**НИТУ «МИСИС»**

**Кафедра инженерной кибернетики**

**ОТЧЕТ**

по

**ЛАБОРАТОРНОЙ РАБОТЕ**

*«Исследование возможностей и оценка качества генерации русскоязычных текстов NLP-моделями «семейства» ruGPT-3.х»*

учебная дисциплина «Методы искусственного интеллекта»

**Студент: Осипова Елизавета Андреевна**

**Группа**: \_\_\_\_\_\_**БПМ-21-2**\_\_\_\_\_\_

**Преподаватель: Хонер П. Д.**

**Оценка: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Дата защиты: \_\_\_\_\_\_\_\_\_\_\_\_\_**

**2024 г.**

Оглавление

**[1.](#_Toc192433863)****[ВВЕДЕНИЕ](#_Toc192433863)** [3](#_Toc192433863)

[**1.1.** **Основные цели и задачи работы** 3](#_Toc192433864)

[**1.2.** **Описание решаемой задачи** 4](#_Toc192433865)

[**1.3.** **Средства для разработки программного обеспечения** 6](#_Toc192433866)

[**2.** **Структура проекта** 8](#_Toc192433867)

[**2.1.** **Модуль main.py** 8](#_Toc192433868)

[**2.2.** **Модуль integration.py** 9](#_Toc192433869)

[**2.3.** **Модуль interface.py** 11](#_Toc192433870)

[**3.** **Вывод результатов** 12](#_Toc192433871)

[**4.** **Анализ результатов** 16](#_Toc192433872)

[**4.1.** **Анализ 1 пункта** 16](#_Toc192433873)

[**4.2.** **Анализ 2 пункта** 20](#_Toc192433874)

[**4.3.** **Общий вывод** 25](#_Toc192433875)

[**5.** **Список используемых источников** 26](#_Toc192433876)

[**6.** **Приложение А. Модуль main.py** 27](#_Toc192433877)

[**7.** **Приложение Б. Модуль integration.py** 28](#_Toc192433878)

[**8.** **Приложение В. Модуль interface.py** 30](#_Toc192433879)

# **ВВЕДЕНИЕ**

## **Основные цели и задачи работы**

*Основная цель работы:*

Оценить качество генерации текста NLP-моделью rugpt3small\_based\_on\_gpt2 в зависимости от введенных параметров текста.

*Задачи:*

1) Сгенерировать текст в NLP-модели rugpt3small\_based\_on\_gpt2

а) Сгенерировать аннотацию КНИР, выполняемой в текущем семестре – длина не более 150 слов и не более 10 предложений, проанализировать работу NLP-модели (3 примера);

б) Сгенерировать текст произвольной длины, тематически связанный с выполняемой КНИР (3 примера).

2) Выполнить анализ качества генерации текста

3) Сделать по результатам анализа аргументированный вывод.

*Основной обобщенный функционал, который должен быть реализован в созданном программном обеспечении:*

* получение исходных данных (непосредственный ввод исходных данных пользователем в запущенное приложение посредством элементов интерфейса);
* непосредственное взаимодействие программы с NLP-моделью rugpt3small\_based\_on\_gpt2 (установление контакта; отправка данных для обработки или анализа; получение результатов работы (ответов) от ИИ-сервисов; завершение сеанса взаимодействия);
* отображение результатов работы ИИ-сервиса на экране работающего приложения.

## **Описание решаемой задачи**

Модели языка Transformer (LM) являются основополагающими для методологий и приложений NLP на разных языках. Однако разработке таких моделей специально для русского языка уделялось мало внимания.

RuGPT3Small\_based\_on\_GPT2 — это российская языковая модель, разработанная на основе открытой архитектуры GPT-2, предназначенная для обработки текста на русском языке. Ее разрабатывали как сокращенную и упрощенную версию RuGPT-3 для более легкого использования и меньших требований к ресурсам.

Основные характеристики RuGPT3Small\_based\_on\_GPT2:

1. Базируется на архитектуре GPT-2:

- Использует архитектуру трансформеров с механизмом внимания (Self-Attention).

- Сокращенная версия по сравнению с крупными моделями GPT-3.

2. Предназначена для русского языка:

- Модель обучена на больших корпусах русскоязычных текстов, включая новости, книги, форумы и другие источники.

- Подходит для задач генерации, дополнения текста, продолжения диалога и написания осмысленных текстов на русском.

3. Размер модели и параметры:

- Имеет меньшее количество слоев, нейронов и параметров по сравнению с GPT-3. Это делает ее менее требовательной к вычислительным ресурсам.

- Такая модель способна запускаться даже на персональных компьютерах с использованием GPU (типа NVIDIA 1050/1650/2060) или CPU (с пониженной скоростью).

4. Тензоры и требования к ресурсам:

- Использует веса и тензорные операции для ускоренной обработки.

- Занимает значительно меньше оперативной и видеопамяти, чем GPT-3.

Качество генерации текста для моделей ИИ очень важно. Когда мы задаем вопрос чату, мы хотим получить развернутый и точный ответ на свой вопрос. Порой нам нужен краткий ответ на вопрос.

В данной лабораторной работе я задам параметры для вывода чатом ответов на заданные вопросы. В первом случае это должен быть текст, содержащий менее 150 слов, а в другом случае текст, содержащий более 250 слов.

Для того, чтобы проанализировать качество генерации текста NLP-моделью rugpt3small\_based\_on\_gpt2, мы проведем 3 исследования с каждыми параметрами.

*Итого:*

*Исходные данные:*

1) «Автоматизация процессов очень важна для компании» - Аннотация на данную тему, содержащая на выходе не более 150 слов и не более 10 предложений.

2) «Актуальность автоматизации систем управления состоит в том» - Текст произвольной длины, но не менее 250 слов и 15 предложений.

*Задача:*

Задать данные 2 вопроса NLP-модели rugpt3small\_based\_on\_gpt2 по 3 раза каждый. Получить ответы, а затем проанализировать качество генерации текста.

## **Средства для разработки программного обеспечения**

1. Используемые NLP-модели:

В работе используется NLP-модель RuGPT3Small\_based\_on\_GPT2.

1. Используемый язык программирования:

Python 3.9 (64-bit)

1. Используемые библиотеки и модули:

* Библиотека tkinter

tkinter — это стандартный графический интерфейс Python для создания оконных приложений. Эта библиотека позволяет разрабатывать интерфейсы пользователя (GUI) с кнопками, текстовыми полями, прокруткой и другими элементами.

* Модуль tk

tk - основной модуль tkinter, который предоставляет базовые функции для работы с окнами. Используется для создания окон, кнопок, меток и других элементов GUI.

* Модуль scrolledtext

scrolledtext предоставляет виджет ScrolledText, который является текстовым полем с функцией прокрутки. Удобен для работы с большими текстами.

* Библиотека torch

torch — ядро фреймворка PyTorch, предназначенного для работы с глубоким обучением. Это одна из самых популярных библиотек для работы с нейронными сетями. Она позволяет создавать, обучать и использовать модели машинного обучения.

* Библиотека transformers

transformers — это фреймворк от Hugging Face, предназначенный для работы с самыми современными моделями обработки текста (NLP). Поддерживает модели, основанные на архитектуре Transformer (например, GPT, BERT, T5 и др.).

* Модуль AutoTokenizer

AutoTokenizer - класс, автоматически загружающий корректный токенайзер для модели. Преобразует текст в числовые представления (токены), которые передаются модели. Подходит для работы с любой моделью, поддерживаемой библиотекой Hugging Face.

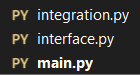
* Модуль AutoModelForCausalLM

AutoModelForCausalLM - класс для создания и загрузки модели, поддерживающей автогегерацию текста (Causal Language Modeling). Используется для задач вроде продолжения текста или генерации на основе заданного контекста.

* Библиотека yaml

yaml — это библиотека для работы с файлами YAML. YAML (YAML Ain't Markup Language) — это популярный формат сериализации данных, человеко-читаемый, распознающий сложные структуры вроде объектов, словарей, списков и строк.

# **Структура проекта**



1. main.py

- Основной файл для запуска проекта.

- Импортирует функции из других модулей.

- Создает конфигурационный файл и загружает его.

- Инициализирует модель и запускает графический интерфейс пользователя (GUI).

2. integration.py

- Отвечает за интеграцию с внешними сервисами.

- Содержит функции для создания и загрузки конфигурации.

- Инициализирует модель и устройство (CPU или GPU).

- Обрабатывает генерацию текста.

3. interface.py

- Управляет графическим интерфейсом проекта с использованием Tkinter.

- Создает окно чата, позволяет пользователю вводить текст и получать ответы от модели.

- Импортирует функцию генерации текста из integration.py для обработки пользовательских запросов.

## **Модуль main.py**

1. Импорт функций:

- create\_gui из модуля interface для создания графического интерфейса.

- initialize\_model, load\_config, create\_config\_file из модуля integration для работы с моделью и конфигурацией.

2. Основной блок выполнения:

- Проверяет, что код выполняется как самостоятельный скрипт.

3. Создание и загрузка конфигурации:

- create\_config\_file(config\_filename) создает файл конфигурации config.yaml с параметрами для модели.

- load\_config(config\_filename) загружает конфигурацию из этого файла.

4. Инициализация модели:

- initialize\_model(config) инициализирует модель и возвращает токенизатор, модель и устройство (например, CPU или GPU).

5. Запуск интерфейса:

- create\_gui(model, tokenizer, device, config) запускает графический интерфейс, который взаимодействует с моделью, используя предоставленную конфигурацию.

## **Модуль integration.py**

Импорт библиотек

- torch: Библиотека для работы с моделями глубокого обучения, поддерживает вычисления на GPU.

- transformers: Из этой библиотеки импортируются AutoTokenizer и AutoModelForCausalLM, которые используются для работы с языковыми моделями.

- yaml: Используется для работы с файлами в формате YAML, что удобно для хранения конфигураций.

Функция create\_config\_file(filename: str)

- Создает конфигурационный файл в формате YAML.

- Содержимое конфигурации:

- Модель:

- "name": Имя модели, которую планируется использовать ("ai-forever/rugpt3small\_based\_on\_gpt2").

- Параметры:

- "max\_new\_tokens": Максимальное количество новых токенов, которые можно сгенерировать.

- "min\_new\_tokens": Минимальное количество новых токенов для генерации.

- "num\_beams": Количество путей для алгоритма поиска с несколькими лучами (beam search).

- "top\_p": Параметр для nucleus sampling, ограничивающий выбор токенов.

- "temperature": Параметр, определяющий случайность в выборе токенов.

- "repetition\_penalty": Штраф за повторение токенов, чтобы избежать избыточных повторов.

- Запись в файл:

- Открывает файл с указанным именем на запись с кодировкой UTF-8.

- Сохраняет конфигурацию в формате YAML.

- Выводит сообщение о том, что файл успешно создан.

Этот код содержит две функции, которые управляют загрузкой конфигурации и инициализацией модели.

Функция load\_config(filename: str)

- Загружает конфигурацию из YAML файла.

- Открывает файл с указанным именем на чтение с кодировкой UTF-8.

- Использует yaml.safe\_load для загрузки содержимого файла в виде Python-словаря.

- Возвращает загруженную конфигурацию.

Функция initialize\_model(config)

- Инициализирует языковую модель и токенизатор на основе конфигурации.

- Получает имя модели из конфигурации (config["model"]["name"]).

- Загружает токенизатор с помощью AutoTokenizer.from\_pretrained.

- Загружает модель с помощью AutoModelForCausalLM.from\_pretrained.

- Определяет устройство для выполнения (использует GPU, если доступно, иначе CPU) с помощью torch.device.

- Переносит модель на выбранное устройство (model.to(device)).

- Выводит сообщение об успешной загрузке модели и устройстве.

- Возвращает токенизатор, модель и устройство.

- Обработка ошибок:

- Если возникает ошибка, выводит сообщение об ошибке и завершает программу с помощью exit().

Этот код реализует функцию generate\_text, которая генерирует текст с использованием языковой модели и заданной конфигурации.

Описание функции generate\_text

Параметры:

- input\_text: str : Входной текст, на основе которого будет производиться генерация.

- model : Объект языковой модели, используемой для генерации.

- tokenizer : Токенизатор для преобразования текста в тензоры и обратно.

- device : Устройство (например, CPU или GPU), на котором выполняются вычисления.

- config : Словарь конфигурации, содержащий параметры для генерации текста.

Процесс выполнения:

1. Извлечение параметров из конфигурации:

- max\_new\_tokens, min\_new\_tokens, num\_beams, top\_p, temperature, repetition\_penalty.

2. Кодирование входного текста:

- Преобразует input\_text в тензор токенов с помощью tokenizer.encode.

- Переносит тензор на устройство, указанное в device.

3. Генерация текста:

- Использует метод model.generate для генерации текста на основе входных токенов и извлеченных параметров.

- Устанавливает do\_sample=True, чтобы включить случайное сэмплирование.

- max\_length и min\_length определяют диапазон длины генерируемого текста.

- no\_repeat\_ngram\_size=2 предотвращает повторение биграмм, повышая разнообразие текста.

4. Декодирование и возврат текста:

- Преобразует сгенерированные токены обратно в строку текста с помощью tokenizer.decode.

- Пропускает специальные токены, такие как <|endoftext|>, используя skip\_special\_tokens=True.

- Возвращает сгенерированный текст.

5. Обработка ошибок:

- Весь процесс обернут в блок try-except для обработки возможных ошибок.

- В случае возникновения ошибки возвращает сообщение об ошибке в виде строки.

## **Модуль interface.py**

1. Импорт библиотек:

- import tkinter as tk : Импортирует библиотеку Tkinter для создания графического интерфейса.

- from tkinter import scrolledtext : Импортирует элемент ScrolledText, который позволяет отображать текст с прокруткой.

- from integration import generate\_text : Импортирует функцию generate\_text, которая, предположительно, отвечает за генерацию ответов чат-бота на основе пользовательского ввода и модели.

2. Функция create\_gui:

- Принимает четыре параметра: model, tokenizer, device и config, которые нужны для работы модели генерации текста.

3. Вложенная функция on\_send\_message:

- Срабатывает при нажатии кнопки "Send".

- Извлекает текст из текстового поля, проверяет, не пуст ли он, и если нет, добавляет сообщение пользователя в журнал чата.

- Вызывает функцию generate\_text для получения ответа от чат-бота и добавляет этот ответ в журнал.

- Прокручивает текстовое поле к последнему сообщению и очищает поле ввода.

4. Создание главного окна:

- root = tk.Tk() : Создает главное окно приложения.

- Устанавливает заголовок окна и позволяет изменять его размер.

5. Настройка сетки для размещения элементов:

- Создает фрейм для чата (chat\_frame) и настраивает его для расширения по размеру окна.

- Создает текстовое поле chat\_log для отображения сообщений, которое позволяет прокручивать текст.

6. Создание фрейма для ввода:

- Создает input\_frame для размещения поля ввода и кнопки "Send".

- Добавляет текстовое поле user\_input для ввода сообщений пользователем и кнопку send\_button, которая вызывает on\_send\_message.

7. Запуск главного цикла:

- root.mainloop() : Запускает основной цикл обработки событий, который удерживает окно открытым и ждет взаимодействия пользователя.

# **Вывод результатов**

Командная строка и созданный кодом интерфейс без вводных данных представлен ниже:

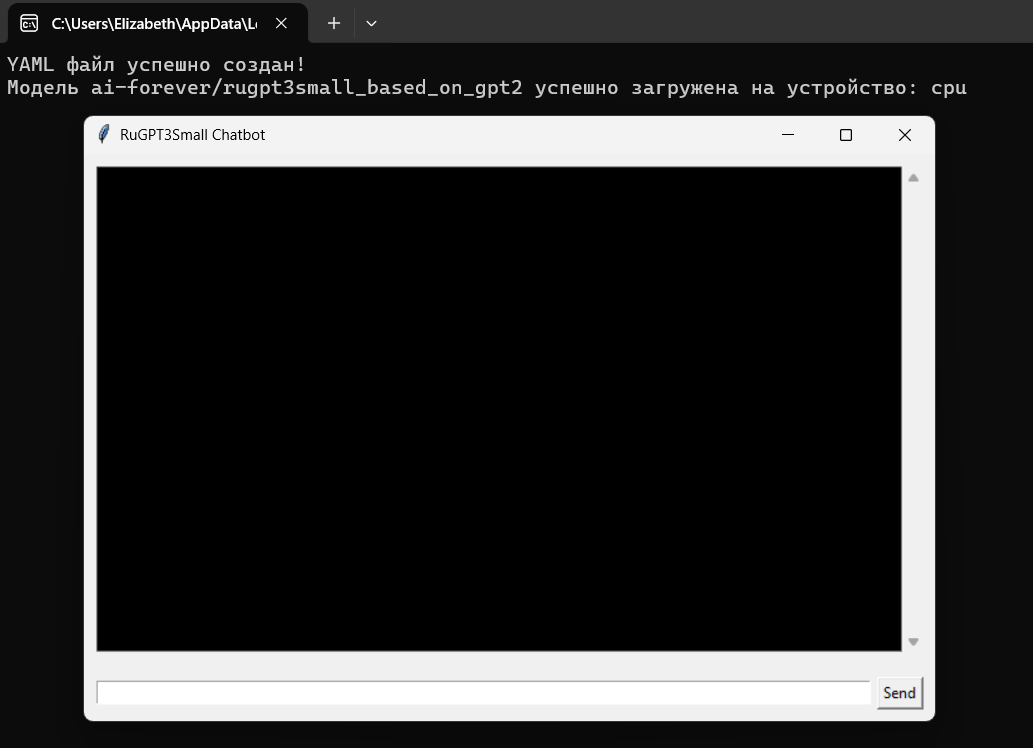


Рисунок 1. Интерфейс

Введем такие параметры:

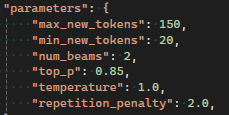


Рисунок 2. Параметры 1

Тогда yaml файл будет выглядеть так:

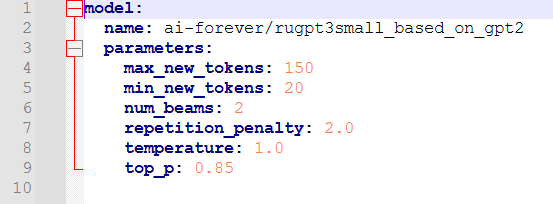


Рисунок 3. Yaml для параметров 1

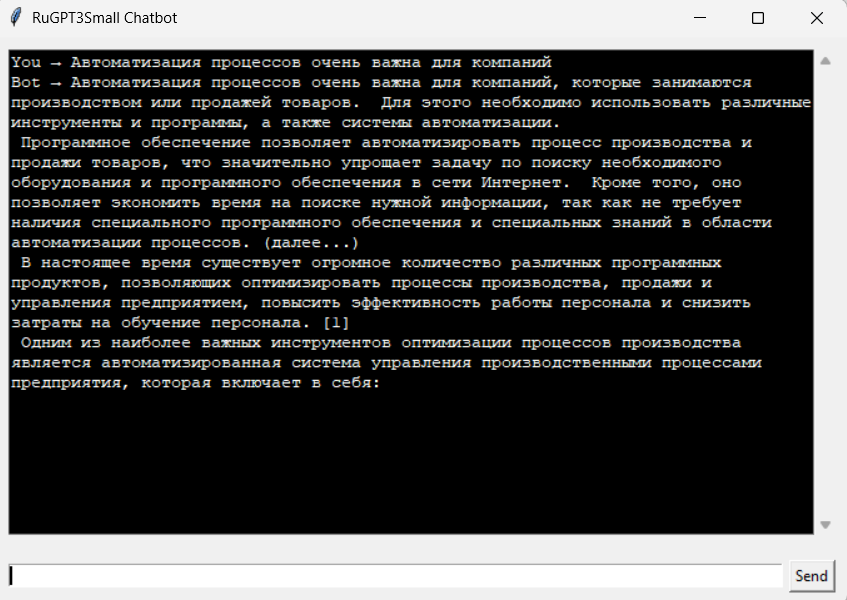
Результаты при вводе: «Автоматизация процессов очень важна для компаний»

Рисунок 4. 1 вывод 1 пункта

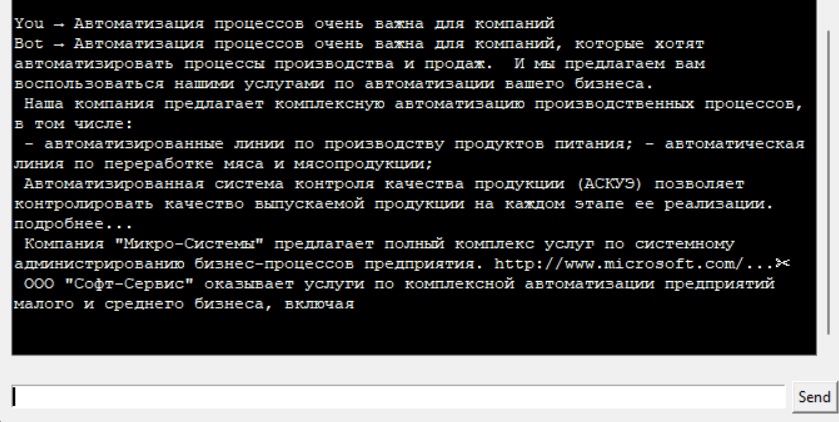


Рисунок 5. 2 вывод 1 пункта

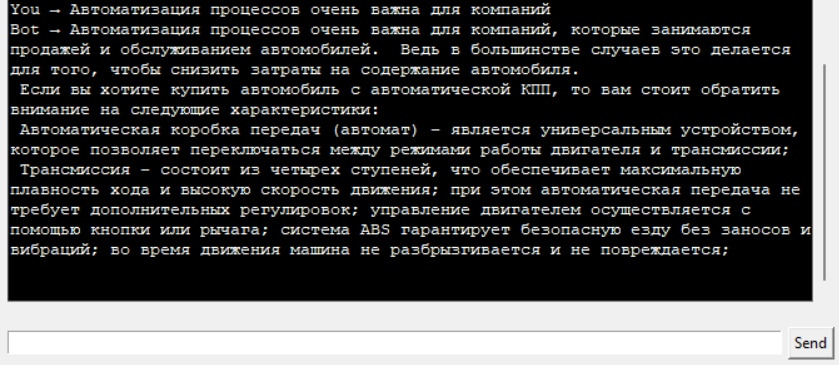


Рисунок 6. 3 вывод 1 пункта

Теперь введем такие параметры:

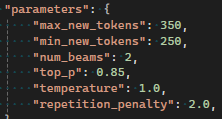
****

Рисунок 7. Параметры 2

Тогда yaml файл будет выглядеть так:

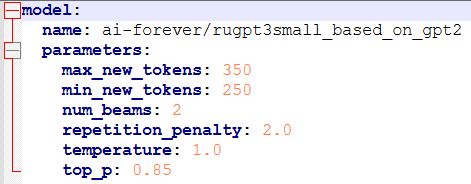


Рисунок 8. Yaml для парметров 2

Результаты при вводе: «Актуальность автоматизации систем управления состоит в том»

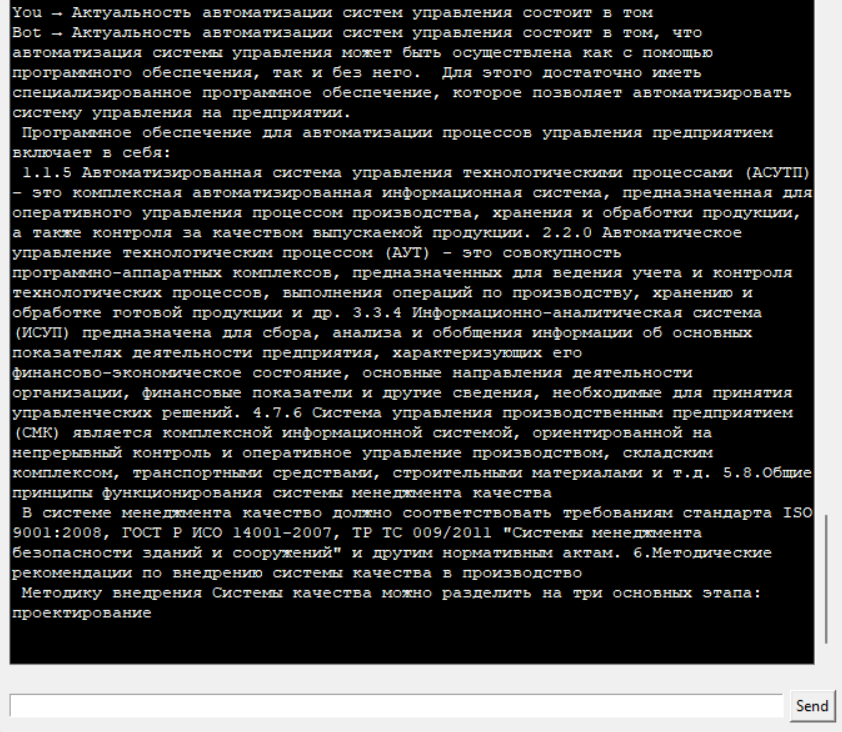


Рисунок 9. 1 вывод 2 пункта

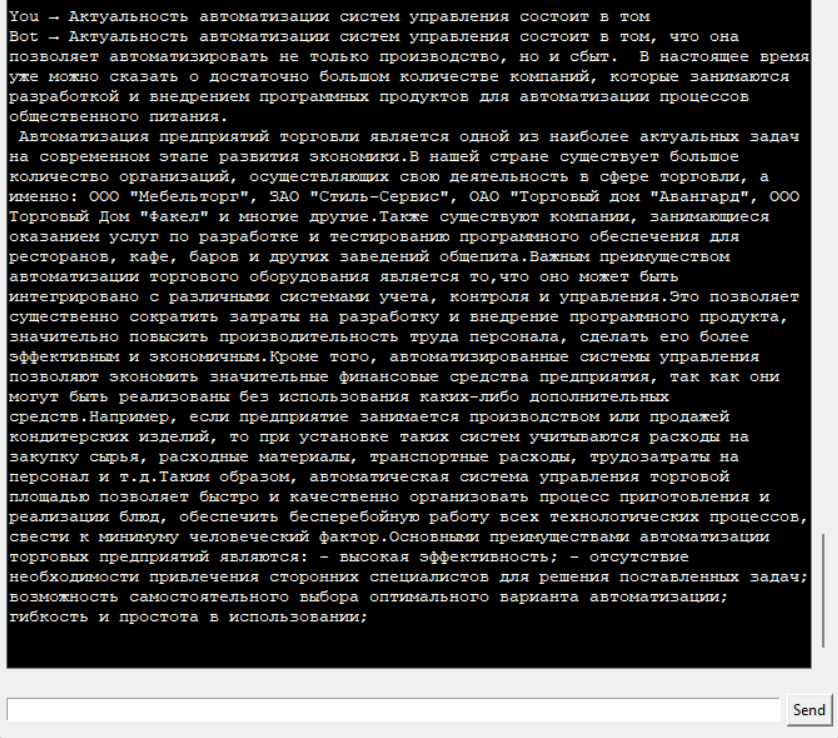


Рисунок 10. 2 вывод 2 пункта

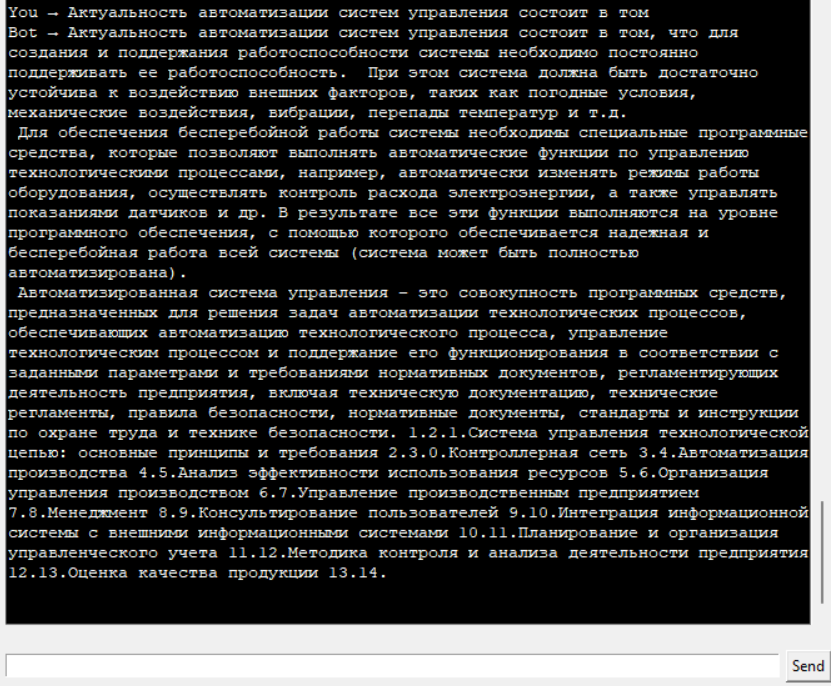


Рисунок 11. 3 вывод 2 пункта

# **Анализ результатов**

## **Анализ 1 пункта**

***1 пункт: «Автоматизация процессов очень важна для компаний»***

***1 вывод:***

Качество сгенерированного текста модели RuGPT-3 Small можно оценить по следующим критериям:

1. Смысловая логика

- Текст релевантен теме: обсуждение автоматизации процессов касается ключевых моментов, таких как:

- Использование программного обеспечения.

- Оптимизация производства и продаж.

- Снижение затрат на обучение персонала.

- Однако некоторые части текста кажутся поверхностными или избыточно общими. Например, фраза: "Программное обеспечение позволяет автоматизировать процесс производства и продажи товаров..." усложнена, не добавляя новой информации.

Оценка: Удовлетворительно.

2. Полнота и детализация

- Модель упоминает важные аспекты автоматизации (настройка, выбор инструментов, выгоды), но завершенность текста страдает. Заключение отсутствует — предложение обрывается на "...включает в себя:", что вызывает ощущение незаконченности.

- Вместо конкретных примеров приводятся абстрактные утверждения.

Оценка: Средне.

3. Грамматика и стиль

- Лексика и грамматика правильные, текст написан на грамотном русском языке.

- Стиль формальный, подходящий для обсуждаемой темы.

- Однако встречается повторение одних и тех же мыслей, что делает текст избыточным. Пример: повтор слов "программное обеспечение" или схожих фраз в соседних предложениях.

Оценка: Хорошо.

4. Когерентность (связанность предложений)

- Предложения связаны логически и плавно следуют друг за другом.

- Фраза "в настоящее время существует огромное количество различных программных продуктов" слишком обобщенная и не раскрывает следующую идею.

Оценка: Средне.

*Итоговая оценка*

Качество текста: 3/5 (удовлетворительно).

Текст демонстрирует потенциал, но ему недостает детализации, конкретных примеров и завершенности. Модель могла бы предложить более специализированный ответ, если бы использовала больше полезных данных по теме (например, названия конкретных инструментов автоматизации).

***2 вывод.***

1. Смысленность и релевантность

- Текст соответствует заявленной теме — автоматизация процессов для бизнеса.

- Упоминается несколько направлений автоматизации, например, производство продуктов питания и переработка мяса, а также контроль качества продукции (АСКУП). Это добавляет конкретики и делает текст более полезным.

- Однако предложение услуг моделирует рекламный стиль — текст больше похож на маркетинговую кампанию, чем на нейтральный обзор автоматизации.

Оценка: Хорошо.

2. Структура текста

- Структура текста относительно последовательная: сначала говорится об актуальности автоматизации, затем перечисляются направления автоматизации и упоминаются компании-поставщики услуг.

- Минусы:

- Не завершённое предложение в конце: "ООО \"Софт-Сервис\" оказывает услуги... включая..." — структура текста кажется обрезанной.

- Первая часть текста вводит рекламный подтекст ("мы предлагаем вам автоматизацию вашего бизнеса"), что снижает практическую ценность текста.

Оценка: Средне.

3. Полнота информации

- Положительный момент: приводятся направления автоматизации и пример с АСКУП. Это делает текст чуть более информативным.

- Негативный момент: тексту не хватает более детального раскрытия тем и обоснования их важности. Упомянуты названия компаний и гиперссылка, но информация о них обрывочна.

Оценка: Удовлетворительно.

4. Грамматика и стиль

- Текст грамматически корректен, ошибки отсутствуют.

- Стиль формальный, однако рекламный характер первых строк и незаконченные мысли снижают его восприятие.

Оценка: Хорошо.

5. Когерентность (связанность)

- Связанность предложений удовлетворительная: текст логично переходит от проблемы автоматизации к услугам и примерам.

- Минус: несоответствие между анонсированным раскрытием темы и фактическим содержанием. Например, перечисление услуг внезапно обрывается.

Оценка: Средне.

*Итоговая оценка*

Качество текста: 3.5/5 (на грани между удовлетворительно и хорошо).

Текст неплохо освещает тему и содержит некоторые элементы конкретики. Однако его пользы могло быть больше, если бы дополнить раскрытие направлений автоматизации, включить больше полезных примеров и убрать маркетинговый стиль, чтобы соблюдать нейтральность.

***3 вывод.***

1. Смысленность и соответствие теме:

- Первая часть текста касается автоматизации процессов для компаний, но затем внезапно переходит к особенностям автомобилей с автоматической коробкой передач. Эта смена темы выглядит непоследовательной.

- Тема автоматической коробки передач раскрывается с рядом общих утверждений, которые описывают её назначение и преимущества. Но многие фразы кажутся повторяющимися и недостаточно глубокими.

- Некорректные утверждения: "во время движения машина не разбалтывается и не повреждается" звучит странно и сомнительно с точки зрения фактов.

Оценка: Низкая релевантность из-за несоответствия основных частей текста и сомнительных утверждений.

2. Структура текста:

- Текст состоит из двух частей, которые плохо связаны между собой:

- Первая часть о компаниях и автоматизации;

- Вторая часть об автоматических коробках передач.

- Продолжение идеи бедное: текст неприятно обрывается и не подводит к логическому завершению.

Оценка: Средне.

3. Полнота информации:

- Тема автомобилей раскрыта поверхностно. Указаны только базовые характеристики автоматической коробки передач.

- Для улучшения текста стоило бы добавить больше технических данных и структурировать преимущества более четко.

Оценка: Удовлетворительно.

4. Грамматика и стиль:

- Грамматических ошибок немного, текст читается достаточно грамотно.

- Стиль текста формальный, но несколько утверждений звучат научно необоснованно, например, про то, что система ABS гарантирует отсутствие "заносов и вибраций".

Оценка: Средне.

5. Когерентность (логичность и последовательность):

- Логика и связность нарушены из-за смены темы в середине текста. Переход от автоматизации бизнеса к коробкам передач кажется случайным и несвязанным.

Оценка: Низко.

*Итоговая оценка:*

Качество текста: 2.5/5.

- Текст содержит два слабо связанных между собой блока без нужного логического перехода.

- Низкая информативность, повторяемость идей в тематике автоматических коробок передач.

- Для улучшения текста требуется более чёткое соответствие единой теме, расширение информации и устранение сомнительных утверждений.

## **Анализ 2 пункта**

***2 пункт: «Актуальность автоматизации систем управления состоит в том»***

***1 вывод:***

1. Смысленность и соответствие теме:

Текст тематически посвящён автоматизации систем управления. На первый взгляд он насыщен терминологией, релевантной данной теме, и даёт краткие описания нескольких систем: АСУТП, АУТ, ИАСУ, СМК. Однако:

- Описание систем даётся без структуры, иногда теряется чёткая логика.

- Некоторые фразы кажутся излишне громоздкими и трудными для понимания.

- Информация выглядит копированием "из учебника", без попытки адаптировать её под целевую аудиторию.

Оценка: Среднее соответствие теме (3/5), но избыточная сложность изложения.

2. Структура текста:

- Текст структурирован в виде перечислений, что делает его удобным для формального прочтения.

- Однако переходы между пунктами выглядят резкими, отсутствуют необходимые подводки или объяснения, как пункты связаны между собой.

- Завершающая часть текста теряет фокус и не возвращается к основной теме.

Оценка: Удовлетворительная структура (3/5).

3. Полнота информации:

- Приведённый текст содержит широкий, но поверхностный обзор различных систем автоматизации.

- Многое из перечисленного не раскрыто: описание стандарта ISO или "методики внедрения" упоминаются, но их содержание оставлено на уровне общей информации без конкретики.

Оценка: 3/5 (широкий охват, но многие аспекты остаются поверхностными).

4. Грамматика и стиль:

- Грамматических ошибок нет, изложение формально, в стиле научно-технической документации.

- Однако стиль слишком монотонный и перегружен канцеляризмами, что снижает удобство восприятия текста.

Оценка: 4/5.

5. Когерентность (логичность и последовательность):

- Последовательность текста оставляет желать лучшего: перечисления начинаются без пояснений и логического перехода между пунктами.

- Ключевые идеи по теме автоматизации управления в конечном итоге выглядят несвязанными.

Оценка: 2/5.

*Итоговая оценка:*

Качество текста: 3/5.

- Технические понятия раскрыты частично, но подача теряет фокус и читабельность. Текст требует улучшения структуры, логических связей и более понятной формулировки идей.

***2 вывод.***

1. Смысленность и соответствие теме:

Текст тематически посвящён автоматизации систем управления, фокусируясь на сфере торговли и общественного питания.

- Он достаточно полно раскрывает значимость автоматизации для этих сфер, с примерами и описанием преимуществ использования специализированного программного обеспечения.

- Упоминание компаний выглядит как перечисление ради объёма текста, никак не продолжено или аналитически осмыслено.

- Некоторые аспекты (например, интеграция с системами учёта, снижение затрат, улучшение точности учёта) упоминаются поверхностно.

Оценка: Хорошая релевантность (4/5).

2. Структура текста:

- Структура текста слабо организована. Начало текста вводит тему актуальности автоматизации, но дальнейшее описание идёт хаотично.

- Взаимосвязь между перечисленными аспектами недостаточно подчёркнута.

- Нет разделения на понятные логические блоки (например, задачи, процессы, преимущества), что ухудшает восприятие информации.

Оценка: Посредственная структура (3/5).

3. Полнота информации:

- Текст затрагивает ключевые вопросы автоматизации в торговле (оптимизация продаж, снижение затрат, повышение производительности персонала) и описывает её возможности.

- Однако информация лишена конкретных примеров успешного внедрения, цифр, фактов, или описания технологий. Всё остаётся на уровне общих фраз.

Оценка: Удовлетворительная полнота (3/5).

4. Грамматика и стиль:

- Грамматических и синтаксических ошибок, очевидно, нет.

- Стиль выдержан в формально-деловом ключе, но он перегружен канцеляризмами, что может затруднить чтение непрофессиональной аудитории.

- Предложения местами излишне громоздкие, что снижает удобство восприятия.

Оценка: Хорошо (4/5).

5. Когерентность (логичность и последовательность):

- Логичность текста страдает из-за чрезмерной длины некоторых предложений и отсутствия явных переходов между темами.

- Информация о компаниях упомянута, но не продолжена, что создаёт впечатление "брошенной" мысли.

- Общая последовательность изложения нуждается в улучшении.

Оценка: 3/5.

*Итоговая оценка:*

Качество текста: 3.5/5.

Текст объясняет важность автоматизации в торговле и общественном питании, но имеет слабую структуру, теряет логическую связанность и ограничивается общими фразами. Необходима доработка структуры, детализация и сокращение громоздких оборотов.

***3 вывод.***

1. Смысленность и соответствие теме:

Текст посвящён автоматизации систем управления и описывает как общие принципы, так и технические аспекты работы систем. Основные функции автоматизации, такие как управление технологическими процессами, описаны кратко, но без глубины.

- Наличие блоков, пронумерованных от 1.1 до 13.14, создаёт ощущение неполноты, так как они не раскрыты.

- Некоторые мысли поверхностны, либо не имеют достаточно логического завершения.

Оценка: Удовлетворительная смысловая наполненность (3/5).

2. Структура текста:

- Текст построен без явного деления на разделы, что делает его восприятие сложным.

- Список под номерами (1.1...13.14) заявлен, но полностью не раскрыт, что создаёт впечатление незавершённости.

- Отсутствие подзаголовков или других способов структурирования затрудняет понимание.

Оценка: Посредственная структура (2/5).

3. Полнота информации:

- Описаны общие требования к автоматическим системам, такие как устойчивость к внешним воздействиям и надёжность функционирования. Однако текст не даёт примеров успешного применения, реальных кейсов или конкретных технологий.

- Большая часть текста сосредоточена на вводной информации, что создаёт впечатление избыточности.

Оценка: Уровень детализации низкий (3/5).

4. Грамматика и стиль:

- Грамматических ошибок нет, однако стиль формально-деловой слишком перегружен канцеляризмами.

- Повторение слов и оборотов ("необходимо", "система") делает текст менее читабельным и монотонным.

- Структура предложений местами усложнённая, но в целом соответствует деловому стилю.

Оценка: Хорошо с некоторыми недостатками (4/5).

5. Когерентность (логичность и последовательность):

- Логика построения текста нарушается из-за введения номеров пунктов (1.1...13.14), которые не получают полного объяснения.

- Повествование скачет между общими фразами и техническими деталями без чёткой связи.

Оценка: Посредственная когерентность (2/5).

*Итоговая оценка:*

Общая оценка текста: 2.8/5.

Текст страдает от отсутствия структуры, завершённости и логически стройного изложения. Для улучшения:

- Чётко делить текст на разделы с заголовками.

- Раскрыть нумерованные пункты с точки зрения содержания.

- Добавить реальные примеры или иллюстрации для практического понимания.

## **Общий вывод**

Итоговая оценка качества текста, сгенерированного NLP-моделью rugpt3small\_based\_on\_gpt2: 3/5 (удовлетворительно).

Текст показывает средний уровень проработки темы, но его основные недостатки заключаются в недостаточной структурированности, поверхностном изложении и отсутствии конкретики. Проблемы включают слабую логику переходов между частями текста, повторяемость идей и недостаточное использование практических примеров.

Для повышения качества текста рекомендуется:

- Добавить логическую структуру (использование заголовков, перечислений и связок между блоками).

- Углубить детали с использованием примеров, терминологии и примеров из реальной практики.

- Исключить избыточные, громоздкие фразы и улучшить фокус на основных идеях.

Текст оценивается как потенциально полезный, но требует значительных доработок для достижения высокого уровня.

# **Список используемых источников**

1. <https://huggingface.co/models?pipeline_tag=summarization&sort=likes> – сайт с моделями ИИ
2. <https://qna.habr.com/>
3. <https://dzen.ru/>

# **Приложение А. Модуль main.py**

from interface import create\_gui

from integration import initialize\_model, load\_config, create\_config\_file

if \_\_name\_\_ == "\_\_main\_\_":

# Создание и загрузка конфигурации

config\_filename = "config.yaml"

create\_config\_file(config\_filename)

config = load\_config(config\_filename)

# Инициализация модели

tokenizer, model, device = initialize\_model(config)

# Запуск интерфейса

create\_gui(model, tokenizer, device, config)

# **Приложение Б. Модуль integration.py**

import torch

from transformers import AutoTokenizer, AutoModelForCausalLM

import yaml

def create\_config\_file(filename: str):

config = {

"model": {

"name": "ai-forever/rugpt3small\_based\_on\_gpt2",

"parameters": {

"max\_new\_tokens": 350,

"min\_new\_tokens": 250,

"num\_beams": 2,

"top\_p": 0.85,

"temperature": 1.0,

"repetition\_penalty": 2.0,

},

}

}

with open(filename, "w", encoding="utf-8") as file:

yaml.dump(config, file, default\_flow\_style=False)

print("YAML file created successfully!")

def load\_config(filename: str):

with open(filename, "r", encoding="utf-8") as yaml\_file:

return yaml.safe\_load(yaml\_file)

def initialize\_model(config):

try:

model\_name = config["model"]["name"]

tokenizer = AutoTokenizer.from\_pretrained(model\_name)

model = AutoModelForCausalLM.from\_pretrained(model\_name)

device = torch.device("cuda" if torch.cuda.is\_available() else "cpu")

model.to(device)

print(f"model {model\_name} successfully loaded to device: {device}")

return tokenizer, model, device

except Exception as e:

print(f"Model loading error: {e}")

exit()

def generate\_text(input\_text: str, model, tokenizer, device, config) -> str:

try:

max\_tokens = config["model"]["parameters"]["max\_new\_tokens"]

min\_tokens = config["model"]["parameters"]["min\_new\_tokens"]

num\_beams = config["model"]["parameters"]["num\_beams"]

top\_p = config["model"]["parameters"]["top\_p"]

temperature = config["model"]["parameters"]["temperature"]

repetition\_penalty = config["model"]["parameters"]["repetition\_penalty"]

input\_ids = tokenizer.encode(input\_text, return\_tensors="pt").to(device)

output\_ids = model.generate(

input\_ids,

do\_sample=True,

max\_length=len(input\_ids[0]) + max\_tokens,

min\_length=len(input\_ids[0]) + min\_tokens,

num\_beams=num\_beams,

top\_p=top\_p,

temperature=temperature,

repetition\_penalty=repetition\_penalty,

no\_repeat\_ngram\_size=2,

)

return tokenizer.decode(output\_ids[0], skip\_special\_tokens=True)

except Exception as e:

return f"Error when generating text: {e}"

# **Приложение В. Модуль interface.py**

import tkinter as tk

from tkinter import scrolledtext

from integration import generate\_text

def create\_gui(model, tokenizer, device, config):

def on\_send\_message():

user\_message = user\_input.get().strip()

if user\_message:

chat\_log.insert(tk.END, f"You → {user\_message}\n")

response = generate\_text(user\_message, model, tokenizer, device, config)

chat\_log.insert(tk.END, f"Bot → {response}\n\n")

chat\_log.see(tk.END)

user\_input.delete(0, tk.END)

root = tk.Tk()

root.title("RuGPT3Small Chatbot")

root.resizable(True, True)

root.grid\_rowconfigure(0, weight=1)

root.grid\_columnconfigure(0, weight=1)

chat\_frame = tk.Frame(root)

chat\_frame.grid(row=0, column=0, sticky="nsew", padx=10, pady=10)

chat\_frame.grid\_rowconfigure(0, weight=1)

chat\_frame.grid\_columnconfigure(0, weight=1)

chat\_log = scrolledtext.ScrolledText(

chat\_frame,

wrap=tk.WORD,

state="normal",

bg="black",

fg="white"

)

chat\_log.grid(row=0, column=0, sticky="nsew")

input\_frame = tk.Frame(root)

input\_frame.grid(row=1, column=0, sticky="ew", padx=10, pady=10)

input\_frame.grid\_columnconfigure(0, weight=1)

user\_input = tk.Entry(input\_frame)

user\_input.grid(row=0, column=0, sticky="ew", padx=(0, 5))

send\_button = tk.Button(input\_frame, text="Send", command=on\_send\_message)

send\_button.grid(row=0, column=1)

root.mainloop()