

Федеральное государственное автономное образовательное учреждение
высшего образования «Национальный исследовательский университет
«Высшая школа экономики»

Факультет компьютерных наук
Основная образовательная программа
Прикладная математика и информатика

ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА
ПРОГРАММНЫЙ ПРОЕКТ НА ТЕМУ
"ПРИМЕНЕНИЕ МЕТОДОВ АВТОМАТИЧЕСКОГО СОКРАЩЕНИЯ ТЕКСТА
ДЛЯ АННОТИРОВАНИЯ СТАТЕЙ НОВОСТНОГО ПОРТАЛА"

Выполнил студент группы 193, 4 курса,
Кульпин Павел Леонидович

Руководитель ВКР:
Соколов Евгений Андреевич, доцент ФКН НИУ ВШЭ

Соруководитель ВКР:
Лобачевский Семён Михайлович, административный сотрудник НИУ
ВШЭ

Москва 2023

Содержание

1	Abstract	4
2	Аннотация	4
3	Ключевые термины	5
4	Введение	5
5	Метрики качества	7
5.1	BLEU	7
5.2	ROUGE-N	9
5.3	ROUGE-L и ROUGE-Lsum	9
6	Набор данных	11
6.1	Предобработка данных	12
6.2	Фильтрация объектов	13
6.3	Расчет статистик	13
6.4	Примеры объектов	14
7	Обзор предшествующих работ	17
7.1	Экстрактивный подход.	17
7.1.1	Алгоритмические методы	17
7.1.2	Глубинное обучение в экстрактивной суммаризации . . .	19
7.2	Абстрактный подход	21
7.2.1	Наиболее производительные архитектуры	21
7.2.2	Токенайзеры	23
7.2.3	Критерии выбора предобученных моделей	24
7.2.4	Мультиязычные модели	24
7.2.5	Русскоязычные модели	25
8	Тестирование моделей без дообучения	27

8.1	Бейзлайн	27
8.2	mT5	28
8.3	Gazeta.ru: mBART, ruT5, ruGPT3	28
8.4	Уменьшенная ruT5	29
8.5	Остальные модели	29
8.6	Результаты тестирования	30
9	Файн-тюнинг избранных моделей	30
10	Результаты файн-тюнинга	32
10.1	Значения метрик качества	32
10.2	Примеры генераций	33
10.3	Выводы и выбор моделей	37
11	Имплементация облачного сервиса	37
11.1	Обработка данных на стороне сервера	38
11.2	Пользовательский интерфейс	40
12	Имплементация автоматического дообучения моделей	42
13	План дальнейших работ	43
14	Заключение	44

1 Abstract

Automatic text reduction, a.k.a. text summarization, is one of the most common sequence-to-sequence tasks in the field of Natural Language Processing (NLP). It is a challenging problem that involves reducing the length of a given text while retaining its most important information. This work reviews various methods for text summarization, including extractive and abstractive approaches, and provides an overview of different deep learning architectures that have been applied to this task, as well as user interface implementation tools and backend technologies. The work then focuses on the training of several neural network models for Russian texts, using a dataset of news articles from the High School of Economics news platform, and analyzing their results with different text quality metrics. The final stage of the work involves deploying the resulting text summarization system to the university's website.

2 Аннотация

Автоматическое сокращение текста, или суммаризация текстов, является одной из наиболее распространенных задач в области обработки естественного языка (NLP). Постановка задачи заключается в уменьшении длины заданного текста при сохранении в нем наиболее важной информации. В работе рассматриваются различные методы резюмирования текста, включая экстрактивный и абстрактный подходы, приводится обзор различных архитектур глубинного обучения, используемые для решения поставленной задачи, различных инструментов имплементации пользовательского интерфейса и технологий, отвечающих за обработку данных на стороне сервера. Затем фокус работы смещается в сторону обучения нескольких нейросетевых моделей для обработки русскоязычных текстов, с использованием набора данных, составленного из новостных статей, размещенных на новостной платформе Высшей Школы Экономики, и анализа их результатов с помощью различ-

ных метрик качества. Заключительный этап работы предполагает размещение полученной системы аннотирования текстов на сайте университета.

3 Ключевые термины

Глубинное обучение, обработка естественного языка, набор данных, полнотекстовое приложение для аннотирования текстов.

4 Введение

В настоящее время пользователи интернета ежечасно сталкиваются с таким объемом новой информации, что успевать обрабатывать ее во всех деталях, необходимых для составления полноценной картины происходящих событий и фактов становится невозможно. Решением этой проблемы становятся краткие изложения книг, сводки новостных статей, абстракты научных или академических работ и так далее. Тем не менее, написание такого рода аннотаций также является сложной задачей, требующей нетривиальных усилий от автора текста, поскольку она включает в себя изучение источника, отделение релевантной информации от нерелевантной, формулирование основной идеи исходного текста, написание и редактирование результата. В целях снижения нагрузки на работников данной сферы и автоматизации редакционных процессов человеческий труд в такой задаче может быть полностью или частично заменен нейросетевыми решениями, способными обрабатывать данные в значительной степени быстрее человека.

Исходная постановка задачи суммаризации текста проста: имеется текст, содержащийся в одном или нескольких документах, требуется сократить его длину, сохраняя его главную мысль и основные понятия. Тем не менее, задача является предметом исследований уже на протяжении нескольких десятилетий, а самые ранние работы относятся еще к 1950-м годам. Современные под-

ходы к автоматическому резюмированию текста с использованием алгоритмов машинного обучения были представлены в конце 1990-х - начале 2000-х годов, а [Sparck Jones et al. \(1999\)](#) считается одной из самых ранних работ по суммаризации текстов. С тех пор технологическое развитие и с точки зрения скорости компьютерных вычислений, и с точки зрения подходов и методов глубинного обучения ушло далеко вперед, и нейронные сети используются для решения самых разных прикладных и теоретических задач.

Цель данной работы - провести тщательный анализ различных методов, достигавших State-Of-The-Art (SOTA) результатов в задаче суммаризации текста за последнее время, и подходов к решению этой задачи, включая экстрактивный, который может быть основан на алгоритмических методах, не требующих технологий глубинного обучения или машинного обучения, и абстрактный, подразумевающий использование sequence-to-sequence моделей нейронных сетей, получающих на вход исходный текст и на выходе возвращающих его сокращенную версию. Впоследствии определенный в результате проведенного анализа набор нейронных сетей будет обучен под имеющуюся задачу на собственном датасете, собранном из статей новостного портала Высшей Школы Экономики. Нейросетевые модели, показавшие наилучшие результаты как с точки зрения метрик качества, так и с точки зрения визуального анализа, станут частью облачной системы для аннотирования текстов с необходимым пользовательским интерфейсом и программой обработки данных на стороне сервера, с привязанными к ней вычислительными мощностями, предоставленными университетом. Впоследствии данная система может быть внедрена в кабинеты редакторов на сайте новостной платформы. Итоговый инструмент будет позволять работникам не только автоматизировать процесс написания аннотаций, но и дообучать модель нейронной сети по мере расширения базы данных новостных статей, не обладая при этом опытом работы в сфере глубинного обучения. Весь код будет написан на Python.

5 Метрики качества

В первую очередь, необходимо определить метрики качества, по которым различные подходы и архитектуры могут быть сравнены между собой. Следующие метрики позволяют сравнивать два текста: первый - эталонный, второй - полученный в результате работы алгоритма или нейросети. Стоит заранее отметить, что для описанных ниже метрик существуют варианты их расширения для ситуаций наличия многих эталонных текстов на один сгенерированный, но в данном случае они нас не интересуют. Для корректной интерпритации обозначений, введем следующие понятия:

- N-грамма - есть непрерывная последовательность из N элементов из данного объекта текста.
- Лемма - основная форма слова. В частности, в русском языке, часто под леммой подразумевается корень слова.
- Наибольшая общая подпоследовательность.

Пусть пусть имеются последовательность $X = \{x_1, x_2, x_3, \dots\}$ и $Y = \{y_1, y_2, y_3, \dots\}$. Y называется подпоследовательностью X тогда и только тогда, когда существует строго возрастающая последовательность индексов $I = \{i_1, i_2, i_3, \dots\}$ такая, что для всех j верно следующее: $x_{i_j} = y_j$. Следовательно, наибольшей общей подпоследовательностью двух последовательностей X_1 и X_2 является такая последовательность Y , что Y является подпоследовательностью и для X_1 , и для X_2 , и при этом не существует другой последовательности Y' , являющейся также подпоследовательностью и для X_1 , и для X_2 , и при этом длина Y' больше длины Y .

5.1 BLEU

В статье [Papineni et al. \(2014\)](#) авторы представили метрику BLEU. Метрика принимает на вход сгенерированный текст и эталонный и возвращает зна-

чения от 0 до 1, где 1 может быть возвращена, только если сгенерированный текст идентичен эталонному. Значение метрики BLEU считается следующим образом:

- 1 В первую очередь производится расчет модифицированной точности N-грамм (англ. modified N-gram precision) p_n . Для сгенерированного текста считается количество N-грамм, которые также присутствуют и в эталонном тексте. Полученное значение делится на общее число N-грамм в сгенерированном тексте. Чтобы избежать завышения значения метрики, для всех N-грамм в сгенерированном тексте вводится максимальное допустимое количество повторений, сверх которого счетчик для данной N-граммы останавливается. Такой алгоритм повторяется для всех длин $n \in [1; N]$.

- 2 Затем по всем значениям n считается геометрическое среднее G полученных значений:

$$G = \left(\prod_{n=1}^N p_n \right)^{1/N} \quad (1)$$

- 3 Далее, в целях избежания ситуации, в которой лучшее значение метрики получает текст, состоящий из, например, всего лишь одного слова, но зато содержащегося в эталонном тексте с тем же набором морфем, рассчитывается штраф за краткость (brevity penalty) BP :

$$BP = \begin{cases} 1, & \text{if } c > r \\ e^{(1-r/c)}, & \text{if } c \leq r \end{cases}, \quad (2)$$

где c - длина сгенерированного текста, а r - длина эталонного.

- 4 Итоговое значение BLEU-метрики вычисляется так:

$$BLEU = BP \cdot G \quad (3)$$

Таким образом, BLEU по большей части штрафует сгенерированный текст за содержание, выходящее за рамки содержания эталонного текста.

5.2 ROUGE-N

Метрика ROUGE-N, где N является длиной интересующих N-грамм, представленная в статье [Chin-Yew Lin \(2004\)](#), вычисляется из перекрытия неповторяющихся N-грамм в эталонном и сгенерированном текстах, и может быть рассчитана для любого желаемого $N \in \mathbb{N}$. Метрика считается по следующей формуле:

$$ROUGE - N = \frac{\sum_{S \in Ref} \sum_{Ngram \in S} Count_{match}(Ngram)}{\sum_{S \in Ref} \sum_{Ngram \in S} Count(Ngram)}, \quad (4)$$

где Ref - множество эталонных текстов, $Count_{match}(gram_n)$ - число вхождений N-граммы в оба текста одновременно, а $Count(gram_n)$ - число вхождений N-граммы в эталонный текст.

Можно заметить, что ROUGE-N в значительной степени контролирует текст с точки зрения полноты содержания относительно эталонного примера.

5.3 ROUGE-L и ROUGE-Lsum

Авторство ROUGE-L также принадлежит [Chin-Yew Lin \(2004\)](#). Для ее расчета необходимо найти длину максимальной подпоследовательности лемм, входящей и в эталонный текст ref , и в сгенерированный - out : $LCS(ref, out)$, из которой вычисляется полнота (англ. Recall) и точность (англ. Precision):

$$R_{LCS} = \frac{LCS(ref, out)}{n} \quad (5)$$

$$P_{LCS} = \frac{LCS(ref, out)}{m}, \quad (6)$$

где n и m - длины эталонного и сгенерированного текста в леммах.

В итоге ROUGE-L рассчитывается, аналогично F1-мере:

$$ROUGE - L = \frac{(1 + \beta^2) \cdot P_{LCS} \cdot R_{LCS}}{\beta^2 \cdot P_{LCS} + R_{LCS}}, \quad (7)$$

где β - коэффициент, обычно выставляемый достаточно большим, чтобы приоритезировать полноту над точностью.

Метрика ROUGE-Lsum является обобщением метрики ROUGE-L на уровне предложений, заключающимся в модификации вычисления LCS . Здесь, для каждого предложения сгенерированного текста ищется предложение эталонного текста, имеющего наибольшее LCS . Далее все полученные значения суммируются, следовательно:

$$LCS_{modified}(ref, out) = \sum_{S \in out} \max_{R \in ref} (LCS(R, S)), \quad (8)$$

Затем производится расчет полноты и точности, аналогично ROUGE-L:

$$R_{LCS_{modified}} = \frac{LCS_{modified}(ref, out)}{n} \quad (9)$$

$$P_{LCS_{modified}} = \frac{LCS_{modified}(ref, out)}{m}, \quad (10)$$

где n и m - также длины эталонного сгенерированного текста в леммах.

Следовательно,

$$ROUGE - Lsum = \frac{(1 + \beta^2) \cdot P_{LCS_{modified}} \cdot R_{LCS_{modified}}}{\beta^2 \cdot P_{LCS_{modified}} + R_{LCS_{modified}}}, \quad (11)$$

где β - коэффициент, аналогичный коэффициенту в ROUGE-L.

Итак, описанные выше метрики были выбраны по причине того, что они дополняют друг друга. BLEU является основанной на точности метрикой, не учитывающей полноту, а значит, не штрафующей сгенерированный текст за

недостаток информации. Важно, что штраф за краткость (BP) не эквивалентен штрафу за недостаток информации, а оперирует лишь длинами поданных текстов. Метрика ROUGE-N, наоборот, ориентирована на полноту, более четко отслеживая пропущенные фрагменты информации в сгенерированном резюме. Вместе они обеспечивают баланс этих критериев, что критично для оценки качества суммаризации. Тем не менее, обе метрики полагаются на точное соответствие N-грамм и не очень хорошо справляются с, например, переформулировками. Устранить этот недостаток в некоторой степени помогают ROUGE-L и ROUGE-Lsum, использующие максимальную общую подпоследовательность и обеспечивающие более точную оценку связности и структуры резюме. Для удобства восприятия во всех дальнейших разделах значения метрик будут умножены на 100.

6 Набор данных

Для выполнения проекта, со стороны Высшей Школы Экономики был предоставлен доступ к базе данных новостного портала¹. Каждый объект предоставленного набора данных содержал следующие поля:

- 1 'id' - идентификационный номер статьи.
- 2 'title' - заголовок статьи.
- 3 'annotation' - аннотация статьи.
- 4 'date' - дата публикации.
- 5 'body' - HTML-код в текстовом формате, содержащий текст статьи и имеющие к ней отношение объекты.

¹ <https://www.hse.ru/news/> (01.02.2023)

6.1 Предобработка данных

При анализе текстовых объектов, полученных из веб-страниц, необходимо корректно предобработать имеющиеся данные. Важным этапом такой предобработки является очистка текста от фрагментов HTML-кода. Одним из подходов к решению этой задачи представляет библиотека BeautifulSoup² - гибкий инструмент для парсинга и обработки HTML и XML документов на языке Python. BeautifulSoup позволяет создать объект, содержащий структуру HTML-документа, и извлечь из него только текстовую информацию, автоматически удаляя все HTML-теги и прочие сопутствующие артефакты.

После извлечения текста из HTML-структуры, он все еще может содержать различные артефакты, контроль которых остался за рамками возможностей BeautifulSoup. Такими объектами могут являться, например, ссылки, хранимые в виде простого текста, адреса электронных почт, номера телефонов или различные специальные символы. Решить такую задачу можно с применением регулярных выражений, воспользовавшись функционалом библиотеки re³. Функция "sub" данной библиотеки пропускает поданный текст через регулярное выражение и заменяет все фрагменты, попадающие под его условия, на указанную в аргументе функции последовательность символов (в нашем случае это пустая последовательность). Таким образом, необходимо создать регулярные выражения, которые будут описывать все виды веб-ссылок, адресов электронных почт, номера телефонов, а также всех символов, не являющихся буквами, цифрами, пробелами, знаками препинания, скобками, кавычками или специальными символами переноса строки и пробела. Однако после удаления подобных фрагментов сохранятся не только значимые элементы текста, но и множественные смежные пробельные символы, прежде разделявшие вычищенные объекты. Соответственно, такие фрагменты необходимо заменить одним пробелом.

² <https://www.crummy.com/software/BeautifulSoup/bs4/doc/> (12.03.2023)

³ <https://docs.python.org/3/library/re.html> (12.03.2023)

6.2 Фильтрация объектов

Так как при сборе данных не учитывались никакие специальные фильтрационные параметры, фильтрацию необходимо провести на данном этапе. В первую очередь, по очевидным причинам из набора данных были удалены примеры, содержащие пустые строки или объекты типа "None", хотя бы в одном из полей 'body', 'annotation'. Также были удалены примеры, содержащие в поле 'body' менее 100 или более 15000 символов.

6.3 Расчет статистик

Далее проведем расчет и анализ статистик полученного набора данных, который по итогам процессов обработки и фильтрации содержал 176428 пар статья-аннотация. В таблице 6.1 представлен всесторонний анализ набора данных, полученный в результате подсчета различных статистических показателей на уровне лемм, слов и предложений для всех текстов и всех аннотаций. Среди оцененных характеристик - среднее число слов и предложений, минимальное и максимальное количество слов, а также общее и среднее число уникальных слов и лемм для обоих полей в наборе данных. Для определения извлечения лемм из исходного вида слов использовалась [pymorphy \(2015\)](#) - библиотека для морфологического анализа и инфлекссионных преобразований текстов на русском языке.

После расчета статистик набор данных был случайно разбит на тренировочную и тестовую части в пропорциях 9:1. Таким образом, размер тренировочной части составил 158785 объекта, размер тестовой - 17643 объектов. То есть, размер тренировочной части выборки более чем в 3 раза превысил один из известных русскоязычных корпусов для суммаризации текста - Gazeta.ru, представленного в статье [Gazeta.ru \(2020\)](#). Кроме того, в оригинальной статье про набор данных Gazeta.ru автор приводит похожий набор рассчитанных статистик, значения которых пропорционально близки к полученным в дан-

Таблица 6.1: Статистики набора данных.

Метрика	Статья	Аннотация
Число объектов	176428	
Среднее число слов	394.3	30.5
Среднее число предложений	23.4	2.2
Минимальное число слов	31	7
Максимальное число слов	4295	170
Число уникальных слова	1490203	241770
Число уникальных лемм	570632	96818
Среднее число уникальных слов	243.1	27.3
Среднее число уникальных лемм	199.8	26.5

ной работе, за исключением основного отличия: исходные тексты статей в нашем наборе в среднем примерно в 2 раза короче, а тексты аннотаций - приблизительно в 1,5 раза, в то время как максимальная длина текстов может быть больше практически в 3 раза. Следовательно, в дальнейшей работе мы можем использовать данные из набора Gazeta.ru в качестве дополнения к имеющимся или рассматривать нейросетевые модели, уже обученные на этом датасете.

6.4 Примеры объектов

В таблицах 6.2 и 6.3 отображены примеры объектов пар статья-аннотация, содержащихся в наборе данных после выполненной предобработки и фильтрации. Для сохранения читаемости примеров, тексты в поле "Статья" были обрезаны таким образом, чтобы они имели возможность поместиться на одной странице.

Таблица 6.2: Объекты в наборе данных. Пример 1.

Статья	Аннотация
<p>В ходе встречи обсуждались причины и возможные последствия кризиса. При этом было замечено, что в настоящее время у России сложились относительные преимущества по многим финансовым показателям. Несоввершенство банковского сектора РФ в условиях кризиса явилось преимуществом в силу того, что был задет меньший круг отношений в сфере производства и финансов, чем за рубежом. Г-н Сухов рассказал о методах стабилизации, используемых надзорными органами в условиях сложившейся ситуации. При этом было отмечено, что меры направлены на оказание помощи в консолидации и финансовом оздоровлении банковского сектора. Это вызвано необходимостью защитить интересы кредиторов и сохранить их доверие к банковскому сектору. Также обсуждались решения о поддержке системно значимых банков, то есть банков с наибольшим сосредоточением кредиторов, и о сохранении рыночной дисциплины с целью роста ответственности со стороны менеджмента банков. Большой интерес вызвал вопрос о ломбардном кредитовании. Как отметил г-н Сухов, Банк России осуществляет практику проведения операций ломбардного кредитования на фиксированных...</p>	<p>10 ноября в рамках научного семинара кафедры банковского дела ГУ-ВШЭ состоялось выступление члена совета директоров, директора департамента лицензирования деятельности финансового оздоровления кредитных организаций Банка России Михаила Сухова тему "Политика ЦБ стабилизации банковского сектора".</p>

Таблица 6.3: Объекты в наборе данных. Пример 2.

Статья	Аннотация
<p>Всем известно, что Дмитрий Быков пишет политические фельетоны в стихотворной форме и называет эти произведения «письмами счастья». Они были опубликованы в журналах «Огонек» и «Собеседник» и издавались отдельными сборниками. Автор объясняет собственное название жанра тем, что "...любое стихотворение — лирическое или сатирическое — может с полным правом называться письмом счастья, потому что нет большей радости, чем называть вещи своими именами". В этом году вышли две книги Быкова: сборник стихов «Отчет» и стихотворные фельетоны «Новые письма счастья», написанные за последние два года. Ведущий семинара ординарный профессор ГУ-ВШЭ Александр Архангельский, открывая встречу, отметил, что, хотя сам Дмитрий — невысокого мнения о своих работах в жанре поэтической сатиры, по мнению ведущих российских литературоведов, «письма счастья» — явление уникальное, заслуживающее пристального внимания как читателей, так и специалистов. Перед тем, как приступить к чтению стихов, автор сказал, что определенная часть литературного сообщества...</p>	<p>16 июня в Культурном центре ГУ-ВШЭ прошла встреча, завершающая годовой цикл семинаров «Важнее, чем политика» Фонда «Либеральная миссия» и Высшей школы экономики. На этот раз гостем был известный российский журналист, писатель, телеведущий и поэт Дмитрий Быков.</p>

7 Обзор предшествующих работ

Традиционно, все подходы к решению задачи суммаризации текста делятся на 2 типа:

- Экстрактивный.
- Абстрактивный.

Экстрактивный подход к суммаризации текста подразумевает под собой составление аннотации путем отбора наиболее релевантных предложений текста. Абстрактивный подход заключается в использовании генеративных нейросетевых архитектур для генерации сжатого текста с сохранением основной мысли. Далее в работе будут рассмотрены оба этих подхода для решения задачи суммаризации русскоязычного текста.

7.1 Экстрактивный подход.

Данный подход может быть реализован как с применением методов глубинного обучения, так и без использования каких-либо нейросетевых решений, алгоритмически.

7.1.1 Алгоритмические методы

Одним из самых известных алгоритмов, применимых к задаче - алгоритм TextRank. Он был впервые представлен в работе [Mihalcea & Tarau \(2004\)](#) и по своей сути является реинтерпритацией алгоритма PageRank за авторством [Brin & Page \(1998\)](#). Этот графовый метод, использующийся для извлечения ключевых слов (если работает на уровне слов) и экстрактивного резюмирования текста (если работает на уровне предложений), состоит из следующих шагов:

- Первый шаг заключается в построении из поданного текста взвешенного неориентированного графа, где вершины представляют предложения

текста, а вес ребра между двумя вершинами отражает степень сходства двух предложений, соответствующими этим вершинам. Близость предложений может рассчитываться следующим образом:

$$Similarity_{i,j} = \frac{|w \in (i \cap j)|}{|w \in (i \cup j)|}, \quad (12)$$

где i, j ($i \neq j$) - два множества лемм, относящихся к предложениям, соответствующим вершинам графа с индексами i и j .

Если же TextRank работает не с текстом, а с векторными представлениями предложений, близостью может быть, например, косинусная близость полученных эмбеддингов.

- Затем начинается итеративный процесс обновления весов вершин графа, отражающих меру центральности предложения, относящегося к данной вершине. На каждой итерации веса рассчитываются по данной формуле:

$$W_i = (1 - d) + d * \sum_{j \in Inc(i)} \frac{w_{j,i}}{\sum_{k \in Inc(j)} w_{j,k}} \cdot W_j, \quad (13)$$

где i и j ($i \neq j$) - индексы вершин графа, d - это коэффициент затухания, значение которого может быть установлено в диапазоне от 0 до 1 (в оригинальной статье авторы предлагают использовать значение 0.85), $Inc(i)$ - это множество вершин, смежных с вершиной с индексом i , $w_{i,j}$ - это вес ребра между вершинами с индексами i и j .

Важно отметить два неочевидных аспекта: во-первых, обычно исходные веса вершин инициализируются из равномерного распределения, во-вторых, итеративный процесс происходит либо фиксированное число шагов, (например, в алгоритме-прародителе PageRank обычно производится 20-30 итераций), либо до сходимости, то есть до тех пор, пока изменение весов не становится меньше определенного заранее порога (обычно используется пороговое значение в 10^{-4}).

- В конце предложения текста сортируются в порядке убывания весов соответствующих вершин графа, и затем заранее установленное число наиболее значимых предложений включаются в итоговую аннотацию.

Кроме оригинального TextRank существует и другие алгоритмы. Например, алгоритм LexRank, представленный в работе [Erkan et. al. \(2004\)](#), также основан на теории меры центральности и по своей сути является модифицированной версией TextRank. Алгоритмические методы просты в реализации и применении, но их качество в значительной степени ограничено комплексностью и многообразием устройства языка. В связи с этим, какая-то из таких техник может быть использована в качестве бейзлайна при тестировании других, более сильных методов.

7.1.2 Глубинное обучение в экстрактивной суммаризации

Существуют и методы экстрактивной суммаризации, в большей степени опирающиеся на технологии глубинного обучения, чем вышеперечисленные алгоритмические подходы. Например, в статье [Nallapati et. al. \(2017\)](#) авторы предлагают метод экстрактивной суммаризации SummaRuNNer, основанный на архитектуре рекуррентной нейронной сети RNN [[Jeffrey L. Elman \(1990\)](#)]. Эта модель использует иерархические RNN для ранжирования предложений в документе и присваивают более высокие баллы наиболее важным предложениям. Предложения с наивысшим рейтингом впоследствии включаются в аннотацию.

В статье Shashi Narayan [Narayan et. al. \(2018\)](#) авторы предлагают систему Refresh, в которой используется гибрид метода pointer-generator [[See et. al. \(2017\)](#)] и техники обучения "субъект-критик". В этой системе сеть "субъект" генерирует аннотацию к тексту и получает ответ от "критика", оценивающего качество полученных резюме. Система обучалась с помощью метода обучения с подкреплением (англ. reinforcement learning [[Sutton & Barto \(1998\)](#)]).

Одним из главных нововведений является использование метрики ROUGE в качестве функции вознаграждения во время обучения нейросети.

В статье [Zhang et. al. \(2018\)](#), авторы предлагают новый подход к экстрактивному суммаризации текста на основе модели последовательности скрытых переменных. Нововведение заключается в добавлении дискретной скрытой переменной к каждому предложению документа, определяющей, будет ли предложение включено в аннотацию. Такой подход подталкивает модель к улавливанию глобальных зависимостей между предложениями, что считалось сложной задачей для традиционных методов экстрактивной суммаризации. Это позволило авторам качественно превзойти предшествующие решения.

На данный момент существует и множество других методов экстрактивной суммаризации, показывающие достаточно хорошие результаты на соответствующих бенчмарках. Тем не менее, специфика описанного в разделе 6 набора данных не позволит этому подходу показывать хорошие результаты во время тестирования. Приведенные в таблицах 6.2, 6.3 примеры иллюстрируют ситуацию невозможности выбора нескольких наиболее релевантных предложений для формирования аннотации по причине необходимости реформулирования и синтеза новой информации, что существенно выходит за рамки экстрактивных методов. Более того, важные детали могут быть разбросаны по всему документу и не могут быть адекватно извлечены без учета контекста. В свете этих особенностей, по ходу работы было принято решение полностью отказаться от использования техник экстрактивной суммаризации в пользу абстрактивной.

7.2 Абстрактивный подход

Как было упомянуто выше, абстрактивный подход к задаче суммаризации текста заключается в использовании генеративных архитектур глубокого обучения для создания нового короткого текста в ответ на поданный длинный текст. В последние годы ключевым поворотом в развитии нейросетевых решений в сфере обработки естественного языка стало создание модели трансформера (англ. Transformer) [Vaswani et. al. (2017)]. На замену традиционным рекуррентным нейросетям, авторы статьи предложили концепцию механизма self-attention, который взвешивает значимость входных элементов по отношению к другим элементам последовательности. Во-первых, это дает модели возможность распараллеливать вычисления над элементами последовательности, что приводит к существенному повышению эффективности обучения. Во-вторых, данный механизм наделяет модель глобальным взглядом на входную последовательность, что не учитывалось в стандартной архитектуре RNN и полезно в задачах, требующих понимания далеких зависимостей. Таким образом, языковые модели на основе Transformer стали достигать SOTA-результатов в самых разных задачах NLP. Далее в работе будут рассмотрены архитектуры наиболее известных и производительных моделей глубокого обучения и их версии, оперирующие русским языком.

7.2.1 Наиболее производительные архитектуры

- **GPT** Основанная на трансформерах архитектура Generative Pretrained Transformer (GPT) была представлена в статье Radford et. al. (2018). Она является авторегрессионной (однаправленной) моделью и состоит из слоя эмбеддингов поданного текста, последовательных блоков декодеро-энкодера, слоя нормализации и линейного выходного слоя. Авторы производят предобучение GPT на различных задачах языкового моделирования, а затем дообучают ее на последующих генеративных задачах (например, ответы на вопросы, машинный перевод). На момент

публикации статьи модель GPT качественно превзошла все предыдущие решения на большинстве наборов данных под различные задачи обработки естественного языка.

Модель GPT-2 [Radford et. al. (2019)] является преемником GPT, имеет ту же архитектуру и предобучалась таким же образом, но на значительно большем объеме данных. Главным отличием от предыдущей версии является увеличение числа параметров модели более, чем в 12 раз, и более длительное предобучение. Эти два аспекта позволили GPT-2 генерировать значительно более связные тексты на длинных отрывках и демонстрировать более тонкое понимание контекста. В работе авторы подчеркивают важность размера модели и масштаба данных для достижения высоких результатов тестирования.

К моменту проведения данного исследования, были представлены также модели GPT-3 [Brown et. al. (2020)] и GPT-4 [OpenAI (2023)]. Однако, обе оригинальные модели не были предоставлены в открытый доступ, вследствие чего, они не будут рассмотрены в данном проекте.

• BART

Модель BART [Lewis et. al. (2020)] также основана на трансформерах. Но в отличие от GPT, в ее архитектуре присутствует как кодеровщик, так и декодеровщик. BART унаследовала от своего предшественника - модели BERT [Devlin et. al. (2019)] - двунаправленный энкодер, имеющий возможность считывать контекст как слева направо, так и в обратную сторону, что способствует более глубокому пониманию контекста длинных текстов. Важнейшей особенностью BART является использование расшумляющего автоэнкодера, обучающегося восстанавливать исходные данные из их зашумленной (полученной, например, путем перемешивания некоторых токенов или, аналогичного BERT, путем маскирования их части) версии. Расшумляющий автоэнкодер вынужден понимать и улавливать основные особенности распределения данных,

что делает его мощным инструментом для извлечения признаков и обучения представлений. В остальном архитектура BART стандартна для моделей, основанных на трансформерах, и одним из решающих факторов успеха модели является ее большой размер.

- **T5**

В статье [Raffel et. al. \(2020\)](#) авторы представляют Text-to-Text Transfer Transformer, или T5. Архитектура модели также состоит из кодировщика и декодероващика. Как и GPT, T5 является авторегрессионной моделью. Но ее предобучение больше похоже на предобучение BERT: она также генеративно обучалась именно на задачах восстановления замаскированных токенов, а не, например, перемешанных, как может быть у модели BART. Одной из отличительных черт T5 считается ее подход к определению задач обработки естественного языка: любая задача становится sequence-to-sequence задачей, то есть входные и выходные данные могут быть произвольными. Так, кроме стандартных постановок как, например, в машинном переводе, это используется и в задачах классификации: T5 преобразует метки классов в текстовые строки.

Классы, соответствующие каждой из описанных моделей, представлены в библиотеке transformers⁴.

7.2.2 Токенайзеры

Во время обучения, тестирования и использования любой из языковых моделей перед подачей входного текста необходимо получить его векторное представление. Каждая архитектура из раздела 7.2.1 имеет свой тип токенайзера, принимающего на вход текст и возвращающего его векторное представление. Так, в библиотеке transformers классу "GPT2LMHeadModel" относящегося к модели GPT-2 соответствует класс токенайзера "GPT2Tokenizer" использующего подход Byte Pair Encoding (BPE) [[Sennrich et. al. \(2015\)](#)], обуча-

⁴ <https://github.com/huggingface/transformers> (01.05.2023)

ющийся таким образом, чтобы итеративно объединять наиболее часто встречающиеся пары символов в один токен. Важной деталью подхода является то, что он позволяет кодировать слова, не входящие в словарь токенайзера, разбивая его на более короткие токены. Классы токенайзеров "BartTokenizer" и "T5Tokenizer", соответствующие моделям BART и T5, основаны на методе SentencePiece [Kudo & Richardson (2018)], объединяющего BPE с вероятностной моделью Unigram Language Model (ULM), использующейся для определения оптимального набора подслов на основе частоты их встречаемости в тексте. Важно, что разным моделям даже одной архитектуры соответствуют разные токенайзеры, и использование некорректного токенайзера ведет к полностью некорректной работе модели.

7.2.3 Критерии выбора предобученных моделей

Для реализации проекта университетом ВШЭ был предоставлен внутренний сервер, имеющий в распоряжении видеокарту с 6 гб памяти. Таким образом, безапелляционным критерием для избираемых архитектур является относительная легковесность модели, чтобы она имела возможность генерировать аннотации, используя предоставленный графический процессор. Также, в силу ограничения предоставленных вычислительных ресурсов, в рамках данного проекта будут проведены эксперименты лишь с нейросетями, предобученными под русский язык, каковыми могут являться мультязычные модели, или модели, обученные только на корпусе русского языка.

7.2.4 Мультязычные модели

Каждая из вышеописанных архитектур имеет многоязычный аналог. Так, в статье Shliazhko et. al. (2022) авторы воспроизвели архитектуру GPT-3 на основе GPT-2 для обучения на текстовом корпусе, содержащем 60 языков и включающем в себя статьи Википедии и корпус C4, представленный в статье Raffel et. al. (2020). Модель получила название mGPT.

В статье [Liu et. al. \(2020\)](#) авторы обучают модель mBART на мультиязычном корпусе из 25 языков CC25, который, в свою очередь, был извлечен из корпуса CC⁵. Существует две версии модели: оригинальный mBART и его расширенная на большее число языков версия mBART-50.

Авторы модели T5 создали и многоязычную вариацию: mT5 [[Xue et. al. \(2021\)](#)]. Эта модель имеет аналогичную архитектуру, отличающуюся лишь слоем активации, и обучалась таким же образом, как и T5, но на корпусе mC4, представленном в той же статье и содержащем тексты на 101 языке. Доступно 5 вариантов размера модели: mt5-small, mt5-base, mt5-large, mt5-xl, mt5-xxl.

На момент написания работы мультиязычной SOTA-моделью является mT0, предложенная в статье [Muennighoff et. al. \(2022\)](#). К сожалению, данная модель из-за своего размера нам не подходит, исходя из описанных в разделе [7.2.3](#) ограничений, а значит, не будет рассмотрена в дальнейшем.

7.2.5 Русскоязычные модели

Абсолютное большинство сильнейших русскоязычных версий моделей, описанных в разделе [7.2.1](#), были разработаны компанией SberDevices⁶. Рассмотрим эти модели в хронологическом порядке. Компания представила семейство моделей ruGPT3, основанных на архитектурах GPT-2. Каждая модель семейства обучалась без разметки на русскоязычном корпусе из 80 миллиардов токенов, содержащем статьи Википедии, книги, новости, русский Common Crawl и не только. В открытом доступе имеются 4 версии модели: ruGPT3Small⁷, ruGPT3Medium⁸, ruGPT3Large⁹, ruGPT3XL¹⁰. Первые 3 модели обучались

⁵ https://github.com/facebookresearch/cc_net (21.04.2023)

⁶ <https://sberdevices.ru> (01.05.2023)

⁷ https://huggingface.co/ai-forever/rugpt3small_based_on_gpt2 (01.05.2023)

⁸ https://huggingface.co/ai-forever/rugpt3medium_based_on_gpt2 (01.05.2023)

⁹ https://huggingface.co/ai-forever/rugpt3large_based_on_gpt2 (01.05.2023)

¹⁰ <https://huggingface.co/ai-forever/rugpt3xl> (01.05.2023)

напротяжении 3 эпох на контексте в 1024 токена, а потом 1 эпоху производился файн-тюнинг на увеличенном контексте в 2048 токенов. Для модели ruGPT3XL был разработан иной пайплайн предобучения, состоящий из 4 эпох на контексте в 512 токенов. Это является важной деталью для данной работы, означающей "готовность" моделей к обработке длинных текстов. Соответствующий моделям токенайзер является объектом класса "GPT2Tokenizer" и имеет размер словаря, равный 50258 токенов.

Русскоязычная модель ruT5 имеется в открытом доступе в 2 версиях: ruT5-base¹¹ и ruT5-large¹². Обе версии модели обучались на том же корпусе текстов, что и модели семейства ruGPT3. Размер словаря токенайзера, соответствующего моделям, составил 32101 токен.

На момент написания работы самой сильной русскоязычной моделью по бенчмарку RussianSuperGlue¹³ является FRED-T5, также обученной компанией SberDevices. Архитектура нейросети базируется на модели T5, но ее обучение в большей степени взято из статьи [Tay et. al. \(2022\)](#). Авторы предложили смесь декойзеров, определяющих способ обучения модели. Задачи повреждения спана (span corruption), на которых учили модель похожи на задачу маскированного языкового моделирования, используемую при обучении T5, но значительно более сложна для модели, так как ей, кроме восстановления последовательности, приходится определять и длину маскированного спана. Таким образом модель обучалась на том же наборе данных, что и ruT5. Существует две версии модели: FRED-T5-large и FRED-T5-1.7B. Сущностно, версии отличаются друг от друга числом голов механизма внимания, размерностью скрытого и полносвязного слоев и имеют похожие результаты на различных задачах бенчмарка. Авторы решили отказаться от SentencePiece подхода токенизации в пользу стандартного BPE, вследствие чего модель

¹¹ <https://huggingface.co/ai-forever/ruT5-base> (01.05.2023)

¹² <https://huggingface.co/ai-forever/ruT5-large> (01.05.2023)

¹³ <https://russiansuperglue.com/leaderboard> (15.04.2023)

использует токенайзер, аналогичный "GPT2Tokenizer", с 50364 токенами в словаре.

8 Тестирование моделей без дообучения

Итак, на данный момент были описаны мультязычные модели mGPT, mBART, mBART-50, mT5, и русскоязычные ruGPT3Small, ruGPT3Medium, ruGPT3Large, ruGPT3XL, ruT5-base, ruT5-large, FRED-T5-large и FRED-T5-1.7B. Исходя из требований, описанных в разделе 7.2.3, мы не можем использовать модели mBART-50, ruGPT3XL и FRED-T5-1.7B из-за их веса. Также по причине ограничения вычислительных ресурсов, мы не можем попробовать дообучить каждую из представленных моделей. Вследствие этого, было принято решение выбрать из оставшихся наиболее производительную модель и дообучить ее под задачу, а для остальных архитектур поступить следующим образом: найти их общедоступные версии, уже обученные под задачу суммаризации русского языка и провести тестирование на выделенной для этого выборке, а затем модели с наивысшими результатами дообучить на тренировочной части набора данных.

8.1 Бейзлайн

Итак, перед обучением и тестированием моделей глубинного обучения, необходимо установить бейзлайн, который будет считаться отправной точкой роста качества суммаризации. В силу простоты реализации и отсутствия необходимости в проведении обучения, бейзлайн-решением в работе будет выступать система экстрактивной суммаризации из алгоритма TextRank на TF-IDF признаках [[Sparck Jones \(1988\)](#)], полученных из предобученных 300-размерных векторных представлениях `naves`¹⁴. Предобученные векторные представления позволяют TF-IDF эффективнее учитывать семантику слов на основе их контекста, что плодотворно влияет на качество аннотаций.

¹⁴ <https://pypi.org/project/naves/> (12.03.2023)

8.2 mT5

Модель mT5, предобученную под задачу суммаризации текста в этой части исследования будут представлять модели mT5_multilingual_XLSum¹⁵ и mT5_m2o_russian_crossSum¹⁶. Первая была обучена на наборе данных под задачу суммаризации текста XLSum [Hasan et. al. (2022)], содержащего объекты пар "текст-резюме" на 44 языках, включая русский, общее количество объектов на котором составило 52712. В результате обучения были представлены следующие значения метрик, полученные на тестовой русскоязычной части набора данных: ROUGE-1 = 32.2; ROUGE-2 = 13.6; ROUGE-L = 26.2. Вторая модель обучалась под нестандартный вариант задачи суммаризации. Авторы модели используют собственный набор данных CrossSum [Bhattacharjee et. al. (2021)], также содержащий объекты пар "текст-резюме", но в данном случае тексты и резюме написаны на разных языках. При этом при обучении именно этой модели все резюме были написаны строго на русском языке. К сожалению, авторы не приводят точного значения метрик после тестирования модели.

8.3 Gazeta.ru: mBART, ruT5, ruGPT3

В статье Gusev (2020), кроме введения набора данных Gazeta.ru, автор производит файн-тюнинг модели mBART под задачу суммаризации. Модель доступна в общем доступе под названием mbart_ru_sum_gazeta¹⁷. Кроме модели mBART автор за рамками статьи также произвел на этом наборе данных файн-тюнинг предобученных моделей ruGPT3Small от SberDevices и уменьшенную версию ruT5-base¹⁸, полученную из модели mT5, благодаря снижению размерности скрытого слоя и уменьшению словаря с 250000 до 30000 токенов. Данные модели также находятся в открытом доступе и называются

¹⁵ https://huggingface.co/csebuatnlp/mT5_multilingual_XLSum (01.05.2023)

¹⁶ https://huggingface.co/csebuatnlp/mT5_m2o_russian_crossSum (01.05.2023)

¹⁷ https://huggingface.co/IlyaGusev/mbart_ru_sum_gazeta (01.05.2023)

¹⁸ <https://huggingface.co/cointegrated/rut5-base>

rugpt3medium_sum_gazeta¹⁹ и rut5_base_sum_gazeta²⁰. Автор представил результаты генераций каждой из моделей на тестовой части Gazeta.ru. Полученные метрики были перенесены в таблицу 8.1.

Таблица 8.1: Результаты тестирования моделей, обученных на Gazeta.ru

Модель	R-1	R-2	R-L	BLEU
mbart_ru_sum_gazeta	32.4	14.3	28,0	12,1
rugpt3medium_sum_gazeta	26.2	7.7	21.7	14.3
rut5_base_sum_gazeta	32.2	14.4	28.1	12.3

8.4 Уменьшенная ruT5

Автор вышеупомянутой уменьшенной версии ruT5-base [18] выложил еще 2 модели. Он произвел файн-тюнинг ruT5-base под множество задач, в том числе на различных специализированных выборках, в том числе под задачи перефразирования, заполнения пропусков, упрощения текста и прочие. Эта модель rut5-base-multitask²¹ стала основой для следующей модели автора: rut5-base-absum²², обучавшейся на 4 разных наборах данных под задачу суммаризации русскоязычного текста. К сожалению, автор не сопровождает модель какими-либо комментариями, а также не раскрывает используемые наборы данных и параметры обучения, а также какие-либо результаты генерации на тестовой выборке. Тем не менее, эта модель также будет протестирована.

8.5 Остальные модели

Другие модели из раздела 7.2, обученные под задачу суммаризации текста на русском языке не представлены в открытом доступе, в силу чего не будут протестированы в этой части исследования.

¹⁹ https://huggingface.co/IlyaGusev/rugpt3medium_sum_gazeta

²⁰ https://huggingface.co/IlyaGusev/rut5_base_sum_gazeta

²¹ <https://huggingface.co/cointegrated/rut5-base-multitask>

²² <https://huggingface.co/cointegrated/rut5-base-absum>

8.6 Результаты тестирования

Тестирование нейросетевых моделей производилось со следующими параметрами генерации: `repetition_penalty=2.0`, `num_beams=5`, `do_sample=False`, `min_length=20`, `max_length=150`. Результаты тестирования приведены в таблице 8.2.

Таблица 8.2: Результаты тестирования до файн-тюнинга

Модель	R-1	R-2	R-L	R-Lsum	BLEU
TextRank + TF-IDF (navec)	15.9	6.3	11.4	11.0	3.8
mT5_multilingual_XLSum	7.1	2.3	6.6	6.7	4.0
mT5_m2o_russian_crossSum	3.4	0.5	3.4	3.4	2.3
mbart_ru_sum_gazeta	13.5	5.7	13.1	13.2	12.8
rut5_base_sum_gazeta	13.7	5.7	13.4	13.5	11.9
rugpt3medium_sum_gazeta	10.1	7.5	9.7	9.8	6.8
rut5-base-absun	9.9	4.0	9.5	9.6	7.9

Как видно из таблицы, по метрике ROUGE-1 ни одна из моделей абстрактивной суммаризации не смогла превзойти бейзлайн. Тем не менее, по иным метрикам модели, предобученные на Gazeta.ru, показывают схожие друг с другом результаты, превзошедшие и бейзлайн-решение, и другие модели. Исключением является лишь `rugpt3medium_sum_gazeta`, достигшая сравнительно худшие результатов, однако, так как эта модель показала относительно неплохие метрики во время тестирования на датасете Gazeta.ru, данная просадка не драматична. Далее в работе из предобученных под задачу моделей будут рассмотрены только следующие: `mbart_ru_sum_gazeta`, `rut5_base_sum_gazeta`, `rugpt3medium_sum_gazeta`.

9 Файн-тюнинг избранных моделей

Итак, на данном этапе из подходящих по размеру моделей остались не опробованными следующие: `mGPT`, `ruGPT3Small`, `ruGPT3Large`, `ruT5-large` и `FRED-T5-large`. Говоря о нейросетях, наиболее корректным выбором в данной

ситуации было бы обучить каждую из этих моделей (кроме ruGPT3Small, по причине лучшей производительности ее больших аналогов ruGPT3Medium и ruGPT3Large на бенчмарке RussianSuperGlue) и определить лучшую эмпирическим путем. Тем не менее, из-за ограничения вычислительных мощностей, было принято решение провести только файн-тюнинг модели FRED-T5-large, так как ее рейтинг по RussianSuperGlue значительно выше остальных моделей, более того, как было описано выше, ее увеличенная версия - FRED-T5-1.7B - является SOTA-моделью для русского языка. Вследствие этого, FRED-T5-large после файн-тюнинга может стать самой производительной из имеющихся моделей.

Дообучение всех моделей на собственном наборе данных, введенном в разделе 6, проходило путем минимизации ошибки перекрестной энтропии на протяжении 2 эпох с использованием алгоритма оптимизации AdamW [Loshchilov & Hutter (2019)] и Cosine Annealing Scheduler со стратегией "warm restarts" [Loshchilov & Frank Hutter (2017)] с установленным значением параметра `num_cycles=0.5`. Такой подход плавно снижает скорость обучения до нуля в соответствии с изменением значения функции косинуса. Также были выставлены следующие параметры для AdamW: `LearningRate=10-4`, `WeightDecay=10-2`, остальные параметры были установлены значениями по умолчанию.

BatchSize для обучения выставлялся таким образом, чтобы стараться использовать всю свободную память графического процессора, а параметр, отвечающий за число шагов разминки (то есть число первых батчей, на которых скорость обучения будет расти с нуля до установленного значения) для Cosine Annealing Scheduler - так, чтобы общее число объектов, пропущенных через модель до окончания разминки, было равно 2000. Таким образом, модель `rut5_base_sum_gazeta` обучалась с параметрами `BatchSize=8` и `num_warmup_steps=500`, модель `mbart_ru_sum_gazeta` - с `BatchSize=4` и `num_warmup_steps=1000`, а модели `rugpt3medium_sum_gazeta` и FRED-

T5-large - с BatchSize=2 и num_warmup_steps=2000. При обучении моделей rut5_base_sum_gazeta и mbart_ru_sum_gazeta использовался метод Mixed Precision [Micikevicius et. al. (2018)], способствующий снижению объема используемой памяти и времени обучения. Модели mbart_ru_sum_gazeta и FRED-T5-large обучались с использованием техники нормирования градиентов clipping, описанной в статье Pascanu et. al. (2013).

10 Результаты файн-тюнинга

10.1 Значения метрик качества

Таблица 10.1: Результаты тестирования после файн-тюнинга

Model	R-1	R-2	R-L	R-Lsum	BLEU
TextRank + TF-IDF (navec)	15.9	6.3	11.4	11.0	3.8
mbart_ru_sum_gazeta	29.7	17.2	29.3	29.4	26.1
rut5_base_sum_gazeta	30.4	19.3	30.1	30.2	29.6
rugpt3medium_sum_gazeta	14.2	12.6	13.7	14.1	6.9
FRED-T5-large	29.7	17.9	29.5	29.5	9.4

Итак, по итогам обучения было проведено тестирование моделей на выделенной части выборки. Значения полученных метрик расположены в таблице 10.1. Лучших результатов достигла модель rut5_base_sum_gazeta, обучавшаяся под задачу суммаризации и на Gazeta.ru, и на нашем наборе данных. Также сравнимые с ней значения демонстрируют mbart_ru_sum_gazeta и FRED-T5-large. Модель rugpt3medium_sum_gazeta не смогла показать сравнимые результаты. Так как значения метрик не могут в полной мере описать валидность генераций, а качество сервиса напрямую зависит от связности и читаемости текста, далее в работе будут рассмотрены примеры генерации трех моделей с наилучшими метриками в целях визуального определения производительности каждой из них.

10.2 Примеры генераций

Для визуального тестирования были выбраны статьи портала ВШЭ, вышедшие после забора данных и не попавшие в располагаемый набор данных. В таблицах [10.2](#), [10.3](#) и [10.4](#), располагаются ссылки на оригинальные статьи, фрагменты их начала, оригинальные аннотации, а также аннотации, сгенерированных моделями `rut5_base_sum_gazeta`, `mbart_ru_sum_gazeta` и `FRED-T5-large`. Стоит отметить, что аннотации были получены с первого раза, без подбора параметров генерации и прочих искусственных модификаций.

Таблица 10.2: Пример генерации моделей 1

Ссылка
https://www.hse.ru/news/life/834095118.html
Фрагмент статьи
Семинар состоялся при поддержке Министерства образования и науки Республики Дагестан, Северо-Кавказского межрегионального управления Росприроднадзора по Республике Дагестан и Дагестанского государственного университета. Кинаят Юнусова, сотрудница Министерства образования и науки Республики Дагестан, отметила, что приоритетной деятельностью ее коллег является создание условий для всестороннего развития и обучения детей, а преференции для абитуриентов при поступлении в вузы мотивируют старшеклассников на участие ...
Оригинальная аннотация
В начале мая Институт экологии НИУ ВШЭ и Международная детско-юношеская премия «Экология — дело каждого» провели в Дагестане совместный семинар, где обсудили подготовку молодежных экологических проектов для федеральных и международных конкурсов. На встрече эксперты института представили методики организации проектной деятельности в сфере экологии и устойчивого развития для педагогов и молодежи Дагестана. Участие в мероприятии приняли преподаватели и учащиеся более 50 школ, колледжей и вузов республики.
rut5_base_sum_gazeta
В НИУ ВШЭ состоялся семинар Международной детско-юношеской премии "Экология - дело каждого" в котором приняли участие представители федеральных и региональных партнеров.
mbart_ru_sum_gazeta
Ирина Резанова, заместитель директора Института экологии, руководитель Ассоциации экошкол "Лидеры устойчивого развития" рассказала об опыте ЭкоШколы НИУ ВШЭ.
FRED-T5-large
Семинар состоялся при поддержке Министерства образования и науки Республики Дагестан. На семинаре обсуждались вопросы взаимодействия образовательных организаций НИУ ВШЭ и Республики Дагестан для реализации проектов в области экологии и устойчивого развития. В качестве спикеров выступили сотрудники Института экологии Ирина Резанова, Наталья Поротникова, Гурам Султанов, Кинаят Юнусова, Наталья Коротаева, Наталья Шакина. В мероприятии также приняли участие эксперты Международной детско-юношеской премии "Экология - дело каждого" Светлана Радионова, Наталья Рощина, Наталья Юсуфова.

Таблица 10.3: Пример генерации моделей 2

Ссылка
https://www.hse.ru/news/life/833947535.html
Фрагмент статьи
«Для нас сегодняшняя встреча — действительно событие знаковое. Дмитрий Артюхов возглавляет комиссию по молодежной политике. Для него общение с молодежью — это возможность понимать еще лучше запросы молодых людей. ВШЭ — это 50 тысяч студентов, средний балл которых при поступлении — 96. Это лучшие студенты нашей страны. То, что сегодня наши студенты приглашены на Ямал, — для нас большая честь и большая радость. В прошлом году была организована одна поездка на Ямал, в этом году будет две. Это свидетельство того, что наши отношения...
Оригинальная аннотация
В стенах Высшей школы экономики состоялась проектная сессия «Большие вызовы Ямала». Ее спикерами стали губернатор Ямало-Ненецкого автономного округа Дмитрий Артюхов, ректор НИУ ВШЭ Никита Анисимов и проректор НИУ ВШЭ Дмитрий Земцов. Студенты Вышки спроектировали концепции экспедиций в регион, в которые они смогут отправиться в 2023–2024 годах в рамках проекта «Открываем Россию заново».
rut5_base_sum_gazeta
В рамках проекта студенческих экспедиций «Открываем Россию заново» состоялась встреча студентов НИУ ВШЭ с губернатором ЯНАО Дмитрием Артюховым.
mbart_ru_sum_gazeta
Губернатор Ямало-Ненецкого автономного округа (ЯНАО) Дмитрий Артюхов встретился студентами НИУ ВШЭ, которые планируют поехать в экспедицию Ямал.
FRED-T5-large
В рамках проекта студенческих экспедиций «Открываем Россию заново» состоялась встреча студентов НИУ ВШЭ с губернатором Ямало-Ненецкого автономного округа Дмитрием Артюховым. В ходе встречи студенты представили свои идеи экспедиций, которые будут реализованы в ЯНАО в 2023 году, а также презентовали регион будущим участникам экспедиций. Студенты представили свои предложения концепций экспедиций губернатору, который ответил на несколько вопросов. В конце встречи Дмитрий Артюхов вручил студентам сертификаты участия в проекте. В рамках экспедиции студенты смогут посетить города Салехард, Муравленко, Лабытнанги, Новый Уренгой, Ноябрьск. В течение двух месяцев студенты будут проектировать концепции экспедиций регионам, которые войдут в состав экспедиции.

Таблица 10.4: Пример генерации моделей 3

Ссылка
https://www.hse.ru/news/edu/833915250.html
Фрагмент статьи
Что такое анализ данных в девелопменте сегодня Девелопмент — та область, в которой качественный скачок благодаря использованию data-driven-методов принятия решений еще впереди. Строительство входит в перечень ключевых отраслей экономики РФ, в которых запланировано масштабное внедрение искусственного интеллекта. В девелопменте...
Оригинальная аннотация
Факультет компьютерных наук НИУ ВШЭ и одна из крупнейших российских девелоперских компаний «Самолет» открывают новую магистерскую программу «Анализ данных в девелопменте». «Самолет» профинансирует обучение 30 лучших абитуриентов, поступивших на программу.
rut5_base_sum_gazeta
Магистерская программа "Анализ данных в девелопменте" НИУ ВШЭ совместно компанией "Самолет" запускает новый образовательный проект
mbart_ru_sum_gazeta
НИУ ВШЭ и «Самолет», лидеры в своих областях, решили объединить усилия в рамках образовательного проекта. Уже сегодня компания создает решения на основе аналитики данных для планировок и проектов на сайте, отделки и дизайна, прогнозов намерений клиентов приобрести продукт, оценки NPS пользователей, стратегии коммуникации с клиентами с задолженностями по ЖКХ и т.д.
FRED-T5-large
Магистерская программа факультета компьютерных наук НИУ ВШЭ "Анализ данных в девелопменте" создана в партнерстве с компанией "Самолет одним из крупнейших девелоперов в России. На программе будут готовить специалистов в области анализа больших данных, которые смогут работать самых разных сферах бизнеса: строительстве, логистике, управлении логистической инфраструктурой, управлении цепями поставок, маркетинге, консалтинге, IT-разработке. Обучение программе бесплатное. Финансирование со стороны компании покроет 100% стоимости обучения 30 лучших абитуриентов, поступивших на программу. Остальные места на программе будут платными. Прием на программу будет вестись по результатам конкурса портфолио, собеседованиям, в оценке которых примут участие эксперты факультета компьютерной науки НИУ Высшая школа экономики.

10.3 Выводы и выбор моделей

Из приведенных выше таблиц видно, что модели `rut5_base_sum_gazeta` и `mbart_ru_sum_gazeta` в среднем пытаются генерировать тексты меньшей длины, чем `FRED-T5-large`, и их тексты больше похожи на заголовки. Их аннотации чаще и в большей степени выглядят экстрактивными. Это не всегда является проблемой, так как некоторые аннотации на новостном портале действительно состоят из первых предложений статьи. Данные модели чаще путаются в понятиях и действующих лицах. Тем не менее, `FRED-T5-large` тоже не лишен данных проблем: например, в таблице 10.4 одно из предложений является копией предложения текста (не вошедшего в представленный фрагмент, но содержащей в оригинальной статье, доступной по ссылке). Также в той же таблице данная модель оставила предложение, и даже слово недописанным при условии отсутствия ограничения максимального числа генерируемых токенов. В целом зачастую такая проблема может встречаться и в первых двух моделях.

Тем не менее, каждая из моделей по своему хорошо генерирует аннотации, которые хоть и не похожи на оригинальные, но содержат мало как фактических, так и логических ошибок, легко читаемы и довольно точно передают суть оригинальной статьи. Таким образом, нельзя сделать однозначный вывод о превосходстве одной модели над другими, вследствие чего, все три модели будут добавлены в итоговую аннотирующую систему.

11 Имплементация облачного сервиса

В рамках данной части проекта были изучены и применены как методы создания пользовательского интерфейса, так и технологии, отвечающие за обработку на данных стороне сервера, для проведения аннотирования текста в облачном сервисе. Далее будет приведен обзор используемых при имплементации сервиса методов.

11.1 Обработка данных на стороне сервера

В рамках реализации данной составляющей сервиса был использован веб-фреймворк FastAPI²³, использующий новый стандарт асинхронности Python ASGI²⁴, вместо традиционного синхронного WSGI²⁵, что обеспечивает высокую производительность и лучшее использование ресурсов при обработке запросов.

В программе, отвечающей за серверную сторону системы, создается экземпляр приложения FastAPI, представляющий собой основную точку входа HTTP-запросов для обработки. Далее полученный декоратор использует метод "post" для отправки данных на сервер по пути "/summarize/", к которому обращается клиент для выполнения операции аннотирования. По этому пути данные поступают на вход функции "summarize()", принимающей в качестве аргумента экземпляр класса "Item", унаследовавшегося от класса "BaseModel" из библиотеки pydantic²⁶, определенный следующим образом:

```
1 class Item(BaseModel):
2     text: str
3     model_name: str
4     min_length: int
5     max_length: int
6     length_penalty: float
7     no_repeat_ngram_size: int
8     repetition_penalty: float
9     top_k: int
10    top_p: float
11    num_beams: int
12    temperature: float
```

Данный класс служит для валидации входных данных, приходящих в

²³ <https://fastapi.tiangolo.com/> (01.05.2023)

²⁴ <https://github.com/django/asgiref> (01.05.2023)

²⁵ <https://wsgi.readthedocs.io/en/latest/> (01.05.2023)

²⁶ <https://docs.pydantic.dev/latest/> (01.05.2023)

формате JSON в теле HTTP-запроса. FastAPI автоматически преобразует входные данные в экземпляр данного класса и проверяет соответствие типов данных и требований к ним. В полях класса объекта указаны параметры метода "generate"²⁷: объект входного текста статьи "text" (строка); название модели "model_name" (строка); минимальное и максимальное количество токенов на генерацию "min_length" и "max_length" (целые числа); штраф за длину генерации "length_penalty" (число с плавающей точкой); длина минимальной неповторяемой последовательности символов "no_repeat_ngram_size" (целое число); штраф модели за повторения "repetition_penalty" (число с плавающей точкой); ограничение числа наиболее вероятных кандидатов при генерации следующего токена: "top_k" (целое число), указывает на число кандидатов, а "top_p" (число с плавающей точкой) отсекает долю наименее вероятных вариантов, пользуясь статистической вероятностью; количество "сценариев рассматриваемых моделью при аннотировании "num_beams" (целое число); параметр контроля разнообразия генерации "temperature" (число с плавающей точкой).

Функция "summarize()" выполняет суммаризацию входного текста с помощью выбранной модели, используя переданные параметры. Код генерации написан с использованием библиотек transformers и pytorch²⁸, поддерживающих произведение вычислений на графическом процессоре, что существенно ускорит процесс суммаризации. Результат суммаризации возвращается в формате JSON в теле HTTP-ответа. Важно отметить, что FastAPI обеспечивает эффективное использование ресурсов за счет асинхронной обработки запросов, что позволяет обрабатывать множество запросов одновременно.

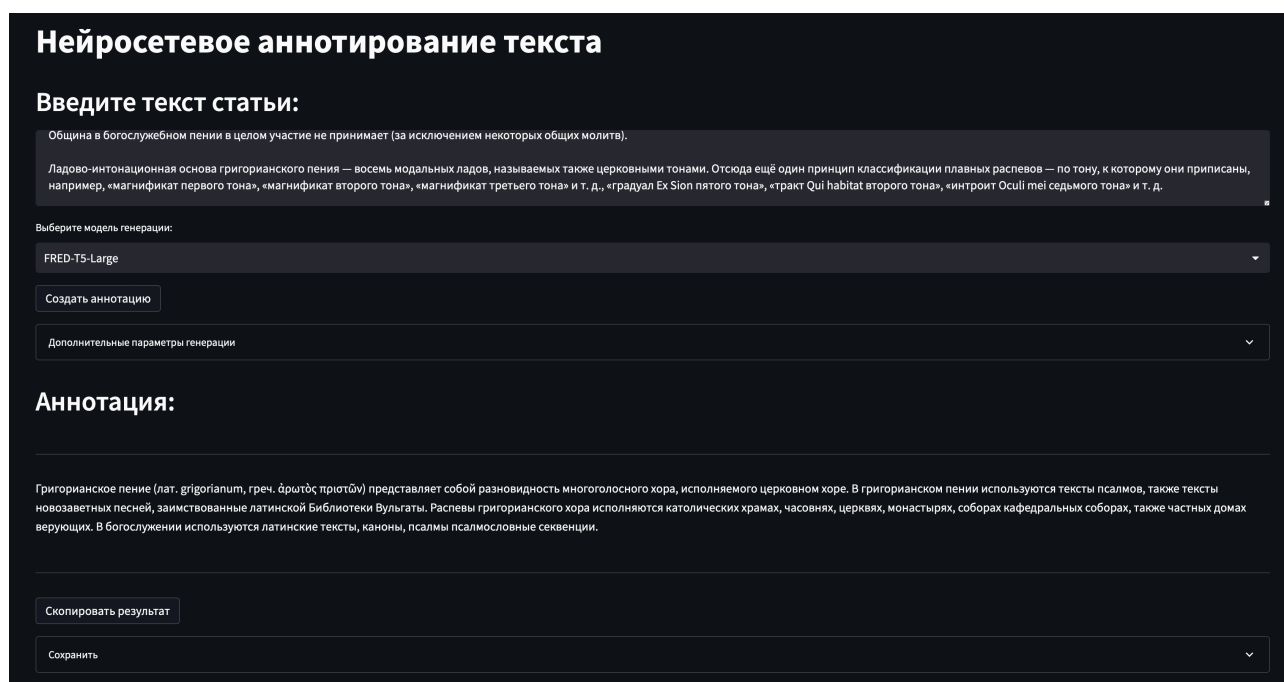
Описанный скрипт будет внедрен на сервер аналитики ВШЭ, имеющий в распоряжении графический процессор "NVIDIA GeForce GTX 1660".

²⁷ https://huggingface.co/docs/transformers/main/en/main_classes/text_generation (07.04.2023)

²⁸ <https://pytorch.org/docs/stable/> (01.05.2023)

11.2 Пользовательский интерфейс

В этом компоненте сервиса было реализовано интерактивное веб-приложение, имплементированное с использованием библиотеки Streamlit²⁹, предоставляющей широкий спектр виджетов для интерактивного ввода данных и обновления интерфейса в реальном времени. Разработанная страница содержит поле для ввода аннотируемого текста, поле вывода сгенерированной аннотации с возможностью копирования или сохранения полученного результата в формате текстового файла (см. рис. 11.1).



Нейросетевое аннотирование текста

Введите текст статьи:

Община в богослужебном пении в целом участие не принимает (за исключением некоторых общих молитв).

Ладово-интонационная основа григорианского пения — восемь модальных ладов, называемых также церковными тонами. Отсюда ещё один принцип классификации плавных распевов — по тону, к которому они приписаны, например, «магнификат первого тона», «магнификат второго тона», «магнификат третьего тона» и т. д., «градуал Ex Sion пятого тона», «тракт Qui habitat второго тона», «интроит Oculi mei седьмого тона» и т. д.

Выберите модель генерации:

FRED-T5-Large

Создать аннотацию

Дополнительные параметры генерации

Аннотация:

Григорианское пение (лат. gr̄igorīanum, греч. ᾠρὴ τοῦ πρίστου) представляет собой разновидность многоголосного хора, исполняемого церковным хором. В григорианском пении используются тексты псалмов, также тексты новозаветных песней, заимствованные латинской Библиотекы Вульгаты. Распевы григорианского хора исполняются в католических храмах, часовнях, церквях, монастырях, соборах кафедральных соборов, также частных домах верующих. В богослужении используются латинские тексты, каноны, псалмы псалмословные секвенции.

Скопировать результат

Сохранить

Рис. 11.1: Пользовательский интерфейс. Поля ввода текста и вывода аннотации.

Также была разработана вкладка настройки параметров генерации, описанных в разделе 11.1. Для их создания использовались специальные виджеты, поддерживающие функции ввода, вывода данных, демонстрации интерфейса: "slider", "number_input", "columns", "expander" и прочие. При изменении значений или состояний этих виджетов Streamlit автоматически перезапускает скрипт и обновляет интерфейс. Визуализация данной вкладки изображена на рисунке 11.2.

После ввода интересующего текста и выбора желаемых параметров гене-

²⁹ <https://docs.streamlit.io/> (01.05.2023)

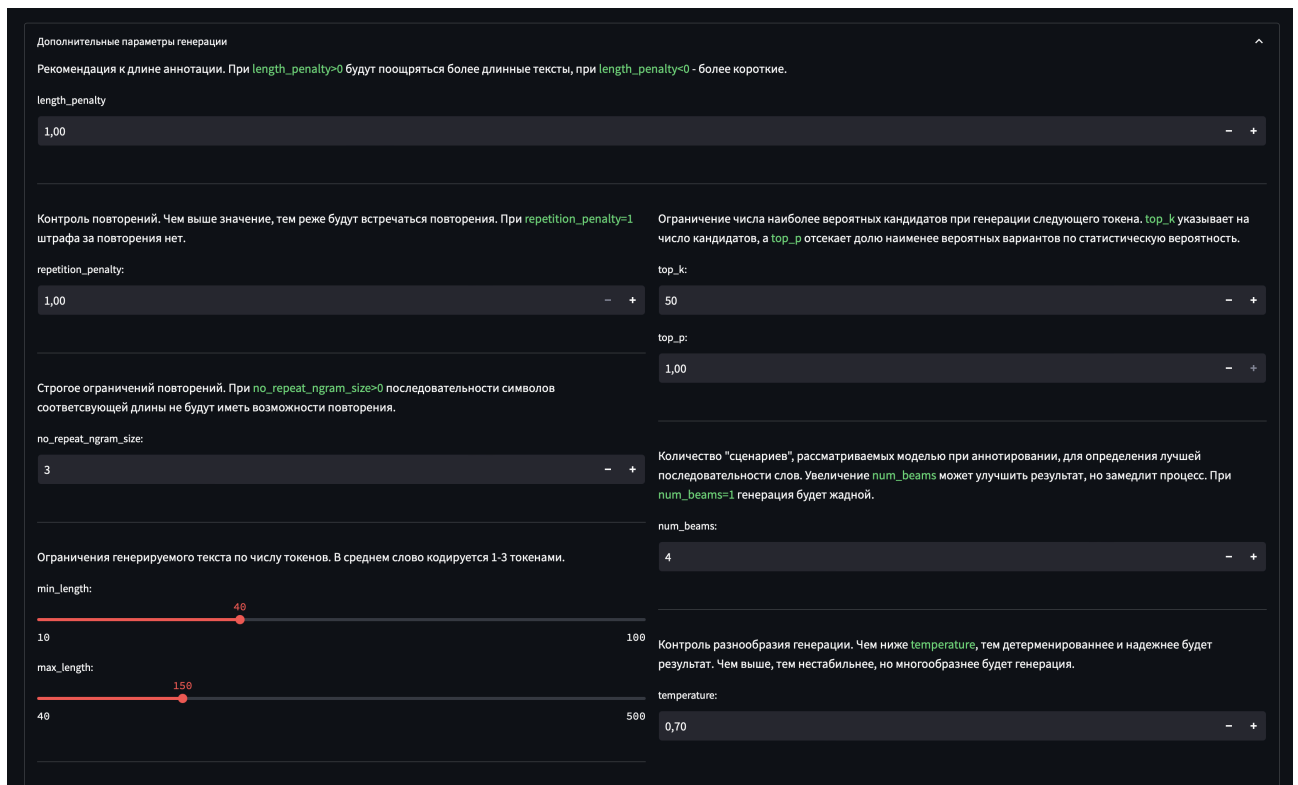


Рис. 11.2: Пользовательский интерфейс. Вкладка настройки параметров.

рации пользователю предлагается сгенерировать аннотацию, нажав соответствующую кнопку интерфейса. При нажатии на кнопку вызывается функция, которая оборачивает все значения введенных параметров, включая поданный текст и имя нейросетевой модели, в объект `json` и создает HTTP-запрос к серверу с помощью метода `"post()"` библиотеки `requests`³⁰.

После принятия ответа от сервера программа выводит текст сгенерированной аннотации в соответствующее текстовое поле веб-интерфейса. Важно упомянуть, что Streamlit по умолчанию не поддерживает сохранение состояния веб-приложения между различными запусками скрипта. Здесь вступает в действие концепция `"session_state"`, используемая для сохранения состояния интерфейса и результатов генерации между запусками, что позволяет пользователю возвращаться к предыдущим результатам в разделе "Последние генерации", где может визуализироваться до 10 последних аннотаций, а также параметры их производства, как показано на рисунке 11.3.

³⁰ <https://requests.readthedocs.io/en/latest/> (12.05.2023)

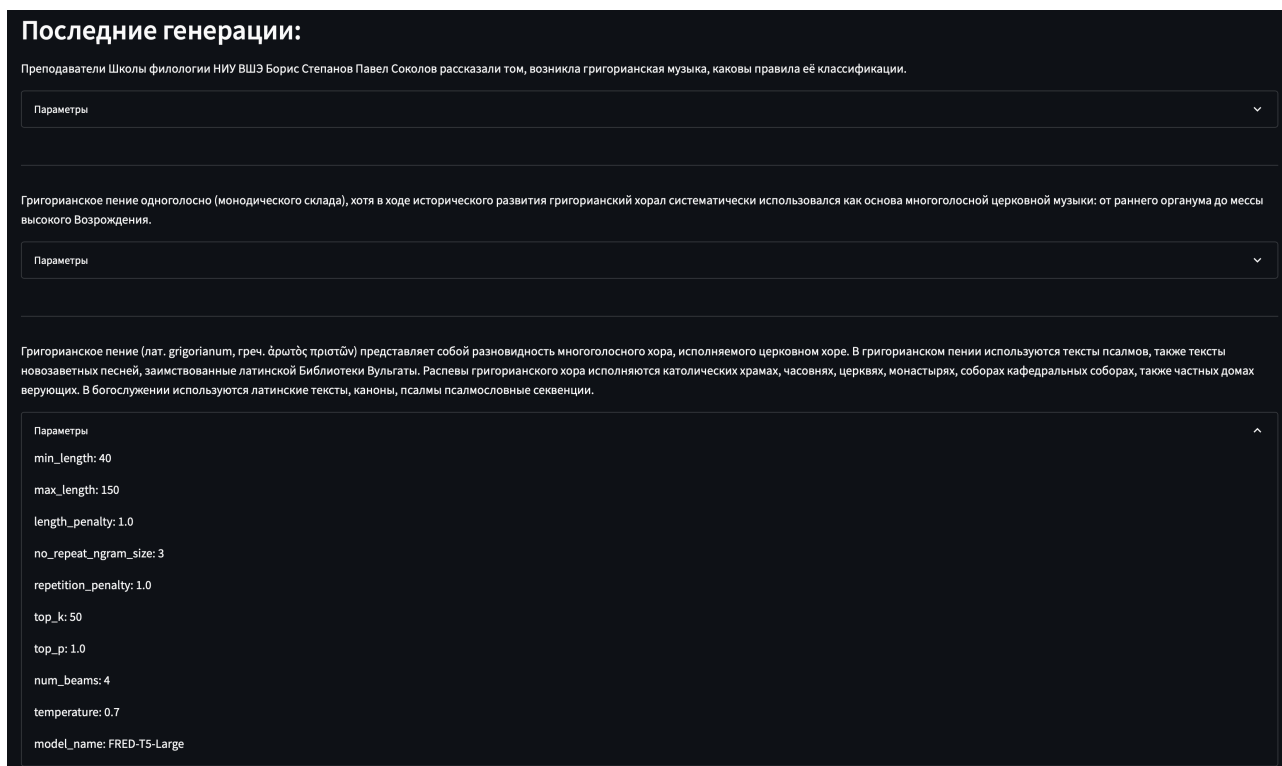


Рис. 11.3: Пользовательский интерфейс. Раздел "Последние генерации".

12 Имплементация автоматического дообучения моделей

В рамках проекта были реализованы два скрипта. Первый содержит программу обработки JSON-файла с данными, располагающимися на сайте новостного портала ВШЭ: он конвертирует его в CSV-файл обучающего набора данных, реализуя процессы, описанные в разделах 6.1 и 6.2. Второй скрипт реализует тестирования и обучения выбранных моделей с нуля или дообучения с сохраненного состояния. Данные скрипты позволят улучшать результаты работы в полуавтоматическом режиме даже человеку, поверхностно ознакомленному с концепциями глубинного обучения или программирования в целом. Далее будут разобраны параметры и функции второго из описанных скриптов.

Скрипт дообучения имеет всего четыре обязательных к указанию параметра:

- "--action" - название осуществляемого действия (обучение или тестирование).
- "--csv_dataset_path" - путь до файла с набором данных.
- "--model_type" - тип модели (ruT5, ruGPT, mBART или FRED-T5).
- "--HF_model_name" - имя прототипа модели в библиотеке transformers.

Остальные параметры являются дополнительными и имеют значения по умолчанию. Такие параметры, в частности, отвечают за пути к различным файлам, непосредственно влияющим на обучение или результаты, LearningRate и WeightDecay алгоритма оптимизации AdamW, значения num_warmup_steps и num_cycles для Cosine Annealing Scheduler, флаги использования техник mixed precision и clipping, опции взаимодействия с инструментом визуализации прогресса и результатов обучения "Weights & Biases" через библиотеку wandb³¹ и прочие. Важной деталью является тот факт, что скрипт может работать при указании значений любого подмножества параметров, кроме обязательных. В этом случае параметрам будут выставлены значения по умолчанию, соответствующие указанным в разделе 9. Полный список параметров, их названий, описаний и значений по умолчанию можно получить, запустив скрипт с флагом "-h". Скрипты не имеют пользовательского интерфейса в привычном смысле и могут быть запущены с помощью команды на языке Bash.

13 План дальнейших работ

Для дальнейшего расширения и улучшения работы предлагается реализовать следующий план. Во-первых, при согласовании с администрацией новостного портала ВШЭ будет возможно имплементировать модели и код, отвечающий за обработку данных и генерацию аннотаций на стороне сервера, в

³¹ <https://docs.wandb.ai/> (01.05.2023)

кабинет редактора новостного портала. Во-вторых, при наличии более объемных и производительных GPU могут быть проведены эксперименты с моделями большего размера и, в частности, с моделью FRED-T5-1.7B. В ином случае, одним из путей развития может быть дообучения имеющихся моделей на тех же или обновленных данных с помощью созданных скриптов или проведение экспериментов с иными неопробованными моделями (mGPT, ruT5-large, ruGPT3Large). В-третьих, с целью обеспечения возможности масштабирования проекта могут быть добавлены дополнительные функции пользовательского интерфейса для облачного сервиса. Например, внедрение функционала регистрации, личного кабинета, добавление базы данных генераций и тому подобное.

14 Заключение

В результате выполненной работы были проведены анализы метрик качества сгенерированных текстов, а также различных подходов и систем суммаризации текста. Был введен, очищен и исследован эксклюзивный набор данных из статей новостной платформы ВШЭ. Затем были проведены анализ и тестирование моделей глубинного обучения для обработки естественного языка, предобученных на различных русскоязычных текстовых корпусах. По результатам исследования наиболее производительные из них были выделены и обучены на приведенном наборе данных. Лучшие из обученных моделей стали частью разработанного облачного сервиса для аннотирования русскоязычных текстов. В целях лучшего пользовательского опыта в качестве дополнительного функционала были созданы скрипты предобработки данных и автоматического дообучения имеющихся нейросетевых моделей. Весь код проекта, а также полная инструкция к запуску описанных скриптов доступен по ссылке: https://github.com/paulkulpin/textsum_HSE_news/.

СПИСОК ИСТОЧНИКОВ

1. Karen Sparck Jones and Mark Sanderson. Automatic summarizing: Factors and directions. *In Cambridge MA: MIT Press, pp 1-12.* 1999.
2. Kishore Papineni, Salim Roukos, Todd Ward, Wei-Jing Zhu. BLEU: a Method for Automatic Evaluation of Machine Translation. *In Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics, ACL, pp 311–318.* 2002.
3. Chin-Yew Lin, Rouge: A package for automatic evaluation of summaries. *In Text Summarization Branches Out, ACL, pp 74–81.* 2004.
4. Korobov Mikhail. Morphological Analyzer and Generator for Russian and Ukrainian Languages. *In Analysis of Images, Social Networks and Texts, pp 320-332.* 2015.
5. Ilya Gusev. Dataset for Automatic Summarization of Russian News. *In Communications in Computer and Information Science, vol 1292.* 2020.
6. Rada Mihalcea and Paul Tarau. TextRank: Bringing Order into Texts. *In Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing, pp 404-411.* 2004.
7. Sergey Brin and Lawrence Page. The PageRank Citation Ranking: Bringing Order to the Web. *Stanford InfoLab.* 1998.
8. Gunes Erkan, Dragomir R. Radev. LexRank: Graph-based Lexical Centrality as Salience in Text Summarization. *In Journal Of Artificial Intelligence Research, vol 22, pp 457-479.* 2004.
9. Ramesh Nallapati, Feifei Zhai, Bowen Zhou. Summarunner: A recurrent neural network based sequence model for extractive summarization of documents. *In Proceedings of the AAAI Conference on Artificial Intelligence.* 2017.

10. Jeffrey L. Elman. Finding Structure in Time. *In University of California, Cognitive Science*, 14(2), pp 179-211. 1990.
11. Shashi Narayan, Shay B. Cohen, Mirella Lapata. Ranking Sentences for Extractive Summarization with Reinforcement Learning. *In Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, vol. 1 (Long Papers)*, pp 1747–1759. 2018.
12. Abigail See, Peter J. Liu, Christopher D. Get To The Point: Summarization with Pointer-Generator Networks. *In Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics*. 2017.
13. Richard S. Sutton and Andrew G. Barto. Reinforcement learning: An introduction. In MIT press, 1998.
14. Xingxing Zhang, Mirella Lapata, Furu Wei, Ming Zhou. Neural Latent Extractive Document Summarization. *In Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pp 779–784. 2018.
15. Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, Illia Polosukhin. Attention is all you need. *Advances in Neural Information Processing Systems*, vol. 30, pp 5998-6008. 2017.
16. Alec Radford, Karthik Narasimhan, Tim Salimans, Ilya Sutskever. Improving Language Understanding by Generative Pre-Training. 2018.
17. Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever. Language Models are Unsupervised Multitask Learners. *OpenAI Blog*, vol. 1, no. 8. 2019.
18. Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, ... and Dario Amodei. Language Models are Few-Shot Learners. *arXiv preprint arXiv:2005.14165* 2019.

19. OpenAI. GPT-4 Technical Report. 2023.
20. Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Ves Stoyanov, Luke Zettlemoyer. BART: Denoising Sequence-to-Sequence Pre-training for Natural Language Generation, Translation, and Comprehension. *In Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pp 7871–7880. 2020.
21. Jacob Devlin, Ming-Wei Chang, Kenton Lee, Kristina Toutanova. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. *In Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, vol. 1 (Long and Short Papers)*, pp 4171–4186. 2019.
22. Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, Peter J. Liu. Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer. *In The Journal of Machine Learning Research*, vol. 21, no. 1, pp 5485–5551. 2020.
23. Rico Sennrich, Barry Haddow, Alexandra Birch. Neural Machine Translation of Rare Words with Subword Units. *In Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics, vol. 1 (Long Papers)*. 2015.
24. Taku Kudo, John Richardson. SentencePiece: A simple and language independent subword tokenizer and detokenizer for Neural Text Processing. *In Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pp 66–71. 2018.
25. Oleh Shliakhko, Alena Fenogenova, Maria Tikhonova, Vladislav Mikhailov, Anastasia Kozlova, Tatiana Shavrina. mGPT: Few-Shot Learners Go Multilingual. *arXiv preprint arXiv:2204.07580*. 2022.
26. Yinhan Liu, Jiatao Gu, Naman Goyal, Xian Li, Sergey Edunov, Marjan Ghazvininejad, Mike Lewis, Luke Zettlemoyer. Multilingual Denoising Pre-

- training for Neural Machine Translation. *In Transactions of the Association for Computational Linguistics, vol. 8, pp 726–742.* 2020.
27. Linting Xue, Noah Constant, Adam Roberts, Mihir Kale, Rami Al-Rfou, Aditya Siddhant, Aditya Barua, Colin Raffel. mT5: A massively multilingual pre-trained text-to-text transformer. *In Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, ACL, pp 483–498.* 2021.
 28. Niklas Muennighoff, Thomas Wang, Lintang Sutawika, Adam Roberts, Stella Biderman, ... and Colin Raffel. Crosslingual Generalization through Multitask Finetuning. *arXiv preprint arXiv:2211.01786.* 2022.
 29. Yi Tay, Mostafa Dehghani, Vinh Q. Tran, Xavier Garcia, Jason Wei, ... and Donald Metzler. UL2: Unifying Language Learning Paradigms. *arXiv preprint arXiv:2205.05131.* 2022.
 30. Karen Sparck Jones. A Statistical Interpretation of Term Specificity and Its Application in Retrieval. *In Document Retrieval Systems, Peter Willett (Ed.), Taylor Graham Publishing, pp 132–142.* 1988.
 31. Tahmid Hasan, Abhik Bhattacharjee, Md. Saiful Islam, Kazi Mubasshir, Yuan-Fang Li, Yong-Bin Kang, M. Sohel Rahman, and Rifat Shahriyar. XL-Sum: Large-Scale Multilingual Abstractive Summarization for 44 Languages. *In Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021, pp 4693–4703.* 2021.
 32. Abhik Bhattacharjee, Tahmid Hasan, Wasi Uddin Ahmad, Yuan-Fang Li, Yong-Bin Kang, Rifat Shahriyar. CrossSum: Beyond English-Centric Cross-Lingual Abstractive Text Summarization for 1500+ Language Pairs. *arXiv preprint arXiv:2112.08804* 2021.
 33. Ilya Loshchilov, Frank Hutter. Decoupled Weight Decay Regularization. *In Proceedings of ICLR.* 2019.

34. Ilya Loshchilov, Frank Hutter. SGDR: Stochastic Gradient Descent with Warm Restarts. *In Proceedings of ICLR*. 2017.
35. Paulius Micikevicius, Sharan Narang, Jonah Alben, Gregory Diamos, Erich Elsen, David Garcia, Boris Ginsburg, Michael Houston, Oleksii Kuchaiev, Ganesh Venkatesh, Hao Wu. Mixed precision training. *In ICLR 2018 Conference*. 2018.
36. Razvan Pascanu, Tomas Mikolov, Yoshua Bengio. On the difficulty of training recurrent neural networks. *In Proceedings of the 30th international conference on machine learning (ICML-13)*, pp 1310-1318. 2013.