

ОТЧЕТ ПО ПРОЕКТУ

Подготовка набора данных и настройки нейросетевой модели для обработки текста

Студент: Ринат

Дата: Ноябрь 2024

1. Название проекта

RAG-система для семантического поиска по базе инструкций технической поддержки

2. Идея построения модели нейронной сети и решаемые задачи

Идея проекта:

Разработка локальной системы на основе архитектуры RAG (Retrieval-Augmented Generation) для автоматизации работы технической поддержки. Система использует нейронные сети для понимания смысла запросов пользователей и поиска релевантной информации в корпоративной базе знаний, после чего генерирует развернутые ответы на естественном языке.

Задачи, которые решает нейронная сеть:

- Семантический анализ текстовых запросов пользователей и преобразование их в векторные представления (эмбединги)
- Поиск наиболее релевантных фрагментов документов по семантическому сходству (не по точному совпадению слов)
- Обработка и понимание текстовой информации из документов различных форматов (.docx, .md, .txt)
- Генерация естественных языковых ответов на основе найденного контекста с сохранением фактической точности
- Ранжирование результатов поиска по степени релевантности к запросу пользователя

Используемые нейросетевые модели:

- 1) Модель эмбедингов: intfloat/multilingual-e5-small (Transformer-based encoder, ~118M параметров). Преобразует текст в векторы размерности 384, обеспечивая семантическое понимание.
- 2) Языковая модель: Llama3:8b (Decoder-only Transformer, ~8B параметров). Генерирует естественные языковые ответы на основе найденного контекста.

3. Подготовка первоначального набора данных

3.1. Краткое описание данных:

Набор данных представляет собой корпоративную базу знаний технической поддержки, содержащую инструкции по работе с системой 1С и кассовым оборудованием (ККМ). Документы включают пошаговые руководства, решения типовых проблем, описания ошибок и методы их устранения.

Типы документов:

- Инструкции по загрузке данных (справочники, закупы)
- Решения проблем с кассовым оборудованием
- Инструкции по работе с системой 1С
- Описания ошибок и способы их устранения
- Процедуры технического обслуживания

3.2. Общий размер набора данных:

Исходные документы: ~40 КБ (8 текстовых файлов в форматах .md, .docx, .txt)

После обработки и индексации: векторная база данных ChromaDB размером ~2 МБ

Количество индексированных текстовых фрагментов (чанков): ~50-100 (зависит от размера документов)

3.3. Источник данных:

Формировался самостоятельно. Данные собраны из внутренней корпоративной документации компании и представляют собой реальные инструкции, используемые сотрудниками технической поддержки в повседневной работе. Все документы содержат конфиденциальную бизнес-информацию и не публикуются в открытом доступе.

4. Формирование обучающей выборки

4.1. Данные для обучения модели:

В данном проекте применяется подход Transfer Learning (переноса обучения). Модели не обучаются с нуля на корпоративных данных, а используются предобученные модели:

- Модель эмбедингов (intfloat/multilingual-e5-small) предобучена на большом многоязычном корпусе с использованием контрастивного обучения для задач информационного поиска (retrieval).
- Языковая модель (Llama3:8b) предобучена на триллионах токенов текста и дополнительно дообучена на инструкциях (instruction tuning) для следования указаниям пользователя.

Обоснование: Обучение больших языковых моделей с нуля требует огромных вычислительных ресурсов (тысячи GPU, недели обучения). Использование предобученных моделей позволяет получить высокое качество работы при минимальных затратах ресурсов.

4.2. Данные для тестирования модели:

Для проверки качества работы системы подготовлен тестовый набор из реальных запросов сотрудников техподдержки:

- Загрузка справочников вручную - ожидаемый документ: ЗагрузкаСправочниковВручную.docx
- Переустановка гостей - ожидаемый документ: Сертификаты, гости.md
- Загрузка закупов на магазин вручную - ожидаемый документ: загрузка закупов на магазин вручную.docx
- Ошибка ккм 409 - ожидаемый документ: Мануал 1С работа.docx

Метрики оценки: Точность поиска (нашелся ли правильный документ в TOP-5), релевантность ответа (содержит ли ответ нужную информацию), полнота ответа (не потеряна ли важная информация).

Результаты тестирования: Система показывает точность ~60-70% на текущем наборе данных. Основные проблемы связаны с синонимами технических терминов и потерей контекста при разбиении на чанки.

4.3. Предобработка и подготовка данных:

Процесс подготовки данных для индексации включает несколько этапов:

Этап 1: Парсинг документов

- Извлечение текста из файлов .docx с помощью библиотеки python-docx
- Чтение Markdown (.md) и текстовых (.txt) файлов
- Извлечение встроенных изображений (сохранение для будущего использования)
- Нормализация текста: удаление лишних пробелов, специальных символов

Этап 2: Разбиение на чанки (chunking)

Документы разбиваются на перекрывающиеся фрагменты для оптимального поиска:

- Размер чанка: 500 токенов (~2000 символов)
- Перекрытие: 50 токенов (10% от размера чанка)
- Обоснование: Размер 500 токенов обеспечивает баланс между сохранением контекста (чанк не слишком маленький) и точностью поиска (чанк не слишком большой). Перекрытие гарантирует, что важная информация не будет разделена на границе чанков.

Этап 3: Добавление контекста

К каждому чанку добавляется заголовок исходного документа для сохранения контекста:

- Формат: 'Документ: [Название файла]\n\n[Текст чанка]'
- Это помогает модели понимать, к какому документу относится чанк

Этап 4: Генерация эмбеддингов

Каждый чанк преобразуется в векторное представление:

- Модель: intfloat/multilingual-e5-small
- Размерность вектора: 384
- Метрика расстояния: косинусное сходство

Этап 5: Индексация в векторной базе данных

Чанки сохраняются в ChromaDB вместе с метаданными:

- doc_id - уникальный идентификатор документа
- filename - название исходного файла
- chunk_index - номер чанка в документе
- total_chunks - общее количество чанков в документе

- active - флаг активности (для пометки устаревших документов)
- author - автор документа
- tags - теги для категоризации
- created_at - дата добавления

Этап 6: Оптимизация индекса

ChromaDB автоматически создает индекс HNSW (Hierarchical Navigable Small World) для быстрого приближенного поиска ближайших соседей (ANN - Approximate Nearest Neighbors).

5. Архитектура системы и процесс работы

5.1. Общая архитектура RAG-системы:

- Frontend: Веб-интерфейс на Streamlit (локальный веб-сервер)
- Backend: Python-приложение с модульной архитектурой
- Векторная БД: ChromaDB (локальное хранилище с персистентностью)
- Модель эмбедингов: Sentence Transformers (работает на CPU)
- LLM: Llama3:8b через Ollama (локальная генерация)
- Хранилище документов: Локальная файловая система

5.2. Процесс обработки запроса (inference pipeline):

1. Пользователь вводит вопрос в веб-интерфейс
2. Запрос преобразуется в эмбединг с помощью модели `intfloat/multilingual-e5-small`
3. Выполняется векторный поиск в ChromaDB для нахождения TOP-K (3-5) наиболее похожих чанков
4. Найденные чанки объединяются в контекст с указанием источников
5. Формируется промпт для LLM: системная инструкция + контекст + вопрос пользователя
6. Llama3:8b генерирует ответ на основе предоставленного контекста
7. Ответ отображается пользователю вместе со ссылками на исходные документы

6. Заключение

В рамках проекта разработана полнофункциональная RAG-система для семантического поиска по корпоративной базе знаний. Система использует современные архитектуры нейронных сетей (Transformers) и эффективно решает задачу автоматизации технической поддержки.

Основные достижения:

- Реализован полный пайплайн RAG: от загрузки документов до генерации ответов
- Система работает полностью локально, обеспечивая конфиденциальность данных
- Используется transfer learning, избегая дорогостоящего обучения с нуля
- Создан удобный веб-интерфейс для взаимодействия с системой
- Реализована модульная архитектура, позволяющая легко расширять функционал

Технические характеристики:

- Модель эмбедингов: intfloat/multilingual-e5-small (~118M параметров)
- Языковая модель: Llama3:8b (~8B параметров)
- Векторная БД: ChromaDB с индексом HNSW
- Размер чанка: 500 токенов с перекрытием 50 токенов
- Точность на тестовых запросах: ~60-70%

Система готова к практическому использованию и дальнейшему улучшению. Определены направления развития: улучшение парсинга, обработка изображений, создание словаря терминов, добавление категоризации документов.