

Министерство науки и высшего образования Российской Федерации

Федеральное государственное автономное образовательное учреждение высшего образования «Уральский федеральный университет имени первого Президента России Б. Н. Ельцина» (УрФУ)

Институт радиоэлектроники и информационных технологий – РТФ

ОТЧЕТ

о проектной работе

по теме: Нормализация данных при помощи LLM

по дисциплине: Проектный практикум

Студент: Королёв Семён РИ-231003

Студент: Верховых Савелий РИ-231003

Студент: Нажимов Игорь РИ-231001

Екатеринбург

2025

**СОДЕРЖАНИЕ**

[ВВЕДЕНИЕ 3](#_Toc165369900)

[1. Целевая аудитория 4](#_Toc165369901)

[2. Определение проблемы 6](#_Toc165369902)

[3. Подходы к решению проблемы 9](#_Toc165369903)

[4. Анализ аналогов 11](#_Toc165369904)

[5. Календарный план проекта 12](#_Toc165369905)

[6. Сценарии использования 14](#_Toc165369906)

[7. Требования к продукту и к MVP 15](#_Toc165369907)

[8. Стек для разработки 17](#_Toc165369908)

[9. Прототипирование 19](#_Toc165369909)

[10. Проектирование и разработка системы 21](#_Toc165369910)

[ЗАКЛЮЧЕНИЕ 22](#_Toc165369911)

[СПИСОК ЛИТЕРАТУРЫ 23](#_Toc165369912)

[ПРИЛОЖЕНИЕ 26](#_Toc165369913)

ВВЕДЕНИЕ

С развитием технологий появляется всё больше требований к качеству и скорости получаемой информации.

Ежегодно объёмы данных растут с немыслимой скоростью, а компании уделяют много внимания автоматизации процессов и повышению точности и эффективности обработки данных.

Чтобы частично решить данную проблему наша команда создала программу, которая занимается нормализацией данных при помощи LLM.

**Цель нашего проекта:** разработать программный продукт, занимающийся нормализацией необработанных текстовых данных, в виде товарных записей, при помощи большой языковой модели (LLM).

Под нормализацией понимается разбиение записей на компоненты, относящихся к определенному атрибуту (характерному свойству объекта).

**Задачи нашего проекта:**

* Определить способ взаимодействия с LLM.
* Определить модель LLM.
* Выбрать способ обработки нормализованных данных.
* Определить метрики.
* Выбрать способ вывода обработанных, нормализованных данных.

1. Целевая аудитория

Для определения целевой аудитории мы использовали методику 5W Марка Шеррингтона. Это наиболее распространенный способ определения целевой аудитории и психологических характеристик, которыми обладают потенциальные потребители.

Сегментация рынка проводилась по 5 вопросам:

1. Что? – мы предлагаем пользователям программу, которая поможет нормализировать необработанные текстовые данные, в виде товарных записей, при помощи LLM.
2. Кто? – нашей целевой аудиторией являются люди\компании, которым необходимо быстро и просто преобразовать входные данные в определённый формат.
3. Почему? – мотивация клиента на использование нашего продукта заключается в желании клиента получить результат, быстро, точно и бесплатно.
4. Когда? – поскольку это не коммерческий проект, то потребители могут получить данную программу бесплатно в любое удобное для них время.
5. Где? – наш продукт находится в открытом доступе на сайте GitHub.

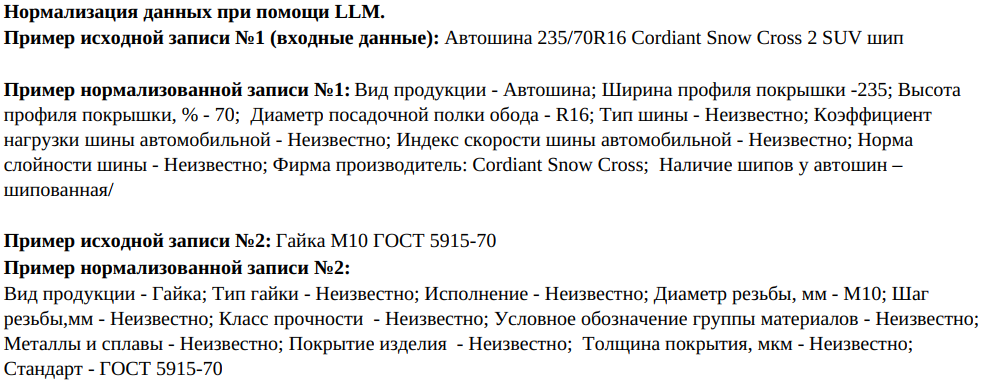
Поскольку наш продукт не нацелен на получение материальной выгоды, то основными вопросами, которыми мы пользовались для определения нашей целевой аудитории были «Что?», «Кто?», «Почему?».

В результате анализа мы убедились, что наша целевая аудитория — это лица (как физические, так и юридические), которым требуется быстрая, точная и бесплатная нормализация данных.

1. Определение проблемы

Для определения проблемы нашей команде были предоставлены примеры исходных и нормализованных данных:

Рисунок 1 – Примеры исходных и нормализованных данных



Определив формат входных и выходных данных, мы столкнулись со следующими вопросами:

* Каким методом взаимодействия с LLM мы будем пользоваться?
* Как выбрать подходящую LLM модель?
* Как мы будем обрабатывать нормализованные данные?
* Как мы собираемся выводить обработанные, нормализованные данные?

Определив ключевые проблемы, мы можем переходить к формулированию подходов к их решению.

1. Подходы к решению проблемы

Чтобы добиться поставленной цели мы рассмотрели каждый вопрос по отдельности:

* **Каким методом взаимодействия с LLM мы будем пользоваться?**

Для решения данной проблемы мы изучили различные интернет-источники и перед нами встал выбор:

1. **Fine-Tuning** — это процесс дообучения заранее обученной модели на специализированных наборах данных для конкретных задач.
2. **Prompt Engineering** — это процесс создания и оптимизации запросов для языковых моделей.
3. **Prompt Tuning** — это метод, при котором к входу языковой модели добавляют небольшой набор обучаемых параметров (так называемых «мягких промптов»).

Рассмотрев каждый вариант, мы остановились на Prompt Engineering из-за простоты его реализации, т.к. для других вариантов требовалось больше времени для изучения и более мощное железо, что являлось нашими ограничениями при выполнении данного проекта.

Следующим шагом мы решали следующую проблему:

* **Как выбрать подходящую LLM модель?**

В качестве вариантов мы рассматривали такие модели как:

1. **Qwen** — китайская нейросеть с открытым исходным кодом от компании Alibaba.
2. **Deepseek** — языковая нейросеть в виде чат-бота. Работает на базе технологии LLM, понимает и генерирует естественный язык.
3. **LLaMA** — серия мультимодальных моделей искусственного интеллекта с открытым исходным кодом от компании Meta.

Главным критерием мы выбрали точность оценки. Протестировав каждую из них, мы решили остановиться на **Qwen** из учета скорости и точности.

Далее мы сформировали prompt и столкнулись со следующий проблемой:

* **Как мы будем обрабатывать нормализованные данные?**

В качестве решения мы сформулировали поэтапную инструкцию:

1. На вход подаются файлы xlsx формата, которые содержат несколько запросов.
2. После получения ответа от LLM мы форматируем ответ, который получаем в виде строки, так чтобы на выходе можно было получить таблицу в формате xlsx.

Получив обработанные данные нам оставалось решить последнюю проблему.

* **Как мы собираемся выводить обработанные, нормализованные данные?**

Изучив различные интернет-источники, мы остановились на 2-х вариантах:

1. **Gradio** —это открытая библиотека на языке Python, созданная для упрощения разработки веб-приложений.
2. **tkinter** — это кроссплатформенная событийно-ориентированная графическая библиотека на основе средств Tk для языка программирования Python.

Мы остановились на tkinter из-за его более простого изучения и наличия русской документации.

1. Анализ аналогов

Прямые конкуренты – это аналогичное программное обеспечение на аналогичном рынке и работающий с целевой аудиторией, что и разрабатываемый продукт.

Косвенные конкуренты – это похожее программное обеспечение с другими характеристиками или абсолютно другой продукт, но работающие с целевой аудиторией, что и разрабатываемый продукт.

Поскольку мы не стремимся занять определённую нишу в данной сфере, а наш продукт является абсолютно бесплатным и общедоступным, мы решили отказаться от анализа аналогов.

1. Календарный план проекта

Название проекта:

Руководитель проекта:

Таблица 1 – Календарный план

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **№** | **Название** | **Ответственный** | **Длительность** | **Дата начала** | **Временные рамки проекта** | | | |
| 1 нед | 2 нед | 3 нед | 4 нед |
| ***Анализ возможных алгоритмов, поиск решений для прототипа. Анализ и подготовка данных.*** | | | | | | | | |
| *1.1* | *Выбор решений для прототипа* | Верховых Савелий | Неделя | 21.03.2025 | + |  |  |  |
| *1.2* | *Подготовка данных* | Королёв Семён | Неделя | 21.03.2025 | + |  |  |  |
| *1.3* | *Выбор между самописной нейронной сетью и готовыми LLM* | Верховых Савелий | Неделя | 21.03.2025 | + |  |  |  |
| *1.4* | *Изучение доступных LLM (выбор подходящей)* | Нажимов Игорь | Неделя | 21.03.2025 | + |  |  |  |
| ***Разработка кодовой базы. Адаптация алгоритмов. Поиск метрик качества работы выбранных решений. Оценка качества работы.*** | | | | | | | | |
| *2.1* | *Подготовить скрипты для запросов, отправляемых в AI* | Верховых Савелий | 3 недели | 03.04.2025 |  | + |  |  |
| *2.2* | *Провести ретроспективу по пройденной итерации* | Королёв Семён | 3 недели | 03.04.2025 |  | + |  |  |
| *2.3* | *Изучить способы подключения данных скриптов для взаимодействия с AI* | Верховых Савелий | 3 недели | 03.04.2025 |  | + |  |  |
| ***Оптимизация алгоритмов. Улучшение работы и повышение уровня метрик. Доработка и оптимизация сборки.*** | | | | | | | | |
| *3.1* | *Разобраться с Gradio (или иными средствами\библиотеками создания GUI)* | Королёв Семён | 3 недели | 25.04.2025 |  |  | + |  |
| *3.2* | *Разобраться с работой в git* | Нажимов Игорь | 3 недели | 25.04.2025 |  |  | + |  |
| *3.3* | *Подготовить код для работы с файлом* | Нажимов Игорь | 3 недели | 25.04.2025 |  |  | + |  |
| *3.4* | *Доработать LLM запросы (скрипт) и взаимодействие с AI* | Верховых Савелий | 3 недели | 25.04.2025 |  |  | + |  |
| ***Представление разработанного решения. Презентация и показ заказчику.*** | | | | | | | | |
| *4.1* | *Оптимизировать работу нейронки* | Верховых Савелий | 2 недели | 15.05.2025 |  |  |  | + |
| *4.2* | *Подготовить отчёт* | Королёв Семён | 2 недели | 15.05.2025 |  |  |  | + |
| *4.3* | *Подготовить презентацию* | Королёв Семён | 2 недели | 15.05.2025 |  |  |  | + |
| *4.4* | *Завершить разработку приложения (подготовить метрики)* | Нажимов Игорь | 2 недели | 15.05.2025 |  |  |  | + |
|  | *Защита проекта* |  | Месяц | 30.05 - 26.06 |  |  |  | + |

1. Сценарии использования

Сценарий использования (use case) — описание поведения системы, когда она взаимодействует с кем-то (или чем-то) из внешней среды.

1. **Запускается исполняемый exe файл (перед вами откроется GUI)**
2. **Добавляете необходимые для обработки файлы в соответствующее окно (поддерживаются только формат xlsx)**

**Важно:** если был подан файл с иным расширением программа не перейдёт к 3 пункту, а выдаст ошибку и попросит загрузить файл с верным расширением.

1. **Ожидайте окончания обработки данных**

* В этот момент программа отправляет prompt в Qwen
* Получив ответ, форматирует данный ответ в xlsx (в зависимости от расширения исходного файла)
* Загружает файл в GUI

1. **Скачивается готовый файл**
2. **Закрывается GUI или нажимается кнопку «Попробовать снова»**

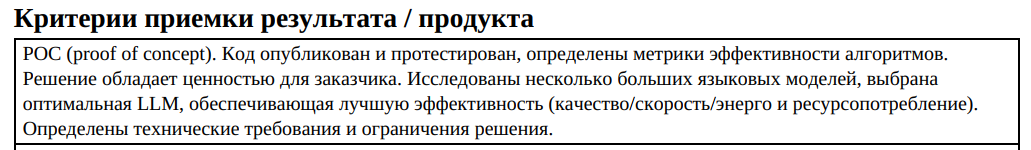
**Важно:** если была нажата кнопка **«Попробовать снова»**, программа возвращается к исходному состоянию, как при запуске.

1. Требования к продукту и к MVP

Функциональные требования

В качестве функциональных требований используется сценарий использования, если один из пунктов сценария использования не выполняется, то MVP не достигнуто.

Требования заказчика



1. Стек для разработки

Стек для разработки — это набор инструментов и технологий, которые используются для создания продуктов и приложений.

В качестве стека разработки нами были взяты:

* Python
* Библиотеки Python: transformers, tkinter, os, shutil, subprocess, threading, uuid, pathlib, sys
* Платформа GitHub
* Qwen

1. Прототипирование

Для дальнейшей разработки и теста выбранного стека разработки нами создавались следующие прототипы:

* Прототип скрипта (prompt)
* Прототип GUI
* Прототип

Все прототипы можно посмотреть на нашем GitHub, ссылка на него находится во вкладке «**ПРИЛОЖЕНИЕ**».

1. Проектирование и разработка системы

|  |  |
| --- | --- |
| Qwen | Использует модель Qwen для генерации ответов (извлечения характеристик) |
| get\_reqs() | Чтение входных данных из /XLSX файла |
| create\_excel() | Сохранение результатов в формате Excel |
| FileProcessorApp | Графический интерфейс (GUI), управление файлами и процессом обработки |
| main() | Точка входа, запуск GUI |

1. Модуль QwenChatbot

* Загрузка модели и токенизатора.
* Установка системного промпта с правилами извлечения.
* Формирование запроса от пользователя (описание товара).
* Генерация ответа с помощью модели.
* Добавление ответа в историю диалога.

2. Функция get\_reqs()

* Открытие файла.
* Чтение по строкам.
* Объединение значений в одну строку через пробел.

3. Функция create\_excel()

* Парсинг JSON-данных в DataFrame.
* Сохранение DataFrame в Excel.
* (Опционально) Вычисление метрики ROUGE для оценки качества извлечения.

4. Класс FileProcessorApp

* Выбор файла (/XLSX)
* Обработка в отдельном потоке
* Показ прогресса и ошибок
* Возможность скачать результат или повторить операцию

Особенности реализации сложных блоков

1. Интеграция LLM

Требования:

* Поддержка CUDA для ускорения работы.
* Работа с длинными контекстами (до 8192 токенов).
* Эффективное управление памятью GPU.

Сложности:

* Долгое время генерации на CPU.
* Ограничения по объему видеопамяти.
* Возможные ошибки при некорректном формате ответа.

Решения:

* Использование float16 для экономии памяти.
* Очистка GPU после каждой генерации.
* Обработка исключений и контроль времени выполнения.

ЗАКЛЮЧЕНИЕ

В рамках реализации данного проекта мы успешно достигли поставленной цели — разработали программный продукт, осуществляющий нормализацию необработанных текстовых данных в виде товарных записей с использованием большой языковой модели (LLM). Программа позволяет автоматически извлекать ключевые технические характеристики из описания товаров и представлять их в структурированном виде, что значительно упрощает дальнейшую обработку и анализ информации.

**Нами были выполнены все задачи, поставленные на начальном этапе:**

* Определён подход к взаимодействию с LLM. Мы выбрали Prompt Engineering, как наиболее простой и доступный метод для нашей команды.
* Выбрана модель Qwen из числа рассмотренных, поскольку она показала высокую точность и скорость работы.
* Разработана логика обработки ответов от нейросети и преобразования их в формат, пригодный для экспорта в Excel или .
* Реализован графический интерфейс пользователя с помощью библиотеки tkinter, что делает программу удобной в использовании.
* Внедрена возможность загрузки файлов в форматах и XLSX, их обработка и сохранение результата в аналогичном формате.

Проект был реализован в условиях ограниченного времени и ресурсов, что потребовало тщательного планирования и распределения задач между членами команды. Мы следовали намеченному календарному плану, что позволило сосредоточиться на оптимизации и улучшении качества работы системы.

Особое внимание было уделено тестированию и анализу результатов. Мы внедрили механизм обработки ошибок и контроля времени выполнения, чтобы обеспечить стабильную работу даже на системах без GPU. Также была реализована возможность оценки качества генерации с помощью метрики ROUGE, что дало объективное представление о корректности извлечения данных.

Мы считаем, что выбранный подход к решению задачи является эффективным и имеет хорошие перспективы для дальнейшего развития. Программа может быть расширена за счёт поддержки большего количества моделей, а также улучшения пользовательского интерфейса.

В качестве проверки качества произведённого продукта использовались метрики. Проведя оценку на 100 записях, мы получили следующие значения:

{'rouge1': 0.8476643446340415,

'rouge2': 0.6612195324316534,

'rougeL': 0.8164970551334186,

'rougeLsum': 0.8180309869703806}

где:

* **rouge1** - оценивает, насколько сгенерированный текст захватывает ключевые слова или основные термины из эталонного текста.
* **rouge2** - определяет, насколько сгенерированный текст захватывает важные фразы или комбинации слов из эталонного текста.
* **rougeL** - измеряет сходство двух текстов на основе самой длинной общей подпоследовательности (LCS).
* **rougeLsum** - разбивает текст на предложения и вычисляет LCS для каждой пары предложений. Затем он объединяет все баллы LCS в единый показатель.

**LCS** - это длинная последовательность слов, которая встречается в обоих текстах в одном порядке, но не обязательно рядом.

Данные результаты достаточно хороши для такого небольшого проекта.

Данный проект не только стал хорошей практической базой для освоения технологий работы с большими языками моделями, но и продемонстрировал возможности применения ИИ в задачах автоматической нормализации данных.

СПИСОК ЛИТЕРАТУРЫ

1. Qwen: Language Model from Alibaba Cloud [Электронный ресурс]. – URL: https://qwenlm.github.io/
2. Transformers: State-of-the-Art Natural Language Processing [Электронный ресурс] / Hugging Face. – URL: https://huggingface.co/docs/transformers/
3. Tkinter — Python GUI [Электронный ресурс]. – URL: https://docs.python.org/3/library/tkinter.html
4. LLaMA: Open and Efficient Foundation Language Models [Электронный ресурс] / Meta AI. – URL: https://ai.meta.com/llama/
5. DeepSeek: Advanced Language Models [Электронный ресурс]. – URL: https://www.deepseek.com
6. Hugging Face [Электронный ресурс] – URL: https://huggingface.co/

ПРИЛОЖЕНИЕ

GitHub – ­ URL: https://github.com/NEONWolfYT/LLM-Project.git