrase-multilingual-MiniLM-L12-v2, которая относится к классу предобученных с использованием эмбеддингов предложений (sentence embeddings). Она является частью библиотеки sentence-transformers, разработанной для работы с текстовыми данными в задачах поиска, сравнения и рекомендаций.

У данной модели трансформерная архитектура, основанная на упрощённой версии BERT, называемой MiniLM. Она обучена на множестве параллельных корпусов на разных языках, что делает её способной понимать не только английский, но и другие языки, включая русский. Это особенно важно для проекта, поскольку данные представлены на русском языке.

Архитектура и особенности модели:

* Имеет 12 слоёв (трансформерных блоков), что обеспечивает хороший баланс между качеством и скоростью обработки текста,
* Поддержка более 50 языков, включая русский язык,
* Основана на модели MiniLM,
* Размерность эмбеддинга составляет 384,

Предназначение модели состоит в вычислении векторных представлений (эмбеддингов) предложений для дальнейшего сравнения по косинусному сходству [5]. Модель преобразует любой текст (например, рецепт) в числовой вектор фиксированной длины, который можно сравнивать с другими векторами, чтобы находить наиболее близкие по смыслу рецепты. Пример работы изображен на рисунке 3.

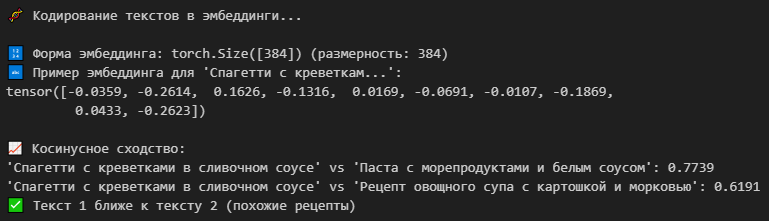


Рисунок 3 – Пример работы кодирования

При использовании модели в проекте применялись такие гиперпараметры, как «batch\_size» с размерностью 32, что означает количество рецептов, обрабатываемых за один проход. Также при создании эмбеддинга использовались «show\_progress\_bar» в положении «True» для отображения прогресса при кодировании рецептов и «convert\_to\_tensor» в положении «True» для сохранения векторов в формате PyTorch Tensor для ускорения вычисления косинусного сходства.

Так как модель является предобученной и есть возможности для дополнительного обучения то, были протестированы различные параметры, влияющие на производительность и точность. Размер количества рецептов влияет на скорость обработки данных. Увеличение значения выше 64 приводило к переполнению видеопамяти. А преобразование в тензор позволило ускорить вычисления при поиске ближайших рецептов.

Для дообучения модели необходимы пары текстов, которые описывают одинаковые или похожие рецепты. Такие рецепты были отобраны с помощью формирования пар на основе общих ингредиентов и методов приготовления. Модель обучается на основе функции потерь, которая помогает ей сравнивать два предложения и находить между ними сходство. Использовалась функция CosineSimilarityLoss, которая работает с числовыми значениями косинусного сходства между парами текстов [6]. Результаты до обучения указаны на рисунке 4 и после обучения на рисунке 5.

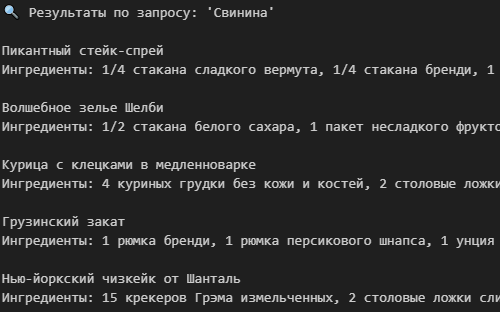


Рисунок 4 – Результат работы до обучения

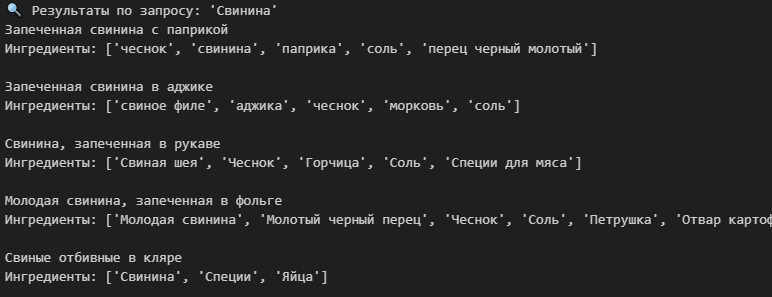


Рисунок 5 – Результат работы после обучения

Модель paraphrase-multilingual-MiniLM-L12-v2 оказалась эффективным инструментом для решения задачи поиска рецептов на основе обработки естественного языка. Её способность обрабатывать текстовые данные на русском языке и создавать числовые представления (эмбеддинги) позволила реализовать быстрый и точный механизм поиска.

Настройка параметров и предварительная обработка данных значительно повысили качество поиска. Полученные результаты демонстрируют высокую применимость данной модели к задачам рекомендательных систем и семантического поиска.

Алгоритмическое и программное конструирование

В данной главе рассматривается процесс реализации системы поиска рецептов. Основное внимание уделяется алгоритмической основе функционирования системы, включая общий алгоритм поиска, этапы применения семантической модели, а также выбор инструментальных средств для разработки. Приводится описание программной реализации, демонстрирующее, как на практике было создано приложение.

3.1 Общий алгоритм работы приложения



Рисунок 6 – Общий алгоритм использования

Общий алгоритм использования пользователем Telegram-бота «RecipeAI» представлен на рисунке 6.

3.2 Алгоритм применения обученных моделей

Для реализации системы семантического поиска рецептов была применена модель, которая позволяет преобразовывать текстовые данные (название, ингредиенты, инструкции) в числовые векторы — эмбеддинги, сравнивая между собой с помощью косинусного сходства. Все происходит в несколько этапов:

* + 1. Перед тем как использовать модель, проводится предварительная обработка текстовых данных. Для каждого рецепта формируется единый текстовый документ, взвешенно объединяющий название, ингредиенты и описание приготовления. Такой подход позволяет увеличить значимость ингредиентов в общем представлении рецепта, что особенно важно для задачи поиска по составу.
    2. После подготовки текста все рецепты кодируются в векторное пространство с помощью загруженной обученной модели. Все полученные текстовые данные преобразуются в эмбеддинги, которые позже сохраняются.
    3. Когда пользователь отправляет запрос, то его также проходит предобработка данных и происходит кодирование в эмбеддинг. В последующем происходит сравнение.

3.3 Выбор и обоснование инструментальных средств

Для реализации проекта по созданию системы семантического поиска рецептов был выбран язык программирования Python 3.12, как один из наиболее популярных и функциональных языков в области анализа данных и машинного обучения. Python предоставляет широкие возможности для работы с текстовыми данными, а также поддерживает множество библиотек, которые позволяют эффективно решать задачи обработки естественного языка и построения рекомендательных систем [7].

В качестве основной среды разработки использовались такие инструменты, как Jupyter Notebook и Visual Studio Code. Jupyter Notebook оказался удобным на начальных этапах проекта для проведения первичного анализа данных, тестирования функций и проверки корректности обработки текста. По мере усложнения логики и перехода к финальной реализации основная часть кода была перенесена в файлы Python, что позволило улучшить структурированность проекта и обеспечить его повторное использование. Для этих задач была выбрана среда разработки Visual Studio Code — мощный и гибкий редактор, поддерживающий работу с Jupyter Notebook, файлами датасета в формате .CSV и файлами Python одновременно.

Работа с датасетом осуществлялась с помощью библиотеки pandas, которая обеспечила простой и эффективный доступ к данным, их очистку, преобразование и анализ. Для выполнения операций над текстами применялась стандартная библиотека Python, включая модуль re для работы с регулярными выражениями, что позволило качественно подготовить данные перед дальнейшей обработкой моделью.

Все вычисления производились с использованием библиотеки PyTorch. Также применялись стандартные средства сериализации, такие как pickle, для сохранения и загрузки рассчитанных эмбеддингов, чтобы избежать повторного кодирования всех рецептов при каждом запуске программы.

Для создания интерфейса взаимодействия с пользователем была реализована система в виде Telegram-бота с использованием библиотеки python-telegram-bot. Такой выбор обусловлен её простотой использования, хорошей документацией и возможностью быстрого развёртывания прототипа. Благодаря этой библиотеке удалось организовать диалоговое взаимодействие с пользователем, реализовать навигацию по рецептам и красиво отображать результаты поиска.

3.4 Программная реализация

Основная логика обработки данных, обучения модели и поиска реализована в модуле search\_alg.py. Этот файл содержит все необходимые функции для подготовки данных, работы с моделью и выполнения поиска по запросам пользователя.

В начале программы происходит загрузка датасета и предварительная обработка текста каждого рецепта с помощью функции preprocess\_recipe(row). Эта функция формирует единое текстовое представление рецепта, усиливая значимость отдельных частей, таких как название, ингредиенты и описание. Такой подход позволяет улучшить качество поиска за счёт увеличения веса ключевых слов.

После подготовки текста всех рецептов система кодирует их в числовое пространство с помощью модели и её функции encode(row, \*\*kwargs), способной создавать эмбеддинги — числовые векторы, отражающие семантическое значение текста. Эти эмбеддинги сохраняются в файл recipe\_embeddings.pkl, чтобы не пересчитывать их при каждом запуске программы, что значительно ускоряет работу системы.

Функция find\_recipes(query, top\_n = 5) является центральной в процессе поиска. Она принимает текстовый запрос пользователя и возвращает список наиболее релевантных рецептов. Для этого сначала применяется функция filter\_by\_keywords(df, query), которая выполняет предварительную фильтрацию рецептов по наличию ключевых слов в тексте. Эта функция старается найти рецепты, содержащие все слова из запроса, а если таковых недостаточно — переходит к более мягкому условию, где достаточно совпадения хотя бы одного слова. Это помогает обеспечить баланс между точностью и полнотой выборки.

После фильтрации вычисляется косинусное сходство между эмбеддингом пользовательского запроса и эмбеддингами найденных рецептов. Результаты ранжируются по убыванию сходства, после чего возвращаются топ-N самых релевантных рецептов. Для повышения читаемости результатов текстовые инструкции разбиваются на шаги, а лишние символы и повторяющиеся номера удаляются. Благодаря этому каждый рецепт выводится в аккуратном и понятном виде.

Демонстрация работы приложения

Произведём тестирование разработанного кулинарного бота RecipeAI - интеллектуального помощника для поиска рецептов. На рисунке 7 представлен Telegram-бот.

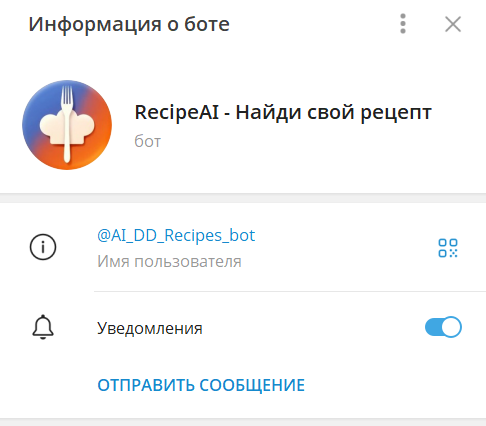
****

Рисунок 7 - бот RecipeAI

Бот эффективно взаимодействует с пользователем. При первом запуске он приветствует и предлагает ввести запрос (например: "Пирог с вишней" или "Спагетти с креветками"). Рассмотрим запуск на рисунке 8.

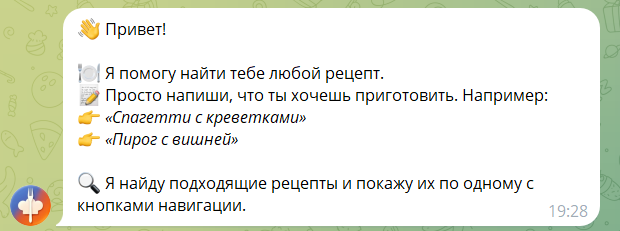
****

Рисунок 8 - бот RecipeAI при запуске

На примере запроса “пирог с вишней” протестируем работу бота на рисунках 9-13.

На рисунке 9 представлен подробный рецепт летнего пирога с вишней.

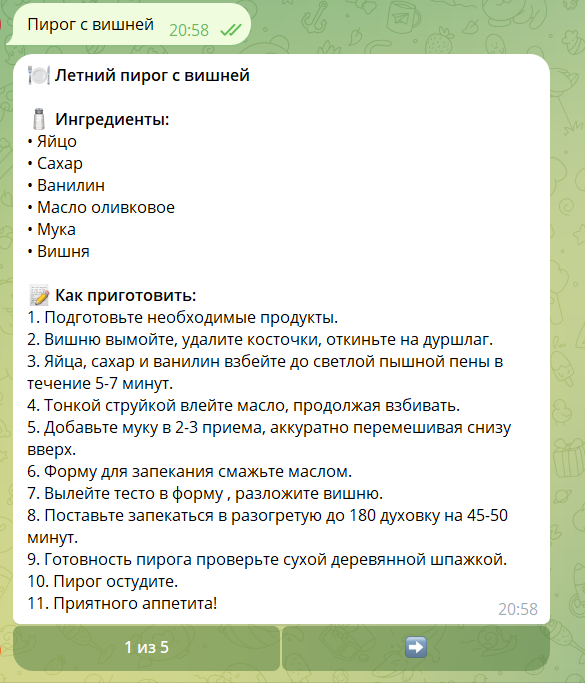
****

Рисунок 9 – летний пирог с вишней

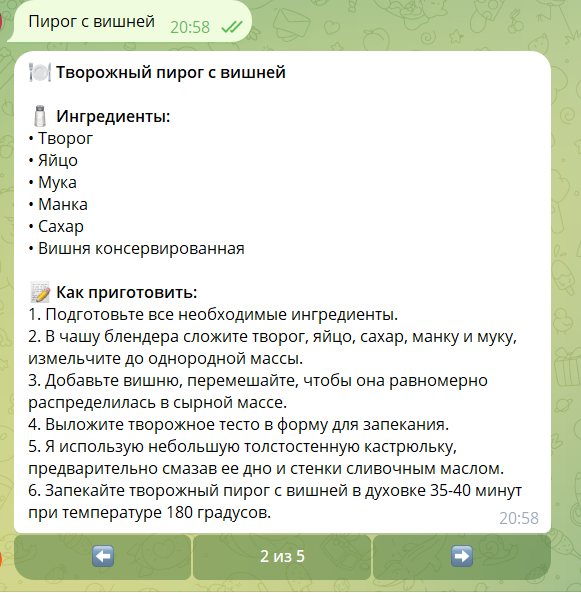
****

Рисунок 10 – творожный пирог с вишней

На рисунке 10 представлен подробный рецепт творожного пирога с вишней. На рисунке 11 представлен подробный рецепт пирога с вишней из песочного теста.

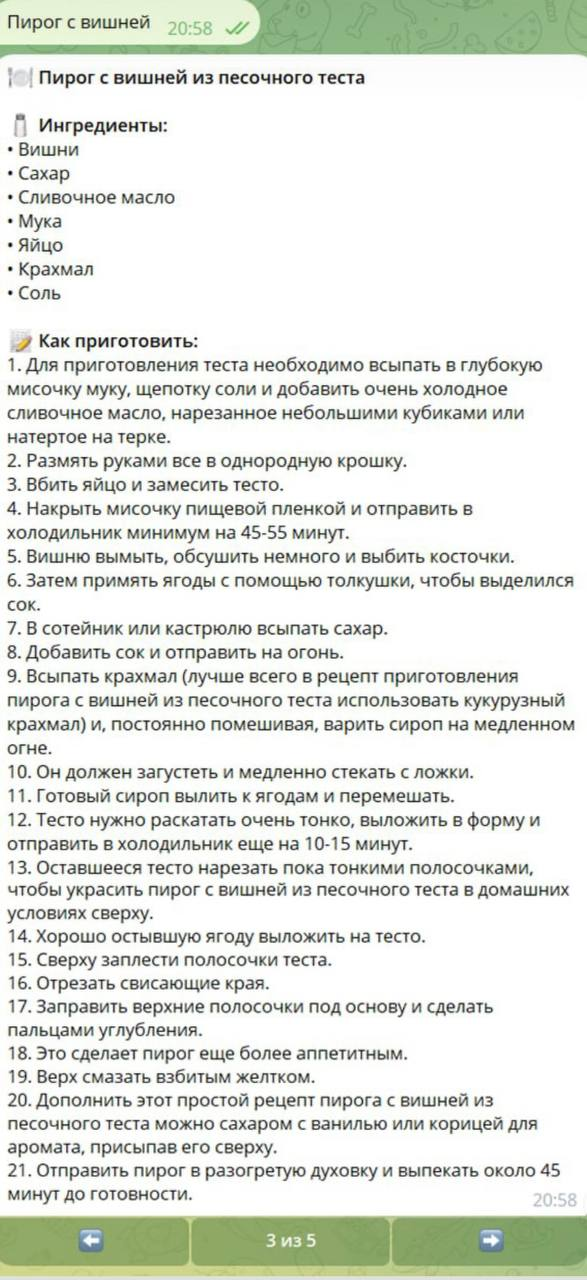


Рисунок 11 – пирог с вишней из песочного теста

На рисунке 12 представлен подробный рецепт венского пирога с вишней.

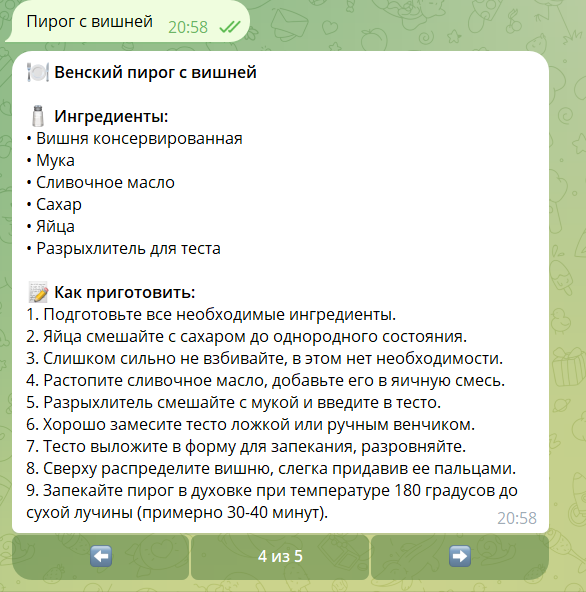
****

Рисунок 12 – венский пирог с вишней

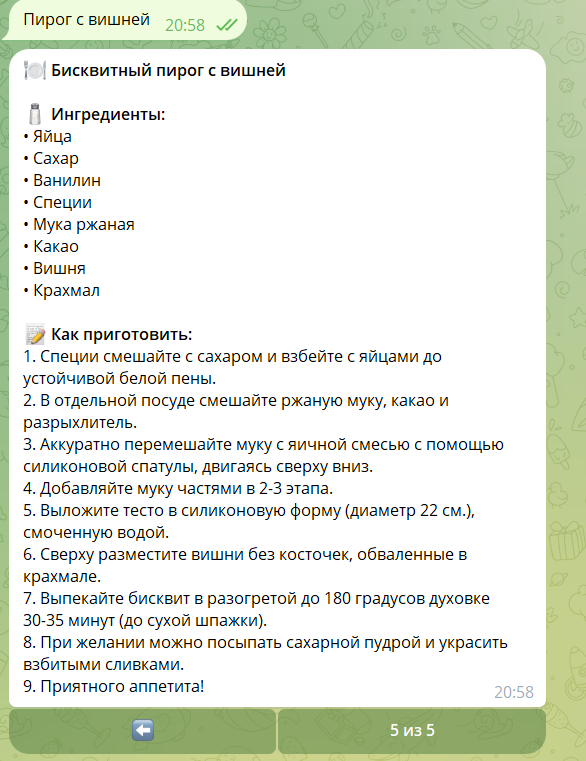
****

Рисунок 13 – бисквитный пирог с вишней

На рисунке 13 представлен подробный рецепт бисквитного пирога с вишней.

В ходе тестирования была продемонстрирована эффективность работы алгоритмов машинного обучения, обрабатывающих пользовательские запросы и осуществляющих поиск по обширной базе данных рецептов, состоящей из более 50000 рецептов.

При обработке запроса "Пирог с вишней" система успешно идентифицировала и извлекла из датасета несколько вариантов рецептов, включая летний, творожный, песочный, венский и бисквитный варианты приготовления. Анализ результатов показал, что алгоритм корректно обрабатывает как точные совпадения в названиях блюд, так и семантически близкие формулировки.

Каждый найденный рецепт представлен в стандартизированном формате, включающем:

1. структурированный перечень ингредиентов с указанием их количества
2. детализированную пошаговую инструкцию приготовления
3. параметры температурного режима и времени приготовления
4. дополнительные кулинарные рекомендации

Особенностью реализации является использование NLP-алгоритмов для обработки естественноязыковых запросов и векторного поиска по предварительно сгенерированным эмбеддингам рецептов. Это позволяет системе эффективно работать даже с неполными или неточно сформулированными запросами.

Пользовательский интерфейс реализован с учетом эргономических требований к мобильным приложениям, обеспечивая интуитивно понятное взаимодействие через чат-бот. Навигация между найденными рецептами осуществляется посредством кнопочного интерфейса, что значительно упрощает процесс ознакомления с альтернативными вариантами приготовления.

Проведенные тесты подтвердили стабильность работы системы при обработке различных типов запросов и корректность ранжирования результатов поиска. Полученные результаты демонстрируют эффективность примененных алгоритмов машинного обучения и соответствие системы заявленным функциональным требованиям.

Заключение

В результате выполненной работы была обучена модель и разработан Telegram-бот, который способен анализировать множество параметров, связанных с кулинарными предпочтениями пользователей, имеющимися ингредиентами и диетическими ограничениями. На основе собранных данных бот обучается и в дальнейшем предлагает персонализированные рецепты в ответ на запросы пользователей. Благодаря этому владельцы кафе и ресторанов теперь смогут быстро находить идеи для меню, а обычные пользователи — легко выбирать блюда для приготовления.

Для взаимодействия с ботом было разработано удобное приложение, выполняющее следующие функции: анализ имеющихся продуктов и предпочтений пользователя, предложение рецептов, адаптированных под заданные параметры, учет диетических ограничений и вкусовых предпочтений.

В процессе работы были получены и закреплены навыки выбора и реализации алгоритмов машинного обучения, подбора гиперпараметров, а также организации процесса обучения и применения обученной модели. Проект демонстрирует значительный потенциал для расширения функциональности, включая интеграцию с "умной" кухонной техникой, что сделает приготовление пищи еще более простым и эффективным.

**Перечень использованных информационных ресурсов**

* + - 1. Markets and Markets. (2020). Cooking App Market by Component (Solutions and Services), Deployment Model, Application (Recipe & Cooking, Meal Planning), Platform, User Type, Region - Global Forecast to 2025. URL:https://www.marketsandmarkets.com/Market-Reports/cooking-app-market-145738577.html) (дата обращения: 16.05.2025).
      2. The Home Chef: Cooking Trends Research. URL: (https://www.nielsen.com/us/en/insights/article/2021/the-home-chef-cooking-trends-research/) (дата обращения: 16.05.2025).
      3. Miller, A. (2021). Chatbots: 2021 and Beyond. The Guide to Bots in Messaging Apps. - New York: Tech Press, 2021. - 159 с.
      4. Ильюшин, А. В. Машинное обучение: традиции и современные подходы. - М.: Издательство "Бином", 2020. – 204 c.
      5. Алмеров, Ф. Глубокое обучение для семантического распознавания. - М.: Научно-техническая литература, 2019. – 318 c.
      6. Алан, Дж. П. Практическое руководство по обработке и анализу данных с использованием Python. - М.: Питер, 2021. – 236 c.
      7. Румельт, К. Обработка естественного языка с помощью Python и библиотек - М.: БХВ-Петербург, 2021. - 125 с.

Приложение Листинг программы

import pandas as pd

import ast

from sentence\_transformers import SentenceTransformer

from sentence\_transformers import util

import torch

import pickle

# --- 1. Предобработка текста ---

def preprocess\_recipe(row):

name = row['name'] if isinstance(row['name'], str) else ''

ingredients = ' '.join(ast.literal\_eval(row['nor\_ingridients'])) if isinstance(row['nor\_ingridients'], str) else ''

instructions = row['instructions'].lower() if isinstance(row['instructions'], str) else ''

enhanced\_text = (

(name + ' ') \* 2 +

(ingredients + ' ') \* 3 +

(instructions + ' ') \* 2

).strip().lower()

return enhanced\_text

# --- 2. Загрузка датасета ---

print("📁 Загрузка датасета...")

df = pd.read\_csv("List\_of\_Recipes.csv")

# --- 3. Обработка текста ---

print("🧹 Обработка рецептов...")

df['clean\_text'] = df.apply(preprocess\_recipe, axis=1)

# --- 4. Загрузка модели ---

print("🧠 Загрузка модели...")

model = SentenceTransformer('sentence-transformers/paraphrase-multilingual-MiniLM-L12-v2')

# # --- 5. Получение эмбеддингов ---

# print("🧬 Кодирование рецептов...")

# recipe\_texts = df['clean\_text'].tolist()

# recipe\_embeddings = model.encode(recipe\_texts, show\_progress\_bar=True)

# with open("recipe\_embeddings.pkl", "wb") as f:

# pickle.dump(recipe\_embeddings, f)

# --- 5. Загрузка эмбеддингов ---

print("🧬 Загрузка эмбеддингов...")

with open("recipe\_embeddings.pkl", "rb") as f:

recipe\_embeddings = pickle.load(f)

# --- 6. Функция поиска ---

def find\_recipes(query, top\_n=5):

filtered\_df = filter\_by\_keywords(df, query)

filtered\_indices = filtered\_df.index.tolist()

filtered\_embeddings = recipe\_embeddings[filtered\_indices]

query\_embedding = model.encode([query.lower()])

cos\_scores = util.cos\_sim(query\_embedding, filtered\_embeddings)[0]

top\_k = min(top\_n \* 3, len(filtered\_indices))

top\_indices\_in\_subset = torch.topk(cos\_scores, k=top\_k).indices.tolist()

result\_indices = [filtered\_indices[i] for i in top\_indices\_in\_subset]

results = []

for idx in result\_indices:

results.append({

'name': df.iloc[idx]['name'],

'ingredients': df.iloc[idx]['pure\_ingridients'],

'instructions': df.iloc[idx]['instructions'],

'score': cos\_scores[result\_indices.index(idx)].item()

})

if len(results) >= top\_n:

break

return results

# --- 7. Функция предфильтрации ---

def filter\_by\_keywords(df, query, column='clean\_text'):

stopwords = {'в', 'на', 'с', 'по', 'для', 'из', 'от', 'до', 'и', 'или', 'не'}

query\_words = [word.strip() for word in query.lower().split(',') if word.strip() not in stopwords]

filtered = df[df[column].apply(lambda x: all(word in x for word in query\_words))]

if filtered.empty:

filtered = df[df[column].apply(lambda x: any(word in x for word in query\_words))]

return filtered if not filtered.empty else df