

МОСКОВСКИЙ ФИЗИКО-ТЕХНИЧЕСКИЙ ИНСТИТУТ  
(национальный исследовательский университет)  
ФИЗТЕХ-ШКОЛА ПРИКЛАДНОЙ МАТЕМАТИКИ И ИНФОРМАТИКИ  
КАФЕДРА ИНТЕЛЛЕКТУАЛЬНЫХ СИСТЕМ

Грицай Герман Михайлович

**ВЕРИФИКАЦИЯ ИСКУССТВЕННО СГЕНЕРИРОВАННЫХ  
ТЕКСТОВЫХ ФРАГМЕНТОВ**

09.04.01 — Информатика и вычислительная техника

Выпускная квалификационная работа магистра

**Научный руководитель:**  
к.ф.-м.н. А. В. Грабовой

Москва — 2025

## **Аннотация**

Исследуется проблема верификации искусственно сгенерированных текстовых последовательностей с целью повышения их интерпретируемости. Исследование охватывает различные постановки задач классификации, направленные на детекцию и интерпретацию машинно-сгенерированных фрагментов. Предлагаются методы, основанные на механизме самовнимания, а также включающие элементы статистического анализа. Также охватываются методы мультизадачного подхода обучения, который позволяет проводить регуляризацию признаковых представлений текстовых фрагментов и повышать обобщающую способность моделей. Эффективность предложенных теоретических результатов оценивается на синтетических и реальных наборах данных, включая наборы из международных соревнований по детекции сгенерированных текстов. Полученные результаты демонстрируют практическую значимость архитектурных решений для выявления искусственно сгенерированных слов, предложений и смысловых блоков документов в условиях многодоменных и многоязычных сценариев, способствуя развитию методов верификации текстовых данных.

*Ключевые слова:* выбор модели; механизм самовнимания; большие языковые модели; архитектура трансформер; многозадачное обучение; множественная проверка гипотез; оценка перплексии.

## Оглавление

	Стр.
Введение . . . . .	4
Глава 1. Детекция автора всего документа . . . . .	7
1.1. Постановка задачи детекции в терминах классификации . . . . .	8
1.2. Увеличение длины входного контекста . . . . .	8
1.3. Статистические языковые модели . . . . .	10
1.4. Анализ качества детекции текстовых последовательной на разных языках . . . . .	14
Глава 2. Детекция сгенерированных фрагментов внутри документа . . . . .	22
2.1. Постановка задачи детекции фрагментов фиксированной длины . . . . .	23
2.2. Суперпозиция преобразований для детекции фрагментов . . . . .	24
2.3. Детекция фрагментов варьируемой длины . . . . .	27
2.4. Анализ качества детекции фрагментов документов . . . . .	31
Глава 3. Агрегация и использование смеси моделей для детекции автора . . . . .	36
3.1. Доменная адаптация при помощи легковесных адаптеров . . . . .	37
3.2. Агрегация моделей для покрытия языкового разнообразия . . . . .	39
3.3. Анализ смеси моделей для детекции фрагментов . . . . .	40
Глава 4. Мультизадачное обучение для задачи детекции . . . . .	43
4.1. Постановка подхода мультизадачного обучения . . . . .	43
4.2. Мультизадачное обучение с архитектурой трансформер . . . . .	44
4.3. Анализ применения мультизадачности для задачи детекции фрагментов . . . . .	48
Заключение . . . . .	52
Список иллюстраций . . . . .	54
Список таблиц . . . . .	56
Список литературы . . . . .	57

## Введение

**Актуальность темы.** Развитие больших языковых моделей (Large Language Models, LLM) в области обработки естественного языка (Natural Language Processing, NLP) привело к появлению искусственно созданных текстов, которые с каждым днем становится все труднее отличить от написанных человеком [1, 2]. В последние годы сфера применения языковых моделей расширилась, что привело к появлению чат-ботов и ассистентов по написанию текстов, таких как ChatGPT [3], Llama [4], Mistral [5], GigaChat [6] и YaGPT [7]. Применение таких инструментов уже давно вышло за рамки тривиальной помощи с грамматикой и пунктуацией. Их применение оставляет след в различных отраслях, включая бизнес [8], журналистику [9], образование [10, 11] и программирование [12]. Данные системы являются важным инструментом, позволяющим людям решать задачи естественного языка, поскольку они способны предоставить ответ на любой корректно поставленный вопрос. Однако, несмотря на очевидные преимущества, у таких сервисов есть и обратная сторона — вредоносное использование. Среди вредоносных сторон генерируемых текстов — спам [13], плагиат [14, 15] и ложная информация, которые все чаще появляются в Интернете. Помимо этого, было обнаружено, что большие языковые модели могут быть неправильно использованы в профессиональной и академической письменной речи [16]. Исследователи из Стэнфорда недавно отметили [17], что с ростом чат-ботов присутствие искусственных фрагментов в научных статьях резко увеличивается и достигает нескольких десятков процентов. Более того, только за последний год было выявлено более 60.000 научных работ, в которых было доказано использование машинной генерации [18].

Пользователи Интернета все чаще используют генеративные сервисы для создания текстовых документов, не проверяя полученную в результате генерации информацию. Это лишь подчеркивает необходимость разработки более гибких и эффективных методов обнаружения и информирования пользователей, чтобы снизить возможные негативные последствия распространения сгенерированных фрагментов.

Область обнаружения машинно-сгенерированных текстов развивается уже много лет [19]. Этот тренд имеет объяснение: с каждым годом генеративные модели создают все более похожие на человеческие тексты, что в свою очередь побуждает исследователей обновлять детекторы, адаптируя их к новым результатам работы генеративных моделей. Еще до появления современных LLM с обнаружением машинно-сгенерированных текстов справлялись методы, основанные на числовых признаках, например, на статистических мерах текста (GROVER [20], N-граммы [21], признаки, созданные человеком [22]). Преимущество этих подходов заключается в том, что они не требуют большого количества обучающих данных. Указанные методы хорошо работают на доменах данных, которые использовались на этапе обучения, но плохо — на незнакомых примерах. В настоящее время также существует огромное количество методов обнаружения,

которые теряют свою актуальность с выходом обновленных и усовершенствованных моделей генерации либо требуют значительных вычислительных ресурсов для обновления [23].

Широко распространены подходы с дообучением моделей с архитектурой Трансформер [24]. Ранее был проведен обзор большого количества методов обнаружения и их сравнение [25]. В методах с дообучением моделей на основе архитектуры трансформер, текст кодируется предварительно обученным энкодером, преобразующим его в векторное представление в скрытом пространстве, а затем классификатор настраивается под конкретную задачу [26]. Переносимость возможностей обнаружения на сценарии с текстами из неизвестных доменов или от неизвестных моделей остается трудно решаемой задачей и представляет собой важную практическую сферу для исследований.

Лингвистические паттерны, такие как перплексия и логарифмическая функция вероятности, также используются во многих исследованиях. Например, zero-shot DetectGPT [27] работает в предположении, что различные вариации текста, сгенерированного машиной, обычно имеют меньшую вероятность, чем оригинальная генерация, в то время как написанные человеком могут иметь любую вероятность. Несмотря на свою эффективность по целевым метрикам качества, использование функции кривизны вероятности требует выполнения около сотни вызовов модели или взаимодействия с такими сервисами, как OpenAI API, для создания модификаций текстов, что приводит к большим вычислительным затратам.

Другой современный алгоритм обнаружения машинной генерации [28] требует доступа к вероятностям и функции потерь, выводимым LLM для скоринговой модели. Однако эти численные метрики и характеристики недоступны для GPT-3.5 и GPT-4. MAGE [29] и FastDetectGPT [30] требуют, чтобы скоринговая и целевая модели были одинаковыми. Ghostbuster [31] работает в предположении, что скоринговая и целевая модели различны, но все равно требует доступа к сгенерированным документам из целевой модели.

## Цели работы.

1. Предложить построение методов поиска, верификации и интерпретации машинно-сгенерированных текстовых последовательностей.
2. Предложить метод детекции искусственно-сгенерированных фрагментов внутри научных документов.
3. Предложить методы снижения количества обучаемых параметров нейросетевых моделей.
4. Предложить метод снижения сложности по Радемахеру обучаемой модели.
5. Предложить метод детекции искусственно-сгенерированных текстов, способный покрывать различные домены и быть настроенным для языкового разнообразия.

**Научная новизна.** Разработаны новые подходы в задаче детекции машинно-сгенерированных текстовых фрагментов. В ходе исследования были разработаны и предложены усовершенствованные методы, направленные на выявление признаков генерации на различных уровнях текстовой структуры: от слов и предложений до логических блоков внутри документа. Предложено использование методов мультизадачного обучения для снижения сложности обучаемой модели, а также регуляризации векторного пространства.

**Степень достоверности и апробация работы.** Достоверность результатов подтверждена математическими доказательствами, экспериментальной проверкой полученных методов на реальных задачах выбора моделей глубокого обучения; публикациями результатов исследования в рецензируемых научных изданиях, в том числе рекомендованных ВАК. Результаты работы докладывались и обсуждались на 9 научных конференциях. В рамках диссертационной работы также был разработан программный модуль, зарегистрированный в реестре государственных программ.

**Публикации по теме диссертации.** Основные результаты по теме диссертации изложены в 11 печатных изданиях, индексируемых в международных базах данных и в журналах, рекомендованных ВАК.

**Структура и объем работы.** Диссертация состоит из аннотации, оглавления, введения, четырех разделов, заключения, списка иллюстраций, списка таблиц и списка литературы из 64 наименований. Основной текст занимает 61 страницы.

**Краткое содержание работы по главам.** В главе 1 вводятся основные понятия, поставлены задачи детекции машинно-сгенерированного текстового фрагмента в рамках задачи бинарной классификации. Проанализированы методы решения задачи классификации при помощи дообучения моделей на основе архитектуры трансформер и при помощи формирования признакового пространства статистическими языковыми моделями.

В главе 2 предложены методы детекции машинно-сгенерированных текстовых фрагментов в составе документов. Исследованы подходы к распознаванию фрагментов с фиксированной и переменной длиной, включая алгоритмы разбиения текстового документа на непересекающиеся сегменты заданного размера.

В главе 3 предложены подходы агрегации методов детекции машинно-сгенерированных текстов.

В главе 4 представлен анализ и разработан метод детекции машинно-сгенерированных текстовых фрагментов, основанный на мультизадачном обучении, направленный на повышение обобщающей способности модели и снижение сложности по Радемахеру.

# Глава 1

## Детекция автора всего документа

Раздел посвящен методам детектирования машинно-сгенерированных документов. Предлагаются методы, основанные на использовании различных признаковых пространств для формирования векторного представления текста исходного документа. В данной главе рассматривается подход построения дискриминатора, основанного на архитектуре трансформер. Проанализирована задача детекции машинно-сгенерированных текстов с помощью бинарной классификации. Предложен подход дообучения предобученной модели, основанной на архитектуре трансформер, с ограниченным входным контекстом. Исследован метод использования статистических языковых моделей для формирования признакового представления текстов [32].

В главе введены предположения, описывающие формирование векторного представления текстов. В рамках данных предположений анализируются параметры, способствующие более точному отражению паттернов исходного текста в формируемом представлении. Рассматриваются ограничения реализаций в современных подходах детектирования.

В главе также анализируется подход формирования векторного представления при помощи статистических языковых моделей [33]. Метод позволяет внутри каждого текста оценивать вероятности последовательностей слов. Данная информация может быть использована для оценки авторства текста документа.

В вычислительном эксперименте проведено сравнение параметров, влияющих на качество решения задачи детекции [34]. Отражены замеры производительности при использовании статистических языковых моделей. Анализируются рассмотренные модели на синтетических выборках: общего доступа и созданных вручную. В подходе оценки параметров используются выборки TuringBench [35], RuATD [36], GPT-2 [37] и набор данных, который был создан собственноручно, используя различные параметры сэмплирования и открытые текстовые генеративные модели для русского языка. Данные, используемые в оценке метода со статистическими языковыми моделями, доступны в открытом доступе, а именно НСЗ [38], SemEval-2024 [39]. Вычислительный эксперимент использует модели разной сложности: BERT-подобные, TF-IDF совместно с LightGBM [40], признаки от статистических языков моделей совместно с LigthGBM, FastDetectGPT, Binoculars [28].

Основным результатом данной главы является обнаружение зависимости качества детектирования машинно-сгенерированных текстов от длины входной последовательности в моделях классификации, основанных на архитектуре трансформер. Также отражена возможность построения модели детекции с формированием признакового пространства, основанного на вероятности последовательностей слов, что позволяет ускорить подход детектирования в несколько раз.

## 1.1. Постановка задачи детекции в терминах классификации

Пусть  $\mathbf{W}$  — алфавит, содержащий минимальные элементы текстовых последовательностей — символы. Введем множество текстовых последовательностей, где каждая последовательность представлена конечной комбинацией символов алфавита:

$$\mathbb{D} = \left\{ \left[ t_j \right]_{j=1}^n \mid t_j \in \mathbf{W}, n \in \mathbb{N} \right\}.$$

Обозначим коллекцию из  $N$  примеров текстовых последовательностей:

$$\mathbf{D} = \bigcup_{i=1}^N D^i, D^i \in \mathbb{D}.$$

Для каждого примера необходимо определить, является ли он искусственно сгенерированным. Для обнаружения сгенерированного текста будет использован подход бинарной классификации.

Определим преобразование, выполняющее бинарную классификацию каждой текстовой последовательности.

$$g : \mathbb{D} \rightarrow \mathbf{C}.$$

В текущей настройке  $\mathbf{C} = \{0, 1\}$ , где метка  $c^i = 1$  соответствует сгенерированному фрагменту, а  $c^i = 0$  — фрагменту, написанному человеком.

Для выбора оптимального преобразования каждому документу из выборки  $\mathbf{D}$  присваивается метка  $c^i$ . Задача состоит в том, чтобы найти бинарный классификатор, который минимизирует эмпирический риск в наборе данных  $\mathbf{D}$ :

$$\hat{g} = \underset{g \in \mathfrak{F}}{\operatorname{argmin}} \sum_{i=1}^N \mathbb{I} [g(D^i) \neq c^i],$$

где  $D^i$  — текстовая последовательность из коллекции  $\mathbf{D}$ , а  $\mathfrak{F}$  — множество всех рассматриваемых алгоритмов классификации.

Функция потерь задачи классификации:

$$\mathcal{L}_{\text{BCE}}(g, \mathbf{D}) = -\frac{1}{N} \sum_{i=1}^N [c^i \cdot \log(g(D^i)) + (1 - c^i) \cdot \log(1 - g(D^i))].$$

Отслеживаемые метрики качества: *precision*, *recall*, *F<sub>1</sub>-score*.

## 1.2. Увеличение длины входного контекста

Исследуются параметры, способствующие более точному отражению структурных особенностей исходного текста в формируемом представлении. Исследователями ранее был выявлен наиболее точный метод детектирования, который

решает задачу бинарной классификации [23]. Данный метод основан на использовании моделей с архитектурой трансформер. Реализация данной архитектуры позволяет создавать модели с достаточно высоким уровнем понимания естественного языка. Данная настройка производится на этапе предобучения, где модель последовательно изучает текстовые документы со всего Интернета и усваивает лингвистические закономерности. Этот шаг позволяет получить алгоритм, который позже может быть адаптирован ко многим специфическим предметным областям.

Однако ключевым ограничением традиционных трансформеров остается максимальная длина входной последовательности. До внедрения современных позиционных энкодеров контекст кодировщиков часто ограничивался 256–1024 токенами, что затрудняет анализ протяженных текстов. Недостаточное покрытие контекста снижает способность модели улавливать удаленные зависимости и глобальные паттерны. Современные методы позиционного кодирования (такие как RoPE [41], ALiBi) частично смягчают эту проблему для больших языковых моделей, но ограничение длины контекста остается важным фактором при выборе модели и обработке данных.

Проанализировав характер данных, подаваемых на тонкую настройку моделей в рассматриваемой задаче детекции, была отмечена их короткая длина. Выдвинута гипотеза о том, что увеличение длины входной последовательности должно улучшить понимание глубокого контекста предобученной моделью.

Для русского языка последние работы в области исследования машинного распознавания текста основаны на примерах из данных RuATD (The Russian Artificial Text Detection). Однако в этих наборах данных тексты представлены короткими последовательностями. Выдвинуто предположение, что формирование набора искусственно-сгенерированных последовательностей с увеличенной длиной положительно повлияет на метрики качества детекции дообучаемого дискrimинатора для русского языка.

**Формирование набора данных.** Для исследования особенностей моделирования контекста создан мультиязычный набор данных на русском и английском языках. Русскоязычная часть сформирована на основе трех открытых моделей (rugpt3small, rugpt3large, XGLM-1.7B), англоязычная — на базе GPT-2. Каждый текст генерировался с варьированием ключевых параметров генерации: top-k (от 0 до 100), top-p (от 0.1 до 1.0) и температуры (0.7–1.5). Для английского языка использованы модели с объемом параметров от 117 млн до 1.5 млрд, что позволяет анализировать влияние масштаба архитектуры на качество контекста. Особенностью подхода стало включение в выборку текстов одинаковой длины, но с различной сложностью семантической структуры за счет комбинирования стратегий генерации. Данный шаг обеспечивает разнообразие паттернов написания текстов в рамках фиксированного объема контекста, что играет важную роль при дообучении моделей задаче распознавания границы смысловых блоков и иерархии логических связей.

В работе предлагается формирование набора данных, содержащего приме-

ры с длинным контекстом и разными настройками генерации, и использование контекста модели, основанной на архитектуре трансформер, вплоть до максимального значения.

### 1.3. Статистические языковые модели

В данной работе предложено рассмотреть решение задачи классификации текстовой последовательности с формированием собственного признакового пространства. Одним из популярных способов решения задачи детекции искусственно-сгенерированных последовательностей на сегодняшний день является дообучение предобученных кодировщиков, основанных на архитектуре трансформер. Однако большое количество методов обнаружения склонны к падению целевых метрик и потере актуальности с выходом обновленных и усовершенствованных моделей генерации либо требуют значительных вычислительных ресурсов для обновления [23]. В работе предлагается эффективный и легко обновляемый метод обнаружения происхождения текстовых фрагментов с использованием предварительно собранной информации с больших языков моделей.

За основу берется принцип, который лежит в задаче оптимизации выходов современных языковых моделей — минимизация уровня перплексии. В связи с этим, значение перплексии и значение логарифмической функции вероятности можно использовать как числовые признаки, характеризующие данный текст. В работе показано, что для поставленной задачи существует множество способов решения на основе этого предположения. Однако в связи с появлением все большего числа языковых моделей, веса и архитектура которых не доступны в открытом доступе или требуют больших ресурсов для запуска, эти методы становятся неактуальными ввиду невозможности вычисления истинных значений отмеченных величин. Доступность детектора для современных больших языковых моделей может стать проблемой для людей, не обладающих необходимым количеством вычислительных ресурсов. Исходя из этого, разработан гибко обновляемый метод, использующий приближенные значения величин перплексии и логарифмической функции вероятности и требующий для настройки только тексты из больших языковых моделей. Скорость работы разработанного подхода на стадии эксплуатации превосходит существующие методы в сотни раз.

Предложенный метод обнаружения сгенерированного текста идейно основан на вычислении значений перплексии и оценке логарифмической функции вероятности для входной текстовой последовательности  $x_{0:M-1} := x_0, x_1, \dots, x_{M-1}$  с помощью различных LLM, на основе которых алгоритм обнаружения вычисляет значение принадлежности текста к классу искусственных фрагментов. Перплексию последовательности можно описать:

$$PPL(x_{0:M-1}) = \exp\left(-\frac{1}{M} \sum_{m=0}^{M-1} \log P(x_m | x_0, \dots, x_{m-1})\right)$$

Значение перплексии можно интерпретировать показатель того, насколько хорошо языковая модель предсказывает распределение токенов в рассматриваемом фрагменте. Следовательно, чем меньше значение, тем лучше. В ряде работ [42, 43] уже отмечалось, что эта величина является полезной оценкой для построения классификатора, а перплексия для текстов, написанных человеком, как правило, велика.

Однако для вычисления истинного значения перплексии требуется сама большая языковая модель. Учитывая огромный размер и производительность современных языковых моделей на пользовательском оборудовании, их использование при каждом запуске метода уводит от цели построения быстрого подхода. Кроме того, вычисление перплексии по одной заранее выбранной языковой модели приводит к сложности адаптации метода к возникающим изменениям, а именно, к появлению новых языковых моделей и, соответственно, текстов с потенциально новой структурой.

**Гипотеза 1.** Пусть дана большая языковая модель, генерирующая распределение вероятностей над последовательностями токенов. Пусть на основе выходов  $LLM$  построен словарь  $N$ -грамм и обучена статистическая языковая модель с алгоритмом Кнезера-Нея порядка  $n$ . Для последовательности токенов  $x = (x_0, x_1, \dots, x_{M-1})$  длины  $M$  определим:

- Истинную перплексию  $LLM$ :

$$PPL_{true}(x) = \exp\left(-\frac{1}{M} \sum_{i=0}^{M-1} \log P_{LLM}(x_i | x_{<i})\right),$$

где  $x_{<i} = (x_0, \dots, x_{i-1})$ .

- Аппроксимированную перплексию с помощью статистической языковой модели:

$$PPL_{approx}(x) = \exp\left(-\frac{1}{K} \sum_{j=0}^{K-1} \log P_{KN}(x_{j+n-1} | c_j)\right),$$

где  $c_j = (x_j, x_{j+1}, \dots, x_{j+n-2})$  —  $j$ -й контекст длины  $n-1$  (для  $j = 0, 1, \dots, K-1$ ),  $K = M - n + 1$ , и  $x_{j+n-1}$  — токен, следующий за контекстом  $c_j$ .

Тогда при достаточно больших  $n$  и  $M$ , а также при условии, что статистическая модель обучена на достаточно большом и репрезентативном корпусе, выполняется соотношение:

$$PPL_{approx}(x) \approx PPL_{true}(x).$$

В приведенной выше формуле аппроксимации суммирование производится по количеству N-грамм входного текста, общее количество суммирований обозначается  $K$ . Основываясь на этой гипотезе, в работе предложен метод обнаружения искусственно созданных фрагментов. Метод представлен на Рисунке 1.1.

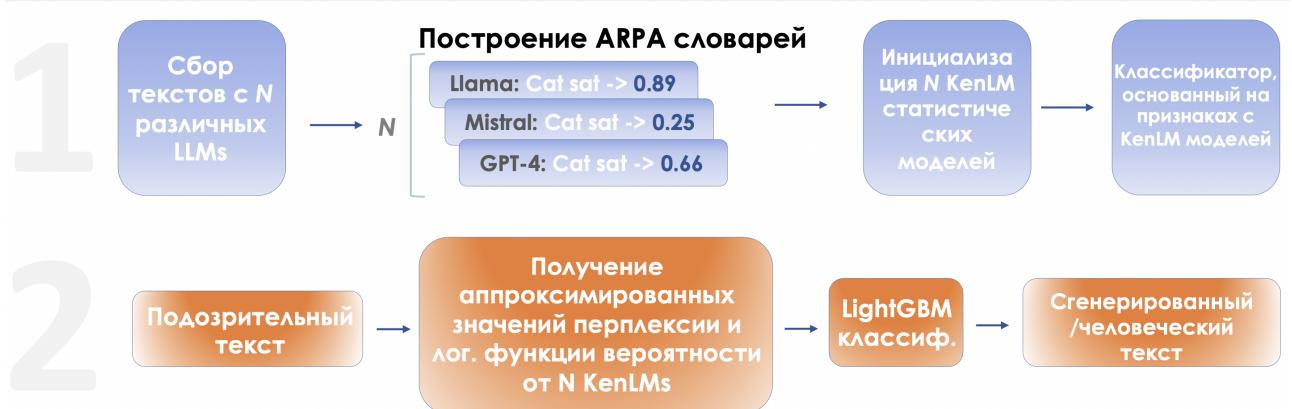


Рис. 1.1: Цикл работы предлагаемого метода, основанного на использовании статистических языковых моделей. На первом этапе выполняется предварительная настройка, для которой требуются тексты из разных LLM. На их основе строятся Агра-словари. Агра-словари используются для инициализации статистических языковых моделей KenLM. На втором этапе представлен алгоритм обнаружения. Подозрительный текст представляется в числовом виде на основе признаков от предварительно собранных KenLM. После этого векторы подаются на вход предварительно обученного LightGBM-классификатора, который и производит окончательную классификацию.

Данный метод требует предварительной настройки, то есть сбора словарей N-грамм и предварительного обучения классификатора на числовых признаках. Процесс заключается в сборе текстов от больших языковых моделей, создании словарей N-грамм с вероятностями генерации следующих символов на основе предыдущих и получении статистических языковых моделей. Предполагается, что с помощью этих инструментов можно аппроксимировать значение перплексии каждой используемой модели на основе встречаемости элементов текстовых последовательностей.

**Алгоритм KenLM.** В области статистических языковых моделей существует множество подходов к построению и оценке вероятностей последовательностей слов или токенов. Наиболее популярными являются N-граммные модели, которые оценивают вероятность очередного слова на основе фиксированного числа предыдущих слов. Для данного исследования был выбран подход KenLM, реализующий эффективные алгоритмы построения N-граммных моделей и их сглаживания.

Подход KenLM заключается в построении иерархии N-грамм (пар, троек, четвёрок слов и т. д.) на основе обучающего корпуса текста. Каждая N-грамма

представляет собой комбинацию из  $n$  последовательных слов, где для каждой такой комбинации вычисляется её вероятность при условии предыдущих  $n - 1$  слов. Модель использует метод сглаживания Кнезера–Нея [44], который позволяет корректно оценивать вероятности редких или даже не встречавшихся ранее N-грамм за счёт перераспределения вероятностной массы между известными и неизвестными событиями.

Формула сглаживания Кнезера–Нея для биграммной модели имеет следующий вид:

$$P_{\text{KN}}(w_i \mid w_{i-n+1}^{i-1}) = \frac{\max(C(w_{i-n+1}^i) - \delta, 0)}{C(w_{i-n+1}^{i-1})} + \alpha \cdot P_{\text{KN}}(w_i \mid w_{i-n+2}^{i-1}),$$

где  $C(\cdot)$  — частота встречаемости соответствующей N-граммы в корпусе,  $\delta$  — параметр сдвига, используемый для сглаживания,  $\alpha$  — нормировочный коэффициент, обеспечивающий сохранение суммы вероятностей,  $P_{\text{KN}}(w_i \mid w_{i-n+2}^{i-1})$  — рекурсивная оценка вероятности на основе  $(n - 1)$ -грамм.

Данный метод особенно эффективен в условиях разреженности данных, что делает его пригодным для работы с большими текстовыми корпусами, а также обеспечивает хорошую аппроксимацию распределений, генерируемых современными большими языковыми моделями.

**Формирование набора данных.** В данной работе преследовалась цель создать как можно более обширную базу данных словарей различных LLM. Поскольку стало понятно, что каждая статистическая языковая модель может хранить особенности конкретной LLM, на выходах которой она была скомпилирована. Агрегируя такие словари, можно получить детектор сгенерированных текстов с высокой обобщающей способностью. Для создания Агра-словарей, ключевой составляющей пакета статистических языковых моделей KenLM, были взяты из открытых источников выходные данные моделей GPT-2, GPT-3.5, GPT-4, PALM, Llama-2 и Mistral. Кроме того, была сохранена часть собранных данных для тестирования и настройки классификатора. Стоит отметить, что не рекомендуется использовать данные, которые включались в формирование Агра-словарей далее в обучающий или тестовый наборы данных, так как это косвенно приводит к утечке данных.

Собрать тексты из современных LLM гораздо проще, чем получить доступ к их весам. Таким образом, самостоятельно построив несколько статистических языковых моделей на основе текстов из актуальных LLM, мы можем предсказать для текстового фрагмента его значение перплексии и логарифмической функции вероятности. Эти значения формируют числовые характеристики подозрительного текста в векторном пространстве. Далее, используя полученные векторные представления для каждого текста, для решения задачи классификации применяется алгоритм градиентного бустинга.

## 1.4. Анализ качества детекции текстовых последовательной на разных языках

Проводится серия вычислительных экспериментов для анализа предложенных методов детекции искусственно-сгенерированных текстовых последовательностей при помощи решения задачи бинарной классификации.

**Анализ контроля длины.** Длина входной последовательности может напрямую влиять на способность классификатора понимать контекст. Д. Ипполито в работе [25] показал, что при использовании различных методов сэмплирования увеличение длины входной последовательности улучшает качество дискриминаторов, основанных на архитектуре трансформер. Однако в ранее опубликованных работах исследователи остановились на длине  $n=192$ . Для выборок, построенных с помощью метода  $top-k$ , этого оказывается достаточно. Тем не менее, как видно из рис. 1.2, для методов  $top-p$  и  $pure$  детектор на английском языке выходит на плато по метрикам качества только при рассмотрении от  $n=512$  до  $n=1024$  токенов входного текста. Солайман и др. в работе был проведен эксперимент, обучая модель на последовательностях длиной до  $n=128$ , а тестируя — на более длинных, до  $n=512$ . С точки зрения задачи это означает, что обучающие и тестовые выборки взяты из разных генеральных совокупностей. В нашем же эксперименте предполагается, что данные согласованы по длине, а значит, принадлежат к одной и той же генеральной совокупности.

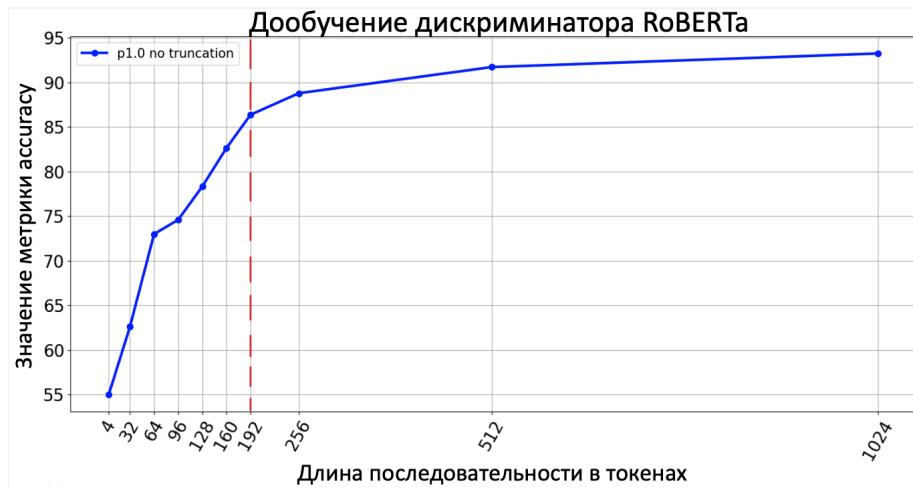


Рис. 1.2: Точность детекции для английского языка растет по мере увеличения длины последовательностей, используемых для обучения дискриминатора. После  $n=512$  метрика достигает плато. Красная линия показывает границу предыдущих экспериментов с увеличением длины.

Увеличение длины входной последовательности значительно повышает время обучения модели-детектора. Однако именно длина позволяет модели лучше улавливать долгосрочные зависимости, что в ряде случаев становится ключевым фактором при определении принадлежности текстовой последовательности к классу машинно-сгенерированных.

На графике (рис. 1.3) представлено распределение количества токенов в датасете RuATD и было замечено, что большинство примеров содержит не более 200 токенов. Точность детектора, обученного на таких данных, будет низкой в общем случае.

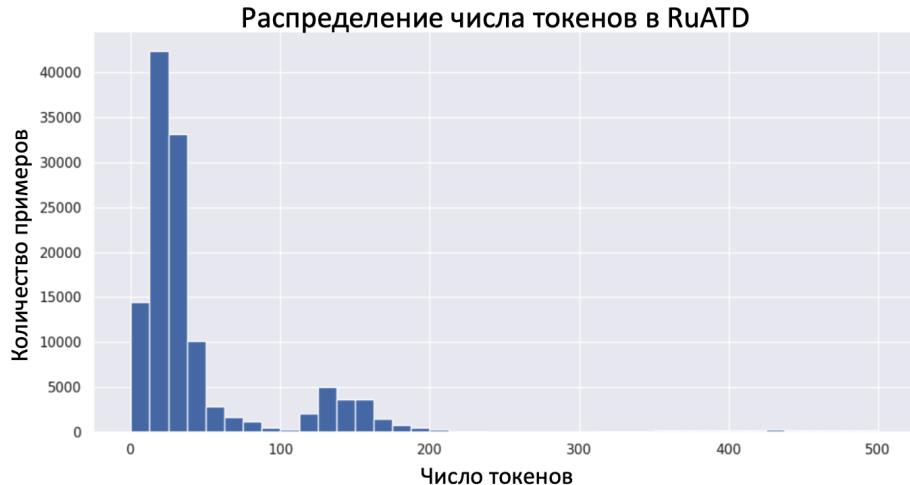


Рис. 1.3: Большинство образцов данных RuATD имеют длину до 200 токенов, при таком количестве токенов модели будет сложно понять длинный контекст.

В экспериментах была проверена гипотеза о том, что увеличение длины текста положительно скажется на качестве обучаемого детектора, протестировав её на собственном сгенерированном наборе данных для русского языка. Помимо этого, тесты были проведены для всех вышеупомянутых методов сэмплирования символов. Из-за ограниченного числа доступных предобученных моделей эксперимент проводился только до длины  $n=512$ , однако этого оказалось достаточно, чтобы метрика достигла плато. Как видно из рис. 1.4, диапазон длин последовательностей от  $n=256$  до  $n=512$  является критическим для понимания контекста в рамках предложенной модели, что подтверждает важность использования более длинных текстов в обучающих данных. Значение точности, полученное при  $n=64$  (средняя длина токенов в выборках из RuATD), коррелирует с результатами конкурса DIALOG-22 RuATD [36].

Кроме того, было замечено, что в отличие от английских датасетов, где тексты, сгенерированные с помощью метода *top-k*, достигают плато уже на 16 токенах [25], для русского языка такое предположение не выполняется — все три указанных метода сэмплирования показали сравнимые результаты.

**Построение словарей для статистических языковых моделей.** В подходе решения задачи детекции при помощи статистических языковых моделей, их инициализация происходит при помощи предварительно собранных словарей, имеющих название Arpa.

На этапе качественных экспериментов было выявлено, что создание Arpa-словаря — один из важнейших элементов успеха будущего классификатора. Для формирования более содержательного объекта был построен словарь на

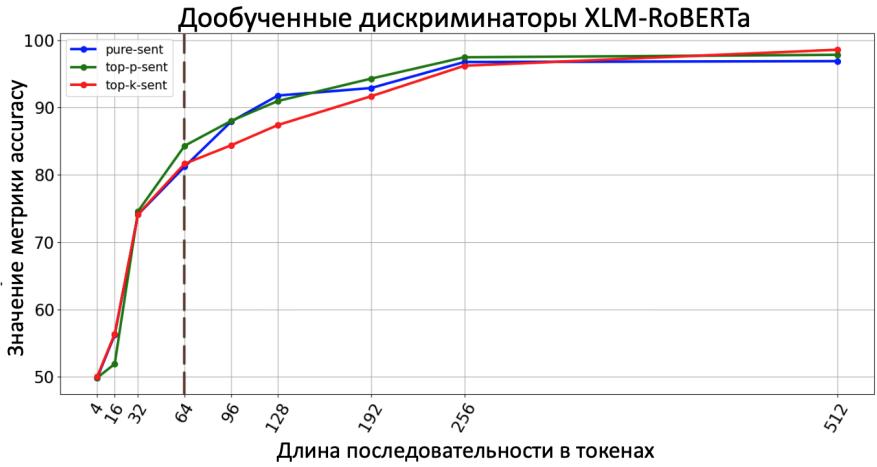


Рис. 1.4: Точность детекции на русском языке возрастает по мере увеличения длины последовательностей, используемых для обучения дискриминатора, для всех представленных методов выборки. После  $n=256$  метрика достигает плато. Коричневая линия показывает среднюю длину лексем в наборе данных RuATD.

30 тысяч текстов из модели Mistral. После было замечено сильное увеличение количества уникальных N-грамм при  $N > 2$ . В наших экспериментах мы построили словарь с N-граммами вплоть до  $N = 4$ . Пример показан на Рисунке 1.5. Было замечено, что можно уменьшить количество уникальных N-грамм, преобразовав тексты в токенизированную форму. Для этого был использован токенизатор, используемый в модели Mistral. Выдвинуто предположение, что он может помочь решить проблему перегрузки словаря и добавить больше стандартизации в этот подход.

**Arpa on words:**  
 ngram 1 = 177.719  
 ngram 2 = 3.969.976  
 ngram 3 = 13.102.059  
 ngram 4 = 20.381.215

**Arpa on tokens:**  
 ngram 1 = 18.420  
 ngram 2 = 2.307.583  
 ngram 3 = 8.636.089  
 ngram 4 = 14.127.773

Рис. 1.5: Сравнение статистики N-грамм Arpa-словарей, построенных на словах и токенах. Можно заметить, что использование токенов при построении позволяет уменьшить количество уникальных N-грамм, но порядок значений остается прежним.

Кроме того, в данном исследовании проверялась гипотеза о том, насколько хорошо статистическая языковая модель KenLM способна предсказывать вероятности появления следующего символа в N-грамме по сравнению с истинными значениями. Для этого эксперимента был взят уже созданный Arpa-словарь на основе 30 тысяч текстов и рассчитан а с помощью модели Mistral для каждой N-граммы из существующего словаря новая вероятность следующего токена.

Замена производилась в несколько шагов от одного уровня ко всем. Для тех строк, в которых вероятность следующего токена была неизвестна, мы оставляли предыдущее значение. Такая замена была проведена для Арга-словаря на основе слов, где удалось заменить 78% строк с N-граммами, и для словаря на основе токенов, где охват замены составил 99%.

Проведя исследование на одном заранее собранном статистическом словаре, идея была расширена, чтобы собрать ансамбль аналогичных словарей с тем же количеством текстов для генерации, но из совершенно разных LLM. Здесь были подключены различные модели семейств GPT и Llama. Эксперименты с ансамблями также проводились с двумя парадигмами построения Арга-словарей: словесной и токенизированной. Включение словарей нескольких семейств моделей должно способствовать расширению признакового пространства и как следствие, более репрезентативного представления для каждого текста.

Стоит отметить, что в случае одиночного словаря было использовано только два признака для классификации, а именно, как уже говорилось выше, перплексия и логарифмическая функция вероятности. В случае ансамбля размерность числового вектора росла пропорционально количеству используемых моделей и достигла 12 признаков. Поверх полученных признаков на отложенных обучающих данных был обучен алгоритм градиентного бустинга LightGBM.

### **Вычислительный эксперимент с использованием статистических языковых моделей.**

Таблица 1.1: Результаты экспериментов со статистическими языковыми моделями KenLM. Рассматриваются различные варианты создания Арга-словаря, а также их объединение в ансамбль. Значение в каждой ячейке представляет собой метрику полноты обнаружения искусственно сгенерированного фрагмента.

Эксперименты KenLM	Основа построения	GPT-2	GPT-3.5	GPT-4	Llama	Mistral
Mistral-arpa	Слова	0.92	0.95	0.90	0.73	0.92
Mistral-arpa	Токены	0.89	0.98	0.71	0.56	0.86
Mistral-arpa and custom uni-grams	Слова	<b>0.92</b>	0.97	0.90	0.73	0.90
Mistral-arpa and custom uni-grams	Токены	0.87	0.98	0.71	0.57	0.87
Mistral-arpa and custom uni+bi-grams	Токены	0.89	0.98	0.74	0.62	0.87
Mistral-arpa and custom uni+bi+tri-grams	Токены	0.89	0.98	0.75	0.67	0.88
Ансамбль (Mistral, Llama, GPT-2, GPT-3.5, GPT-4)	Слова	0.89	<b>0.99</b>	<b>0.97</b>	<b>0.87</b>	<b>0.96</b>
Ансамбль с отбором признаков	Токены	0.85	0.98	0.94	0.65	0.94

В Таблице 1.1 приведены результаты описанных экспериментов по формированию устойчивой модели на общедоступных выходах различных LLM. Для каждой доменной модели приведены значения полноты обнаружения текстов, сгенерированных каждой из них соответственно. Видно, что при использовании ансамбля моделей наблюдается значительный разрыв в качестве. Комбинирование признаков N-грамм, присвоенных выходам различных больших языковых моделей, позволяет охватить более широкий диапазон генерируемых текстов. Кроме того, в эксперименте по замене вероятности следующего слова/токена в данной работе эмпирически доказана корректность поставленной ранее гипотезы 1. Замена вероятности следующего слова/токена на истинную вероятность исследуемой модели показывает незначительную разницу. Это свидетельствует о высокой способности статистических языковых моделей предсказывать вероятности на основе большого количества текстов, которые значимо не отличаются от истинных вероятностей. Что позволяет сделать вывод, что величина перплексии, которая является результатом комбинирования этих вероятностей, аппроксимируется достаточно близко. Стоит также отметить, что важен вклад каждого словаря, так как при отборе признаков наблюдается падение метрик, когда рассматривается только часть из них.

Таблица 1.2: Результаты сравнения предложенного метода с уже известными подходами на трех разных наборах данных. Собранный вручную набор данных с публично доступными выходами различных моделей содержит 2 тыс. сбалансированных примеров, а HC3 и Binoculars - по 1 тыс. примеров. Можно заметить, что производительность предложенного метода значительно выше, чем у аналогов.

Метод	Accuracy	Precision	Recall	F1-score	Время(s)
<i>Данные, собранные на основе открытых коллекций</i>					
TF-IDF	0.90	0.93	0.87	0.90	0.36
Fast-DetectGPT	0.58	0.72	0.25	0.37	471*
Binoculars	0.54	<b>0.97</b>	0.08	0.14	236*
<i>Описанный подход</i>	<b>0.91</b>	0.87	<b>0.95</b>	<b>0.91</b>	<b>0.27</b>
<i>HC3</i>					
TF-IDF	0.55	0.54	0.66	0.59	0.14
Fast-DetectGPT	0.90	0.88	0.96	0.86	214*
Binoculars	<b>0.94</b>	<b>0.99</b>	0.89	<b>0.94</b>	108*
<i>Описанный подход</i>	0.90	0.84	<b>0.97</b>	0.90	<b>0.09</b>
<i>SemEval-2024</i>					
TF-IDF	0.44	0.45	0.51	0.48	0.20
Fast-DetectGPT	<b>0.93</b>	0.93	0.90	0.91	229*
Binoculars	0.65	<b>0.99</b>	0.31	0.48	125*
<i>Описанный подход</i>	<b>0.93</b>	0.87	<b>0.98</b>	<b>0.92</b>	<b>0.16</b>

Результаты экспериментов на внешних данных приведены в Таблице 1.2. В качестве меры оценки были выбраны точность, полнота и  $F_1$ -score, так как

решается задача классификации, кроме того, приведен столбец с временем запуска в секундах. Для сравнения с предлагаемым в данной работе методом были выбраны базовые, быстрые и современные подходы обнаружения машинно-генерированных фрагментов: TF-IDF, Fast-DetectGPT, Binoculars. По результатам сравнения можно заметить, что предложенный метод имеет высокий уровень полноты обнаружения, который достигается на всех представленных наборах данных. Также стоит отметить, что уровень точности обнаружения поддается корректировке при изменении порога классификации на финальной стадии. Кроме того, предложенный метод быстрее всех своих оппонентов работает на CPU, в несколько сотен раз быстрее state-of-the-art на GPU, что еще раз говорит о том, что полученный подход сочетает в себе как качество работы, которое гибко обновляется и настраивается, так и отличную производительность. Запуск на CPU осуществлялся на AMD Ryzen 9 5900X с 24 ядрами, измерения на GPU проводились на NVIDIA GeForce RTX 3090.

**Абляционное исследование.** Для уточнения более тонких настроек предложенного метода было решено провести исследование абляции. В качестве первого исследования была замерена зависимость качества метрик классификации от количества статистических моделей KenLM, использованных для построения числового вектора текстового фрагмента. На Рисунке 1.6 виден рост метрик с увеличением количества задействованных Арга-словарей. Это отражает предложенное в данной работе свойство, основанное на том, что большое количество N-грамм, взятых из выходов LLM, способно представлять свойства и приближенное распределение конкретных моделей.

Таким образом, объединение статистических моделей, построенных на текстах разного времени, позволяет создать пространство текстов с характеристиками, охватывающими большое количество паттернов, присущих генерированным текстам разных времен. Кроме того, в данной работе было проведено исследование, в котором произведена попытка выявления зависимости качества от количества текстов, используемых при формировании словарей, но все вариации от 30 до 60 тысяч оказались незначительными.

В работе были исследованы пороги в финальной классификации, где необходимо отнести пример к одному из двух классов в соответствии со значением предсказания вероятности. На Рисунке 1.7 можно увидеть результат изменения порога от минимального до максимального значения. Стоит отметить, что для каждого из наборов можно подобрать порог, который удовлетворит как метрику полноты обнаружения, так и уровень ложных срабатываний. Что касается свойств классификатора, то исследование проводилось не только лишь на одном методе градиентного бустинга, были опробованы различные методы из классического машинного обучения, а также вариации деревьев, но это не привело к существенным изменениям качества.

В данной работе также был исследован вопрос, на отрывках какой длины предложенный подход работает лучше. В результате эксперимента выяснилось, что на текстах длиной до 1 500 символов качество полноты поиска достигает

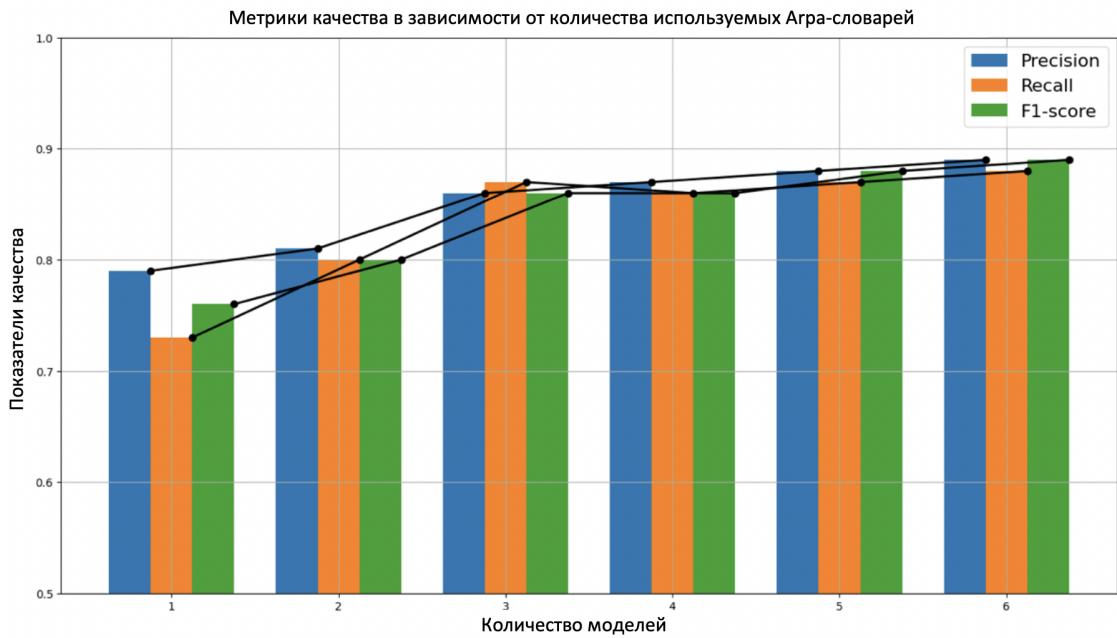


Рис. 1.6: Сравнение метрик качества классификации в зависимости от количества статистических языковых моделей, используемых для построения числового представления подозрительного текста. Можно заметить, что качество становится выше при использовании большего числа моделей KenLM.

89%, а на текстах длиной более 1 500 - только 79%. Точность в обоих случаях остается сравнительно высокой.

Из анализа проведенных вычислительных экспериментов видно, что в случае обучения трансформерного дискриминатора более длинная входящая текстовая последовательность примеров, позволяет достичь более высокого качества детекции машинно-сгенерированных текстовых последовательностей. Также в главе представлен иной подход решения задачи, основанный на признаках, формируемых аппроксимируемыми значениями больших языков моделей, и моделей градиентного бустинга. По результатам на целевых метриках, подход отмечается быстродействием и высоким качеством работы. Помимо этого метод является легко обновляемым и не требует наличия графического процессора для вычислений.

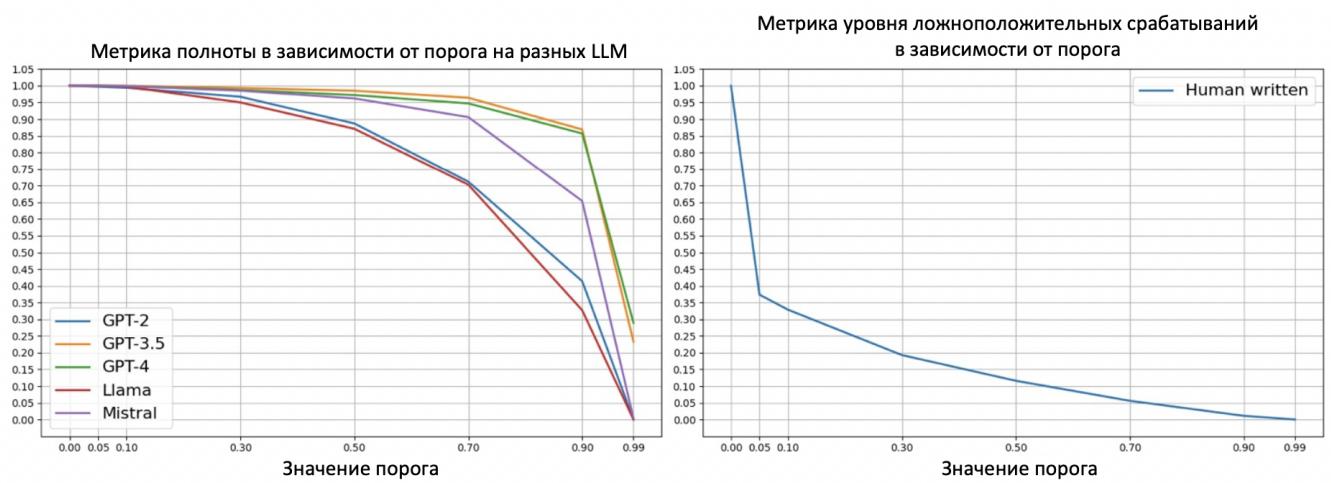


Рис. 1.7: Графики изменения порогового значения классификации. Левый график показывает значение полноты обнаружения при изменении порога, а правый — уровень ложных срабатываний на документах, написанных человеком, при изменении порога.

## Глава 2

### Детекция сгенерированных фрагментов внутри документа

Рассматривается проблема детектирования машинно-сгенерированных текстовых фрагментов. Исследуются методы распознавания искусственно созданных фрагментов фиксированной и варьируемой длин. Предлагаются методы разделения документа на множество непересекающихся фрагментов фиксированной длины. Показан способ учета, потенциально возникающей проблемы, большой мощности множества фрагментов внутри одного документа. Исследована постановка задачи детекции машинно-сгенерированных фрагментов в виде классификации токенов. Продемонстрирован способ решения задачи классификации токенов при помощи мультиголовой модели, основанной на архитектуре трансформер.

В главе поставлена задача детектирования сгенерированных фрагментов внутри документа. Предполагается делить документ на множество непересекающихся фрагментов фиксированной длины. Для учета потенциального накопления ошибки первого рода вводятся поправки на множественную проверку гипотез. Данная корректировка позволяет более корректно проводить финальную классификацию фрагмента. В качестве повышения обобщающей способности детектора в главе дополнительно показаны эксперименты по настройке этапе дообучения дискриминатора фрагментов, включающие конкатенацию признаков созданных вручную, перефразирования, перевод и мультиязычность [10].

Другим подходом к детектированию сгенерированных фрагментов внутри документа является постановка задачи в терминах задачи классификации токенов. Данный метод позволяет устраниТЬ зависимость от разделения документа на фрагменты фиксированной длины и выдаТЬ разметку для каждого токена для исходного текста любой длины. Фрагменты не привязаны к конкретному алгоритму разделения и формируются лишь на основе контекста. Рассматривается решения задачи при помощи обучения мультиголовой модели, основанной на архитектуре трансформер. А именно, подход сосредоточен на использовании архитектуры многозадачного обучения с двумя головами [1].

Для анализа качества работы предложенных методов детекции искусственно-сгенерированных фрагментов проводилась серия вычислительных экспериментов. Сформирован ряд наборов данных, включающих примеры с перефразированием, переводом и мультиязычными примерами (русский и английский языки). Проводилось дообучение предобученных трансформерных моделей на собранных наборах данных, преимущественно научного домена. Помимо этого тестирование проводится на наборах данных из открытого доступа DagPap24 [45], Yandex-Q, Russian Essays, Alpaca, DeepFake [46].

## 2.1. Постановка задачи детекции фрагментов фиксированной длины

Пусть  $\mathbf{W}$  — это алфавит, содержащий минимальные элементы текстовых последовательностей, — символы. Вводится множество документов, где каждый документ представляется конечной комбинацией символов алфавита:

$$\mathbb{D} = \left\{ \left[ t_j \right]_{j=1}^n \mid t_j \in \mathbf{W}, n \in \mathbb{N} \right\}.$$

Зададим выборку из  $N$  документов:

$$\mathbf{D} = \bigcup_{i=1}^N D^i, D^i \in \mathbb{D}.$$

Требуется найти в каждом документе  $D^i$  машинно-сгенерированные последовательности символов. Для этого представим двухэтапный алгоритм. Изначально необходимо разделить текст исходного документа на фрагменты (непересекающиеся подпоследовательности) фиксируемой длины, далее провести бинарную классификацию каждого полученного текстового фрагмента.

Зададим модель  $\phi$  в виде суперпозиции двух преобразований  $\mathbf{f}$  и  $\mathbf{g}$ . Преобразование  $\mathbf{f}$  представляет собой разделение документа на фрагменты, а преобразование  $\mathbf{g}$  — классификатор текстовых последовательностей, верифицирующий наличие сгенерированных частей:

$$\phi = \mathbf{g} \circ \mathbf{f},$$

$$\phi : \mathbb{D} \rightarrow \mathbb{T}, \quad \mathbb{T} = \left\{ \left[ t_{s_j}, t_{f_j}, c_j \right]_{j=1}^J \mid t_{s_j} = t_{f_{j-1}}, s_j \in \mathbb{N}_0, f_j \in \mathbb{N}, \right.$$

где  $J$  количество фрагментов после этапа фрагментации документа,  $t_{s_j}$  стартовый индекс  $j$ -ого фрагмента,  $t_{f_j}$  завершающий индекс  $j$ -ого фрагмента,  $c_j$  метка класса  $j$ -ого фрагмента.

В текущей постановке  $\mathbf{C} = \{0, 1\}$ , где метка  $c_j = 1$  соответствует машинно-сгенерированному фрагменту, а  $c_j = 0$  соответствует фрагменту, который был написан человеком.

Первый элемент суперпозиции — преобразование, позволяющее разделить текст на непересекающиеся фрагменты:

$$\mathbf{f} : \mathbb{D} \rightarrow \mathbf{T}^*, \quad \mathbf{T}^* = \left\{ \left[ t_{s_j}, t_{f_j} \right]_{j=1}^J \mid t_{s_j} = t_{f_{j-1}}, s_j \in \mathbb{N}_0, f_j \in \mathbb{N} \right\},$$

где  $s_j$  обозначает стартовый индекс  $j$ -ого фрагмента и  $f_j$  обозначает завершающий индекс  $j$ -го фрагмента. Таким образом,  $\mathbf{T}^*$  представляет собой множество всех возможных непересекающихся фрагментов, которые полностью покрывают документ.

Вторым преобразованием проводится бинарная классификация каждого текстового фрагмента.

$$\mathbf{g} : \mathbf{T}^* \rightarrow \mathbf{C}.$$

Внутри документа  $D^i$  из заданной выборки  $\mathbf{D}$  каждому фрагменту поставлена в соответствие метка  $c_j^i$ . Далее задача состоит в том, чтобы найти бинарный классификатор, который минимизирует эмпирический риск в наборе данных  $\mathbf{D}$ :

$$\hat{g} = \operatorname{argmin}_{g \in \mathfrak{F}} \sum_{D^i \in \mathbf{D}} \sum_{x_j^i, c_j^i \in D^i} [g(t(x_j^i)) \neq c_j^i], \quad t : \mathbf{T}^* \rightarrow (V)^n,$$

где  $x_j^i$  фрагмент документа  $D^i$ ,  $t$  - токенизатор,  $V$  - словарь всевозможных токенов предобученной модели,  $n$  - фикс. длина входного вектора, а  $\mathfrak{F}$  набор всех рассмотренных алгоритмов для классификации.

Функция потерь задачи классификации:

$$\mathcal{L}_{\text{BCE}}(g, \mathbf{D}) = -\frac{1}{|\mathbf{D}|} \sum_{D^i \in \mathbf{D}} \sum_{(x_j^i, c_j^i) \in D^i} [c_j^i \cdot \log(\hat{g}(t(x_j^i))) + (1 - c_j^i) \cdot \log(1 - \hat{g}(t(x_j^i)))] ,$$

Отслеживаемые метрики качества: *precision*, *recall*, *F<sub>1</sub>-score*.

## 2.2. Суперпозиция преобразований для детекции фрагментов

Исследуется задача детектирования сгенерированных фрагментов внутри документа. Предполагается решать задачу суперпозицией преобразований, где каждый документ делится на множество непересекающихся фрагментов фиксированной длины, где далее проводится классификация каждого фрагмента из образованного множества.

**Поправки на множественное тестирование** При описанном подходе детекции машинно-сгенерированных текстовых последовательностей чем больше оказывается длина документа, тем больше возникает фрагментов, которые затем подаются на вход обученной модели классификации. Сформулируем задачу уточнения авторства фрагмента на языке статистических гипотез:

$$\begin{aligned} H_0 &: \hat{g}(\text{fragment}) = 0, \\ H_1 &: \hat{g}(\text{fragment}) = 1. \end{aligned}$$

Зафиксируем уровень значимости  $\alpha = 0.05$ . При проверке поставленной гипотезы возникает ситуация, когда во время множественного тестирования, а именно классификации  $m$  фрагментов исходного документа с уровнем значимости  $\alpha$ , будут накапливаться ошибки первого рода (ложные отклонения гипотезы). Верхняя оценка вероятности того, что хотя бы один из них будет неверным:

$$P(\text{false positive}) = 1 - (1 - \alpha)^m,$$

величина которой достаточно велика уже при небольших значениях  $m$ . Требуется контролировать ошибку первого рода. Для этого существует несколько способов корректировки уровня значимости, которые различаются мощностью тестов. Для соблюдения баланса в описанных сущностях вводятся две меры, контролирующие ошибки первого рода: family-wise error rate (FWER) — групповая вероятность ошибки и false discovery rate (FDR) — ожидаемая доля ложных отклонений.

$$FWER = P(V > 0), \quad FDR = \mathbb{E}\left(\frac{V}{V + S}\right),$$

где  $V$  — число ложно положительных результатов (ошибок первого рода), а  $S$  — число истинно положительных результатов.

Одним из распространенных методов контроля на FWER на уровне  $\alpha$  является метод Холма [47]. Первый шаг — сортировка по возрастанию уровней  $p$ -value. Тогда уровень значимости при этом меняется в следующем порядке:

$$\alpha_1 = \frac{\alpha}{m}, \quad \alpha_2 = \frac{\alpha}{m - 1}, \dots, \quad \alpha_i = \frac{\alpha}{m - i + 1}, \dots, \quad \alpha_m = \alpha.$$

Далее процесс выстраивается следующим образом: если  $p_1 \geq \alpha_1$ , то все нулевые гипотезы не отвергаются, иначе отвергается первая, и процесс продолжается. В результате мы гарантированно удерживаем  $FWER \leq \alpha$ .

Иной метод, используя который можно добиться высокого уровня мощности — метод Бенджамини-Хохберга [48], где контролируется FDR. После сортировки по возрастанию  $p$ -value уровень значимости в данном подходе меняется по следующему закону:

$$\alpha_1 = \frac{\alpha}{m}, \quad \alpha_2 = \frac{2\alpha}{m}, \dots, \quad \alpha_i = \frac{i\alpha}{m}, \dots, \quad \alpha_m = \alpha.$$

Здесь процесс выстроен с иной стороны, в отличие от метода Холма: если  $p_m < \alpha_m$ , то необходимо отвергнуть все гипотезы, иначе — не отвергать  $m$ -ую, и продолжить.

**Гипотеза 2.** Для любой процедуры множественного тестирования

$$FDR \leq FWER,$$

поскольку  $\frac{V}{V + S} \leq \mathbb{I}[V > 0]$ , и при  $V + S = 0$  по соглашению  $\frac{V}{V+S} = 0$ .

Из гипотезы можно сделать вывод, что в большинстве случаев метод Бенджамини-Хохберга оказывается мощнее метода Холма, он отвергает не меньше гипотез с теми же  $\alpha_i$ . В связи с этим в данной работе выбор был сделан в сторону метода Бенджамини-Хохберга.

На Рисунке 2.1 изображен полный цикл работы алгоритма детекции машинно-сгенерированных фрагментов для домена научных документов. Данная комбинация идеи фрагментации и статистических поправок позволяет провести проверку научной работы всецело, не теряя при этом части текста из-за

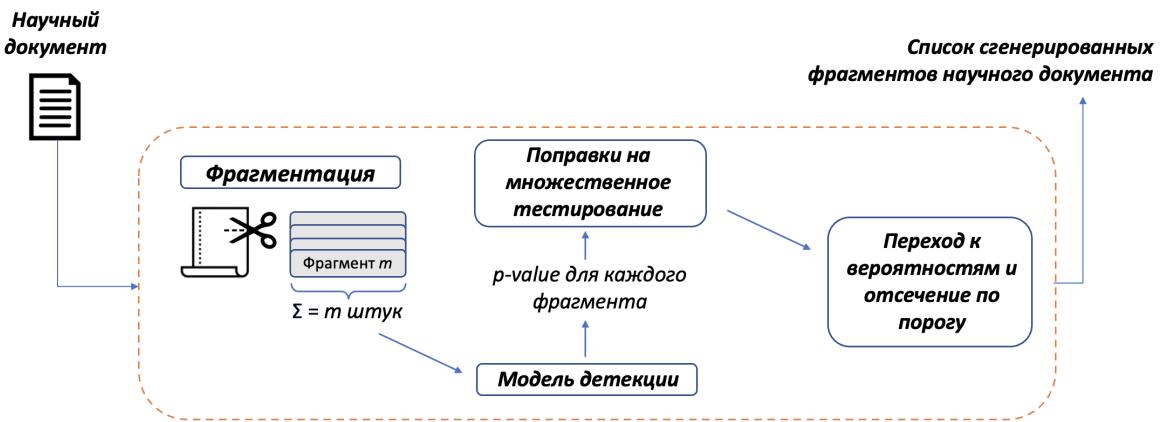


Рис. 2.1: Полный цикл работы алгоритма детекции сгенерированных фрагментов в научных документах.

ограничения моделей детекции по длине входной последовательности и не накапливая ошибку отклонения верной гипотезы.

**Формирование набора данных** Для исследования особенностей детекции текстовых фрагментов документов собран мультиязычный набор данных на русском и английском языках. Исследователями ранее было показано, что высокий уровень полноты детекции и большая обобщающая способность модели достигается при обучении на мультидоменных текстах [23]. Чем больше разнородных по тематикам документов будет на стадии обучения, тем выше будут метрики качества классификации при использовании детектора в реальных задачах поиска искусственных отрывков [49]. На английском языке существует большое число наборов данных в открытых источниках, на русском — выбор ограничен. В рамках данной работы проведен анализ имеющихся датасетов и проведено объединение некоторых из них для каждого из представленных языков. Стоит отметить, что наборов данных для детекции машинно-сгенерированных текстов, примеры из которых сравнимы с публикуемыми статьями и исследовательскими работами по размеру, в открытом доступе не найти.

В собранном датасете для русского языка настоящие тексты представлены следующими источниками: MGTDR — статьи с ресурса Википедии, Yandex Q — ответы на вопросы с одноимённой платформы, и Russian Essays — сборник школьных сочинений на гуманитарные темы. Искусственно сгенерированные тексты включают: материалы MGTDR, созданные по заголовкам статей с Википедии; задания и ответы из Alpaca, сгенерированные моделью ChatGPT; диалоги из Saiga, также полученные с помощью ChatGPT; и тексты на темы из Russian Essays, сгенерированные той же моделью. Для английской части датасета настоящие тексты взяты из: DeepFake (эссе, новости, истории, научные тексты) и GPT-4 on Instructions. Искусственные тексты представлены сгенерированными примерами из DeepFake (материалы от 27 различных языковых моделей, включая LLaMA, OpenAI и др.) и корпуса НСЗ [38].

Для более детального анализа собранных текстов были подсчитаны ста-

тистики по представленным частям речи в используемых текстах. Для сбора информации о частях речи в русском языке использовалась библиотека обработки текстовых последовательностей Pytymorphy, для английского — NLTK. На Рис. 2.2 показано их соотношение для русского и английского языков. Статистические значения у искусственных и настоящих текстов близкие, но имеют различия.

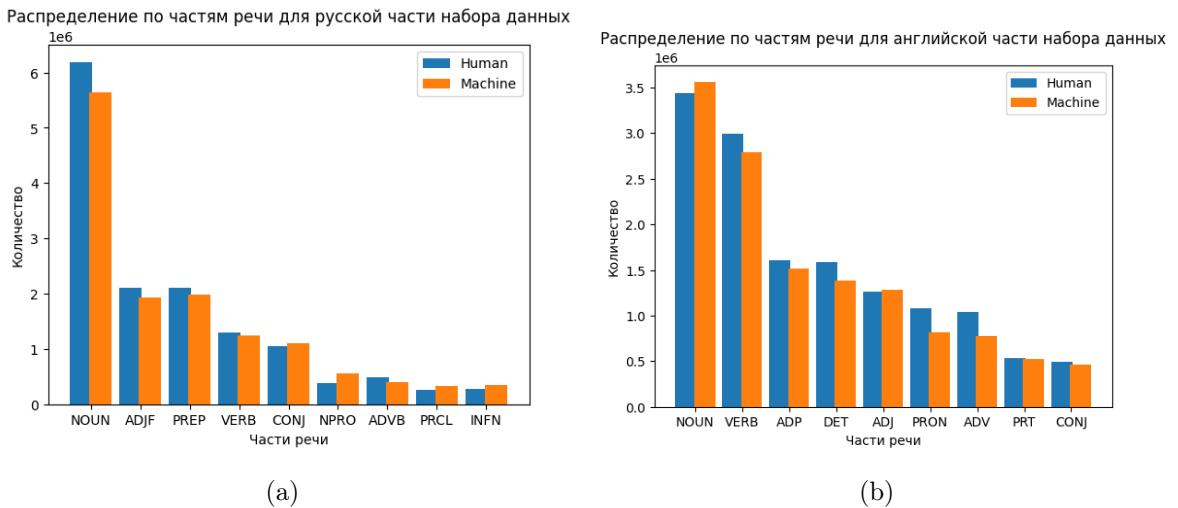


Рис. 2.2: На рисунке (а) представлено количественное разбиение текстов русской части данных по частям речи, а на рисунке (б) для английского. Здесь NOUN — существительное, ADJ(F) — прилагательное (полное), PREP — предлог, VERB — глагол, CONJ — союз, NPRO — местоимение, ADV(B) — наречие, PRCL — частица, INFN — инфинитив, ADP — дополнения (предлоги), DET — артикли, числительные, PRON — местоимения, PRT — частицы.

### 2.3. Детекция фрагментов варьируемой длины

Предложенный ранее алгоритм фрагментирования и детекции текстовых последовательностей имеет высокое качество работы, однако обладает существенным минусом — фиксированная длина каждого текстового фрагмента из документа. С прогрессом больших языковых моделей и их вхождению в привычный обиход, пользователи склонны к нарезке и вставке отдельных сгенерированных фрагментов в документы, нежели всего текста целиком. Это ставит перед сообществом исследователей более утонченную задачу, ведь отныне необходимо детектировать фрагмент варьируемой длины. Для решения задачи детекции с варьируемой длиной распознаваемого фрагмента можно сделаем переход от классификации фрагментов к токенам.

Пусть  $\mathbf{V}$  — словарь токенов, используемый предобученной языковой моделью. Определим множество текстовых последовательностей, где каждая после-

довательность представлена в виде конечной комбинации токенов из  $\mathbf{V}$ :

$$\mathbb{D} = \left\{ D^i = [t_k^i]_{k=1}^{n_i} \mid t_k^i \in \mathbf{V}, n_i \in \mathbb{N} \right\},$$

где  $D^i$  — текстовая последовательность (например, документ или его фрагмент),  $n_i$  — количество токенов в  $i$ -м примере.

Рассмотрим обучающую выборку из  $N$  размеченных примеров:

$$\mathbf{D} = \{(D^i, \mathbf{c}^i)\}_{i=1}^N,$$

где  $\mathbf{c}^i = [c_1^i, \dots, c_{n_i}^i]$  — бинарные метки на уровне токенов:  $c_k^i = 1$ , если токен  $t_k^i$  считается сгенерированным, и  $c_k^i = 0$ , если написан человеком.

Цель состоит в построении функции

$$\hat{g} : \mathbf{V}^* \rightarrow [0, 1]^n,$$

которая по входной токенизированной последовательности возвращает вероятности принадлежности токенов к искусственно сгенерированным.

Для обучения модели минимизируем эмпирический риск на всей выборке:

$$\hat{g} = \operatorname{argmin}_{g \in \mathfrak{F}} \sum_{i=1}^N \sum_{k=1}^{n_i} \mathbb{I}[g(t_k^i) \neq c_k^i],$$

где  $g \in \mathfrak{F}$  — множество допустимых алгоритмов классификации на уровне токенов.

В качестве функции потерь используется бинарная кросс-энтропия:

$$\mathcal{L}_{\text{BCE}}(g, \mathbf{D}) = -\frac{1}{\sum_i n_i} \sum_{i=1}^N \sum_{k=1}^{n_i} [c_k^i \cdot \log(g(t_k^i)) + (1 - c_k^i) \cdot \log(1 - g(t_k^i))].$$

Для оценки качества классификации на уровне токенов используются метрики *precision*, *recall*, и  $F_1$ -score.

В данной работе предлагаемый метод сосредоточен на использовании архитектуры многозадачного обучения с двумя головами. Применение такого подхода оправдано спецификой задачи, в которой классовые диапазоны непрерывны на протяжении нескольких сотен символов. Были проанализированы различные варианты кодировщика для получения вектора состояния для каждого токена в последовательности, а также варианты разбиения фрагментов на токены для дальнейшей подачи на вход кодировщика, основанного на архитектуре трансформер. Задача состоит в том, чтобы создать систему обнаружения, устойчивую к сгенерированным текстовым фрагментам из принципиально разных способов генерации и разнообразных научных доменов.

**Анализ набора данных.** Исследуемый набор данных для решения данной задачи содержит примеры на английском языке в количестве 5000 образцов.

Text	Annotations	Tokens	Token Label Ids
The number of osteoporotic fractures ...	[[0, 'chatgpt'], ...]	['The', 'number', 'of', 'osteoporotic', 'fractures', ...]	[2, 2, 2, 2, 2, 2, ...]
Blade surface roughness ranks amongst ...	[[0, 'human'], ...]	['Blade', 'surface', 'roughness', 'ranks', 'amongst', 'the', ...]	[0, 0, 0, 0, 0, 0, ...]

Таблица 2.1: Пара примеров представления строк в предоставленном наборе данных. Каждый пример представлен текстовой строкой, ее разметкой, разбиением на токены и метками для каждого токена.

Каждый образец представлен текстом, аннотациями, токенами соответствующего текста и метками для каждого токена. Примеры строк из набора данных можно увидеть в таблице 2.1. Все тексты в строках набора данных — документы из научной области, в которых некоторые части были заменены машинно-сгенерированными последовательностями. Сгенерированный текст может начинаться даже с середины предложения, поэтому при классификации важно учитывать контекст. Набор данных содержит научные тексты, токены которых должны быть отнесены к одному из четырех классов: человеческие, сгенерированные генеративной моделью ChatGPT, сгенерированные с использованием синонимов из библиотеки работы с текстовыми последовательностями NLTK и тексты, сгенерированные с использованием неназванной модели суммаризации.

### Мультиголовой классификатор токенов.

В данной работе предлагается мультиголовая модель, состоящая из слоев кодировщика и двух классификаторов для каждого токена. Классификаторы обучаются совместно и позволяют модели достигать более высокой способности к обобщению и скорость обработки текстовых последовательностей, поступающих на вход. Интервалы, которые предлагается обнаружить, не фиксируются заранее и должны быть помечены автоматически алгоритмом обнаружения на основе изученного контекста.

Одна из задач, в которой широко применяется формулировка в терминах классификации токенов, — это исправление грамматических ошибок (GEC) [50]. Цель данной задачи заключается в максимально точном выявлении и исправлении ошибок в тексте, что требует высокой точности в определении границ фрагментов, подлежащих корректировке. Подобные требования актуальны и в рамках поставленной в данной работе задачи, где переход между классами на уровне токенов может происходить в произвольной позиции, а сами интервалы могут иметь произвольную длину.

Одним из современных и эффективных решений задачи GEC является архитектура GECToR [51], использующая мультиголовой трансформер и подход к

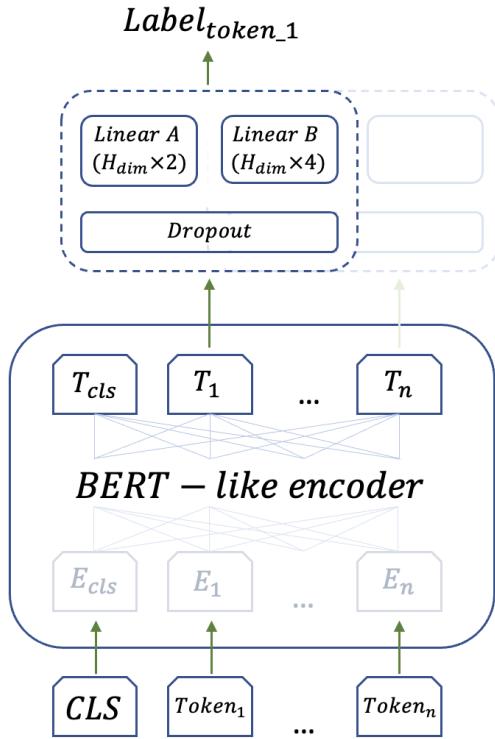


Рис. 2.3: Многозадачная архитектура для решения задачи классификации токенов. Каждый токен, после получения векторного представления текста с помощью кодировщика, классифицируется с помощью двух голов: линейная А — бинарная классификация, линейная В — мультиклассовая классификация.

предсказанию меток для каждого токена. В настоящем исследовании за основу была взята аналогичная архитектура, адаптированная для задачи обнаружения машинно-сгенерированных фрагментов текста на уровне токенов. Структура предложенного решения представлена на рисунке 2.3.

Последовательность токенов, поступающая на вход алгоритму, векторизуется с помощью BERT-подобного энкодера, затем вектор для каждого токена параллельно подается на вход двух линейных слоев с установленной регуляризацией в виде dropout-слоя. Основное отличие от стандартного BERT-подобного подхода для этой задачи заключается в наличии двух классификационных голов и их обучении в многозадачном режиме. Одна голова (линейная А) отвечает за бинарную классификацию — текст человеческий или искусственный, а вторая (линейная В) решает задачу мультиклассовой классификации для 4 классов, которые были заданы в наборе данных, предназначенном для этой задачи. Таким образом, при обучении такой сети функция потерь агрегирует значения ошибок от обеих голов и суммирует их.

Ранее проведённые исследования показывают, что подход многозадачного обучения позволяет снизить вероятность переобучения и повысить обобщающую способность обученной модели [52]. В процессе инференса для каждого токена входной последовательности рассчитывается вероятность принадлеж-

ности к машинно-сгенерированному классу с помощью слоя *Linear A* и функции softmax. Если максимальное значение вероятности среди всех токенов последовательности превышает заранее заданный порог, то для всей последовательности применяется мультиклассовая классификация на уровне токенов с использованием слоя *Linear B*. В противном случае всей последовательности присваивается метка класса человеческого текста.

Данный подход к инференсу отражает специфику решаемой задачи. Как показано в предыдущем разделе, машинно-сгенерированные вставки, как правило, не проявляются в виде единичных токенов, а имеют форму непрерывных фрагментов значительной длины. Кроме того, тексты, написанные человеком, чаще имеют большую длину по сравнению с остальными классами. В условиях ограничения на максимальную длину последовательности, подаваемой на вход энкодера BERT-подобной архитектуры, это означает, что большинство таких последовательностей будет содержать токены одного и того же класса.

## 2.4. Анализ качества детекции фрагментов документов

Проводится серия вычислительных экспериментов для анализа предложенных методов детекции искусственно-сгенерированных фрагментов внутри документов при помощи решения задачи классификации последовательностей и токенов.

**Классификация фиксированных фрагментов.** В рамках вычислительного эксперимента предлагается работа с текстами, средняя длина которых приблизительно равна длине потенциального фрагмента из описанной ранее суперпозиции преобразований, наборов с такими данными преобладающее количество среди опубликованных. Помимо этого, акцент был сделан на текстах, относящихся к различным тематикам.

Для построения качественного классификатора текстовых фрагментов в данной работе была проведена серия экспериментов с дообучением моделей на основе архитектуры трансформеров. В качестве базовой модели используется мультиязычный энкодер XLM-RoBERTa. Помимо базового варианта, реализованы пять дополнительных подходов:

### 1. Добавление ручных признаков.

Кодировщики на базе трансформеров формируют векторные представления текста (например, размерности 768, используя [CLS]-токен). В данном эксперименте к ним добавляются вручную извлечённые статистические признаки (всего 26), формирующие дополнительное признаковое пространство. Совмещенный вектор подаётся на вход классификатору.

### 2. Мультиязычное дообучение.

Исследуется влияние объединения данных на разных языках на итоговое качество модели. Обучение проводится на смешанной выборке при использовании модели XLM-RoBERTa.

### **3. Перевод текстов.**

Проверяется гипотеза о том, что переводные тексты улучшают обобщающую способность классификатора. С помощью сервиса Google Translate подготовлены версии обучающих выборок, в которых 10%, 25% и 50% сгенерированных текстов заменены на переводы с другого языка.

### **4. Перефразирование.**

С учётом восприимчивости трансформерных моделей к атакующим переформулировкам, проведён эксперимент с добавлением перефразированных фрагментов в обучающие данные. Были созданы две выборки:

- первая включает тексты, в которых от 0% до 100% предложений перефразированы;
- во второй — от 50% до 100% предложений перефразированы в половине сгенерированных текстов.

Для русского языка используется модель `rut5-base-paraphraser`, для английского — T5 Sentence Paraphraser.

### **5. Применение моделей Т5.**

Рассматриваются два сценария:

- использование полной энкодер-декодер модели Т5, где задача формулируется как генерация метки `human` или `machine`;
- использование только энкодера модели Т5 с добавлением нейросетевого классификатора, что приближено к классической схеме дообучения BERT-подобных моделей.

Выборки для тестирования остаются зафиксированными во всех описанных выше экспериментах и не отличаются от изначально сформированных. Во всех экспериментах дообучение было произведено на 2 эпохах, с размером батча равным 16 и с темпом обучения —  $4e-5$ . При токенизации применялась обрезка фрагментов по максимальной длине входной последовательности 512 токенов, в силу ограничений используемой трансформерной модели.

В Таблице 2.2 приведены результаты поставленных экспериментов с различными техниками дообучения модели с архитектурой трансформер. Результаты эксперимента схожи для обоих заявленных языков. В случае русского языка лучшие качество достигается в рамках подхода с перефразированием каждого текста в сгенерированной части собранного набора данных с диапазоном покрытия 0-100% предложений, для английского — замена половины сгенерированной части собранного набора данных на переводы текстов с русского языка. Заметим, что оба подхода решают задачу с потенциальными состязательными атаками, проводимыми над моделью классификации. Результатом исследования показано, что переведенные и перефразированные тексты позволяют повысить качество и обобщающую способность модели детекции машинно-сгенерированных фрагментов.

**Классификация варьируемых фрагментов.** Модели на основе архитектуры BERT имеют ограничение на максимальную длину входной последовательности. В предыдущих главах диссертации было проанализировано влияние

Таблица 2.2: Сводная таблица результатов вычислительного эксперимента.

Язык	Эксперимент	F1-score	Precision	Recall
ru	базовое решение	0,955	0,958	0,955
	ручные признаки	0,960	0,962	0,959
	мультязычное обучение	0,964	0,964	0,966
	перевод текстов 10%	0,955	0,958	0,955
	перевод текстов 25%	0,958	0,961	0,958
	перевод текстов 50%	0,966	0,968	0,966
	парафраз предложений 100%	<b>0,968</b>	<b>0,970</b>	<b>0,968</b>
	парафраз предложений 50%	0,964	0,965	0,963
	T5 энкодер	0,953	0,956	0,952
	T5 полная модель	0,952	0,954	0,954
en	базовое решение	0,796	0,855	0,802
	ручные признаки	0,801	0,856	0,807
	мультязычное обучение	0,823	0,867	0,828
	перевод текстов 10%	0,812	0,859	0,815
	перевод текстов 25%	0,821	0,865	0,826
	перевод текстов 50%	<b>0,825</b>	<b>0,868</b>	<b>0,830</b>
	парафраз предложений 100%	0,822	0,866	0,827
	парафраз предложений 50%	0,816	0,862	0,817
	T5 энкодер	0,795	0,854	0,801
	T5 полная модель	0,787	0,850	0,794

длины входа на качество обнаружения. В текущей задаче каждый пример сопровождается разметкой в виде разделения на словесные токены. Однако, в зависимости от выбора токенизатора, эти слова могут быть разбиты на разное количество подсловных токенов для подачи на вход модели. Например, словарь токенизатора, адаптированного под научную область — SciBERT [53] — содержит большое количество цельных словоформ, что минимизирует их дальнейшее разбиение на более мелкие токены. Учитывая этот факт, необходимо удостовериться, что после токенизации выбранным токенизатором последовательность, поступающая на вход модели, не обрезается и не теряет важную информацию, полезную для обучения алгоритмов.

Базовый результат, полученный с использованием модели SciBERT, показал высокое качество. В настоящее время существует большое количество предобученных энкодеров, которые способны демонстрировать хорошее качество решения задачи «из коробки». Однако дополнительное дообучение на специализированной предметной области даёт различный эффект в зависимости от выбранной модели, что и послужило мотивацией для экспериментов с заменой энкодеров.

Опираясь на опыт работы [51], в котором была предложена архитектура GECToR, в данной работе также исследовалась модель XLNet [54]. Кроме того, был опробован подход с использованием адаптера QLoRa [55], применённо-

BERT-like encoder + tokens partition length	Development Set		
	P	R	$F_1$
DistilBERT + 512 (baseline)	0.86	0.83	0.84
SciBERT + 512 (baseline)	0.89	0.86	0.87
Mistral w. QLoRA + 512	0.93	0.89	0.91
XLNet + 512	0.91	0.92	0.91
XLNet + 350	0.96	0.95	0.95
SciBERT + 185	0.92	0.89	0.91
SciBERT + 300	0.94	0.93	0.93
SciBERT + 350	<b>0.97</b>	<b>0.96</b>	<b>0.96</b>
SciBERT + 400	0.93	<b>0.96</b>	0.94
SciBERT + 512	0.89	0.93	0.9

Таблица 2.3: Результаты на валидационном наборе данных с различными настройками для предложенной многозадачной архитектуры.

го к обучению большой языковой модели Mistral [5] по задаче классификации токенов. Согласно результатам бенчмарков, данная языковая модель способна решать широкий спектр задач с высоким качеством. В ряде случаев дообучение LLM с адаптером под конкретную предметную область может обеспечивать качество, сопоставимое или превосходящее устоявшиеся решения на базе энкодеров, таких как RoBERTa [56], DeBERTa [57] и др.

Результаты проведённых экспериментов представлены в таблице 2.3. Специфика задачи и предметной области (научные тексты) позволила использовать модель SciBERT в качестве основного энкодера в итоговой архитектуре. Однако, как показано в таблице, наилучшее качество достигается при разбиении текстов на интервалы длиной 350 словесных токенов. Учитывая ограниченный входной контекст трансформерных моделей, такой размер обеспечивает минимальные потери информации после токенизации и, как следствие, наибольшую точность.

Также была подобрана оптимальная величина регуляризации dropout-слоя, которая после серии экспериментов была установлена равной 0.7. Кроме того, было замечено, что при использовании порогового значения 0.55 наблюдается рост качества при отбраковке фрагментов, написанных человеком, во время инференса. Описанная ранее архитектура с двумя линейными классификаторами в сочетании с указанными параметрами позволила добиться среднего значения метрики macro  $F_1$ , равного 0.96, на валидационной выборке. На тестовой выборке результаты оказались сопоставимыми со средним значением метрики 0.96.

В данной главе рассмотрены варианты решения задачи поиска искусственно сгенерированных фрагментов в текстовых последовательностях. Предложен подход детекции в домене научных работ, а именно суперпозиция: разбиение исходного документа по разделам, фрагментация наиболее значимых и после-

дующая их подача на вход дообученному классификатору, основанному на архитектуре трансформер. На выходе классификатора фрагменты одного текста проходят процедуру поправок на множественное тестирование, что позволяет уменьшить систематическое накапливание ложноположительных срабатываний алгоритма. Проведены эксперименты с конкатенацией вручную собранных признаков с выходом нейросетевого кодировщика, дообучением энкодер-декодер модели, мультиязычным обучением и интеграцией в сгенерированные части наборов данных фрагментов с переводами и перефразированием. Помимо этого, предложен подход детекции фрагментов варьируемой длины в домене научных работ. Подход основывается на решении задачи классификации токенов и использовании многозадачного обучения с мультиголовой трансформерной архитектурой. Метод позволяет достичь высокого качества обнаружения для сгенерированных последовательностей произвольной длины.

## Глава 3

### Агрегация и использование смеси моделей для детекции автора

Раздел посвящён объединению методов детектирования машинно-генерированных текстов. Рассматриваются подходы, основанные на дообучении моделей трансформерной архитектуры, а также их последующем ансамблировании и агрегации. В данной главе предлагается стратегия построения дискриминатора, финальное решение которого формируется на основе предсказаний нескольких обученных моделей. Особое внимание уделяется оптимизации задачи детекции посредством обучения адаптеров LoRA для решения задачи бинарной классификации. Предложен подход дообучения компактных адаптеров, содержащих существенно меньше параметров, с учётом доменной и мультиязычной специфики данных. Кроме того, исследуется возможность применения адаптеров не только в рамках BERT-подобных моделей, но и для адаптации больших языковых моделей в условиях мультиязычной постановки задачи.

В данной главе введены предположения, лежащие в основе обучения нескольких адаптеров LoRA, направленных на охват широкого спектра доменов и распределений текстовых генеративных моделей. Поскольку каждая большая языковая модель характеризуется собственным распределением выходных данных, обучение набора адаптеров, специализированных на различных семействах моделей, позволяет учитывать разнообразие потенциальных источников генерации фрагментов. В рамках этих предположений проводится анализ эффективности дискриминатора, чьё итоговое решение основывается на механизме голосования между адаптерами при выполнении бинарной классификации [12].

Дополнительно рассматривается подход агрегации моделей: объединение выходов BERT-подобных энкодеров и адаптера, встроенного в большую языковую модель, после этапа их индивидуального дообучения. Данный подход позволяет повысить качество детекции машинно-генерированных текстов в многоязычной среде. BERT-подобные модели обеспечивают более точное представление фрагментов на малоресурсных языках, тогда как адаптер поверх большой языковой модели, которая в свою очередь обладает более глубоким пониманием языка, способен повысить обобщающую способность системы и улучшить результаты классификации в условиях языкового разнообразия [58].

В вычислительном эксперименте проведено сравнение конфигураций предложенных методов для наборов данных с текстами различных доменов и языков. В подходе оценки способов агрегации используются выборки PAN и IberAuTeXTification. Эксперименты включают примеры на английском, испанском, португальском, каталонском, галлего и эускера языках.

### 3.1. Доменная адаптация при помощи легковесных адаптеров

Рассматривается введенная в главе 1.1 постановка классификации автора документа. С развитием больших языковых моделей увеличивается количество сгенерированных текстов, где новые фрагменты зачастую существенно отличаются по распределению от ранее наблюдаемых. Совершенствование моделей генерации приводит к появлению текстов, обладающих иными языковыми паттернами, что требует регулярного обновления методов детекции. В ранее описанных подходах, включая предложенные модификации, основным методом оставалось дообучение предобученного трансформерного кодировщика. Хотя описанный подход демонстрирует высокую точность, его применение ограничено из-за проблем с масштабируемостью. По мере расширения разнообразия анализируемых распределений возрастает объем данных необходимый для обучения одной модели. Это приводит к ухудшению производительности и повышает риск возникновения «катастрофического забывания» — эффекта, при котором модель теряет способность воспроизводить ранее выученные знания. Если обратить внимание на архитектуру современных больших языковых моделей, можно понять, что для решения задачи детекции можно дообучать и саму языковую модель. Благодаря существенно большему числу параметров и более глубокому пониманию структуры естественного языка, такие модели обладают способностью выявлять сложные паттерны в текстах и обеспечивать более высокое качество распознавания сгенерированных фрагментов.

Обучение всей совокупности параметров большой языковой модели для адаптации к узкой области представляется избыточным как с точки зрения вычислительных затрат, так и с позиции практической реализации. Полноценное дообучение таких моделей требует значительных ресурсов и времени, что делает данный подход малопригодным в условиях ограниченного доступа к высокопроизводительным серверам.

В качестве альтернативы в ряде современных исследований [59] предлагаются использование легковесных адаптеров, позволяющих значительно сократить объём обновляемых параметров. Одним из наиболее эффективных решений является метод *Low-Rank Adaptation* (LoRA), который используется для дообучения моделей в задачах типа sequence-to-sequence и показывает высокую эффективность при минимальных затратах в плане памяти и вычислений.

Метод LoRA позволяет дообучать только небольшое количество параметров, сохраняя при этом основной корпус модели неизменным. В частности, вместо прямого обновления весов  $\Delta W \in \mathbb{R}^{d \times k}$  обучаемой модели, их изменение аппроксимируется с помощью произведения двух низкоранговых матриц  $A \in \mathbb{R}^{d \times r}$  и  $B \in \mathbb{R}^{r \times k}$ :

$$W_{\text{upd}} = W + AB,$$

где  $W \in \mathbb{R}^{d \times k}$  — изначальные предобученные веса,  $A \sim \mathcal{N}(0, \sigma^2)$  — случайно инициализированная матрица, а  $B = [0]_{r \times k}$  — матрица нулей. Параметр  $r$  пред-

ставляет собой ранг аппроксимации и служит гиперпараметром, позволяющим управлять степенью сжатия.

Таблица 3.1 иллюстрирует различие между классическим дообучением и подходом с использованием LoRA:

Полное дообучение	Дообучение с LoRA
$W_{\text{upd}} = W + \Delta W$	$W_{\text{upd}} = W + AB$
$\hat{y} = xW + x\Delta W$	$\hat{y} = xW + xAB$

Таблица 3.1: Структура обновления весов при использовании адаптера LoRA

Можно сделать вывод, что использование LoRA-адаптеров позволяет для каждого специфического домена создавать отдельный обучаемый модуль с ограниченным числом параметров. Это даёт возможность гибко управлять моделью и её поведением в различных сценариях, не затрагивая основную архитектуру LLM.

**Формирование набора данных.** Для того чтобы получить устойчивый детектор, который будет понимать, где находится конкретный домен и область можно прибегнуть к использованию адаптеров, а именно — обучить под каждый домен свой адаптер.

Для отбора данных использовались исключительно открытые источники, содержащие датасеты, соответствующие различным тематикам. Для реализации идеи обучения нескольких адаптеров собранный корпус был разделён на несколько подмножеств:

- Набор А, содержащий тексты, сгенерированные моделями семейства GPT (GPT-3, GPT-3.5, GPT-4);
- Набор В, включающий тексты, полученные от моделей семейств LLaMA и Mistral (Llama-2, Llama-2, Mistral-v0.1, Mistral-v0.2);
- Набор С, состоящий из ограниченного количества текстов, сгенерированных другими моделями (Vicuna, OPT, BLOOM, Alpaca, Gemini Pro).

**Множественное голосование.** Для обучения использовалась идея адаптации, заключающаяся в повторном использовании одной и той же базовой модели с различными адаптерами, обученными под конкретные задачи. В рамках данного исследования применяется улучшенная версия метода LoRA — QLoRA [55], позволяющая проводить квантование весов предобученной языковой модели до 4 бит. Основной подход заключался в обучении лёгких адаптеров для языковой модели Mistral-v0.2, при этом каждый адаптер обучался на отдельном поднаборе данных, отражающем различное распределение текстов. В результате для одной модели было обучено три независимых адаптера.

На этапе инференса, после обучения адаптеров была проведена процедура агрегирования результатов с целью повышения качества итоговой модели.

Попытки объединения адаптеров путём усреднения весов или слияния параметров не дали прироста метрик. Однако использование ансамбля из предсказаний каждого адаптера позволило достичь высокой точности. Принцип агрегирования следующий: если все три адаптера для конкретного примера предсказывают метку 0 (человеческий текст), то финальное предсказание также 0; в противном случае — метка 1 (машинно-сгенерированный текст). Такой способ агрегирования позволяет значительно повысить точность детекции именно человеческих текстов: если ни один из адаптеров не показывает низкое значение вероятности у класса человек и не выбирает метку генерации, то текст с высокой вероятностью написан человеком.

Использование нескольких адаптеров позволяет повысить обобщающую способность системы, поскольку каждый адаптер фиксирует зависимости, характерные для своего распределения данных. Выбор модели Mistral обусловлен её высокой эффективностью в широком спектре задач [5]. Применение лёгких адаптеров также позволяет избежать полного переобучения модели, сохраняя при этом её знания. В случае появления новых семейств языковых моделей возможна простая интеграция дополнительных адаптеров без необходимости переобучения всей архитектуры, что обеспечивает модульность и масштабируемость подхода.

### 3.2. Агрегация моделей для покрытия языкового разнообразия

Решение задачи детекции сгенерированных фрагментов в условии языкового многообразия требует учета ограничений, присущих ранее рассмотренным в данной работе подходам. При решении поставленной задачи с моделями, основанными на архитектуре трансформер, выбор предобученных моделей оказывается ограниченным, поскольку количество высококачественных доступных мультиязычных кодировщиков значительно меньше, чем моноязычных. Кроме того, мультиязычные модели часто предобучаются на несбалансированных объемах данных для различных языков, что может привести к недостаточной точности представления текстов в векторном пространстве модели. Это способствует искажению семантических связей, ведь тексты с различной смысловой нагрузкой могут находиться близко друг к другу в пространстве модели, что негативно может влиять на результат последующего дообучения.

В рамках настоящего исследования использовался набор данных, охватывающий шесть языков: английский, испанский, португальский, каталонский, галлего и баскский. Анализ предобученных трансформерных моделей выявил, что их эффективность существенно различается при обработке низкоресурсных и высокоресурсных языков. Это наблюдение послужило основой для разработки подхода, основанного на голосовании смеси моделей, адаптированных под различные языковые группы, представленные в датасете.

Для реализации смеси моделей были отобраны модели для языковой группы высокоресурсных языков (английского, испанского и португальского), а так-

же для оставшихся низкоресурсных. Для первой группы была выбрана модель Multilingual E5, а для второй — XLM-RoBERTa соответственно. Кроме того, предыдущие исследования показали, что большая языковая модель BLOOM [60] обладает хорошим качеством признакового описания для различных языков, пять из шести языков, использованных в задаче, входили в число поддерживаемых BLOOM. Примечательным оказалось то, что язык галлего, не входящий в список явно поддерживаемых BLOOM, показал сопоставимые значения ключевых метрик даже без специальной настройки. Это указывает на потенциал модели к обобщению на языки, не включенные в обучающую выборку, а также высокий уровень распознавания паттернов различных языков.

В результате многоэтапного тестирования различных стратегий обучения был выбран подход к дообучению легковесного адаптера с использованием метода параметрической эффективной настройки (PEFT). Модель BLOOM загружалась в 4-битном формате с применением двойного квантования для снижения вычислительной нагрузки. Оптимальными параметрами адаптера оказались: размер матрицы бокового обновления  $r=64$  и коэффициент шкалирования  $\alpha=16$ . Адаптация проводилась для матриц запросов, ключей и значений в слоях трансформера.

Анализ различных методов агрегирования предсказаний моделей показал, что наиболее эффективным является подход, ориентированный на класс «человек». Для каждого тестового образца сначала определялся язык текста: если он относился к высокоресурсным (английский, испанский, португальский), учитывались предсказания моделей BLOOM и Multilingual E5, в свою очередь для низкоресурсных языков (каталонский, галлего, баскский) использовались результаты BLOOM и XLM-RoBERTa. Итоговая метка формировалась по принципу, если обе модели указывали на принадлежность текста к классу «человек», финальная метка также устанавливалась как «человек». В остальных случаях текст классифицировался как «сгенерированный».

### 3.3. Анализ смеси моделей для детекции фрагментов

Проводится серия вычислительных экспериментов для анализа предложенных методов детекции искусственно-сгенерированных фрагментов в постановке различных доменов и многоязычного разнообразия при помощи решения задачи классификации методами агрегации выходов моделей.

**Доменная адаптация.** В таблице 3.2 отражен результат работы метода с дообучением легковесных адаптеров под различные домены, а также его сравнение с иными подходами, предложенными исследователями области ранее. Подход, описанный в данной главе, выигрывает по качеству практически у всех базовых моделей, за исключением базовой модели Binoculars. Стоит также отметить, что предложенный метод не ограничен ресурсами, поскольку доступно обучение любого количества адаптеров для разных семейств данных, генерируемых моделями.

Таблица 3.2: Результаты предложенного метода с сравнением с иными методами детекции. Отражены метрики качества ROC-AUC, Brier, C@1, F<sub>1</sub>, F<sub>0.5u</sub>, а также их среднее значение.

Подход	ROC-AUC	Brier	C@1	F <sub>1</sub>	F <sub>0.5u</sub>	Mean
Предложенный в главе	0.967	0.955	0.965	0.939	0.943	0.954
Baseline Binoculars	0.972	0.957	0.966	0.964	0.965	0.965
Baseline Fast-DetectGPT (Mistral)	0.876	0.8	0.886	0.883	0.883	0.866
Baseline PPMd	0.795	0.798	0.754	0.753	0.749	0.77
Baseline Unmasking	0.697	0.774	0.691	0.658	0.666	0.697
Baseline Fast-DetectGPT	0.668	0.776	0.695	0.69	0.691	0.704
95-th quantile	0.994	0.987	0.989	0.989	0.989	0.990
75-th quantile	0.969	0.925	0.950	0.933	0.939	0.941
Median	0.909	0.890	0.887	0.871	0.867	0.889
25-th quantile	0.701	0.768	0.683	0.657	0.670	0.689
Min	0.131	0.265	0.005	0.006	0.007	0.224

Адаптеры, реализованные в рамках метода QLoRA, обеспечивают эффективное моделирование распределения данных, что позволяет повысить точность классификации текстовых последовательностей. Использование адаптеров позволяет адаптировать предобученные языковые модели к специфическим доменам без необходимости повторного обучения всей архитектуры, сохраняя вычислительную эффективность. В ходе экспериментов были протестированы различные стратегии агрегирования предсказаний, однако окончательный выбор был сделан в пользу логического оператора ИЛИ (OR). Итоговая классификация возвращает нулевой класс («человеческий текст») исключительно в случае, если все обученные адаптеры сформировали такое предсказание. Предложенный метод отличается высокой вычислительной эффективностью и масштабируемостью, что подтверждается стабильной производительностью как по времени работы, так и по ресурсоемкости.

**Агрегация моделей.** Результаты, полученные в ходе эксперимента с языковым многообразием на тестовых данных, представлены в таблице 3.3. В качестве метрики был выбран F<sub>1</sub>-score. Агрегация ответов улучшила пространство ответов модели и положительно повлияла на повышение качества. Ответы BLOOM, агрегированные с ответами XLM-RoBERTa и Multilingual E5, показали самые высокие результаты по выбранной метрике на очищенных данных. Помимо этого, в таблице 3.4 отражена разность показателей качества для различных кодировщиков и языков.

В работе предлагается модель для детекции текста, основанная на агрегировании ответов большой языковой модели BLOOM, которая была предварительно обучена на 5 из 6 представленных языков для обнаружения с использова-

нием легких адаптеров с подходом QLoRa и двух трансформерных кодировщиков XLM-RoBERTa (дообучен на каталонском, галлего, эускера) и Multilingual E5 (дообучен на испанском, английском, португальском). Агрегация ответов улучшила пространство ответов модели и положительно повлияла на повышение качества. Ответы BLOOM, агрегированные с ответами XLM-RoBERTa и Multilingual E5, показали самые высокие результаты по выбранной метрике на очищенных данных.

Модель	Предсказания BLOOM	
	<i>Без агрегации</i>	<i>С агрегацией</i>
XLM-RoBERTa	94.38	94.75
Multi E5	94.38	95.15
XLM-RoBERTa and Multi E5	95.89	<b>96.94</b>

Таблица 3.3: Сравнение метрик моделей на отложенной выборке данных. Столбец «без» означает, что для языков, на которых обучалась модель, используется только указанный в строке ответ модели, а для остальных примеров используется ответ BLOOM. В столбце «с» указаны метрики агрегирования, когда ответ модели сочетается с выходом BLOOM.

Модель	Результат классификации					
	<i>cat<sup>1</sup></i>	<i>eus<sup>2</sup></i>	<i>glg<sup>3</sup></i>	<i>eng<sup>4</sup></i>	<i>spa<sup>4</sup></i>	<i>por<sup>4</sup></i>
XLM-RoBERTa-large	<b>51.69</b>	<b>84.44</b>	<b>74.38</b>	42.46	38.39	38.59
Multilingual-E5-large	51.45	61.87	69.02	<b>53.43</b>	<b>50.85</b>	<b>50.31</b>

Таблица 3.4: Сравнение показателей качества детекции для двух отдельных кодировщиков. Можно заметить, что качество на низкоресурсных и высокоресурсных языках сильно различается.

## Глава 4

### Мультизадачное обучение для задачи детекции

Раздел посвящен методам детектирования машинно-сгенерированных текстовых фрагментов. Исследуется применение подхода мультизадачного обучения в задаче детекции текстовых последовательностей. Предлагается использование мультизадачного подхода для повышения качества целевых метрик, а также регуляризации модели в виде индуктивного смещения через разделение параметров.

В случае наличия некоего количества задач, при помощи подхода, представленного в данной главе, появляется возможность быстрее получить желаемый результат, а в случае одной задачи - улучшить показатели, внеся в модель дополнительные знания. Продемонстрирован способ решения задачи классификации текстовых фрагментов при помощи мультизадачной модели, основанной на архитектуре трансформеров. Отражено формирование кластерной структуры в векторном пространстве модели при проведении мультизадачного этапа обучения. Исследовано влияние добавления задач в общую мультизадачную архитектуру на качество целевых метрик [61, 62]

В главе аналитически доказывается снижение сложности модели по Радемахеру при использовании подхода мультизадачного обучения с архитектурой трансформер для решения задачи классификации. Результат актуален как для подхода с одной целевой задачей, так и для нескольких одновременно.

Для анализа качества работы предложенных методов детекции искусственно-сгенерированных фрагментов проводилась серия вычислительных экспериментов. Сформирован ряд наборов данных, включающих примеры с генерацией от различных текстовых генеративных моделей, включающий чередование методов сэмплирования и уровня сложности модели. Проводилось дообучение предобученных трансформерных моделей на собранных наборах данных. Помимо этого тестирование проводилось на наборе данных из открытого доступа Coling [63].

#### 4.1. Постановка подхода мультизадачного обучения

Пусть  $\mathbf{W}$  — алфавит, содержащий минимальные элементы текстовых последовательностей (символы). Определим множество текстовых последовательностей:

$$\mathbb{D} = \left\{ \left[ t_j \right]_{j=1}^n \mid t_j \in \mathbf{W}, n \in \mathbb{N} \right\}.$$

Для  $M$  задач классификации задано множество датасетов  $\mathbb{D} = \{d_1, d_2, \dots, d_M\}$ , где  $d_i$  соответствует  $i$ -й задаче.

Архитектура модели мультизадачного обучения (MTL) с HPS-архитектурой состоит из:

- Общей подсети  $h_{\theta_s}$  с параметрами  $\theta_s$ , выделяющей общие признаки для всех задач;
- $T$  специфичных подсетей  $g_{\theta_1}, \dots, g_{\theta_T}$  с параметрами  $\{\theta_i\}$ , решающих индивидуальные задачи.

Общая параметризация модели:

$$\theta = \theta_s \cup \bigcup_{i=1}^T \theta_i.$$

**Преобразование для мультизадачного обучения.** Для каждой задачи  $t \in [T]$  определим отображение:

$$\mathbf{g}_t : \mathbb{D} \rightarrow \mathbf{C}_t,$$

где  $\mathbf{C}_t$  — множество меток  $t$ -й задачи ( $|\mathbf{C}_t| \geq 2$ ).

**Целевая функция.** Задача заключается в минимизации суммарной функции потерь по всем задачам и примерам датасета:

$$\hat{\theta} = \operatorname{argmin}_{\theta} \sum_{x_j \in \mathbb{D}} \sum_{t=1}^T L_t \left( g_{\theta_t} \circ h_{\theta_s}(x_j), c_t^j \right),$$

где  $L_t$  — функция потерь для  $t$ -й задачи,  $c_t^j$  — истинная метка  $t$ -й задачи для примера  $x_j$ .

**Функции потерь.** Для бинарной классификации ( $\mathbf{C}_t = \{0, 1\}$ ) используется бинарная кросс-энтропия:

$$\mathcal{L}_{\text{BCE}}^{(t)}(\theta, \mathbb{D}) = -\frac{1}{|\mathbb{D}|} \sum_{x_j \in \mathbb{D}} \left[ c_t^j \cdot \log(g_{\theta_t} \circ h_{\theta_s}(x_j)) + (1 - c_t^j) \cdot \log(1 - g_{\theta_t} \circ h_{\theta_s}(x_j)) \right].$$

Для многоклассовой классификации ( $|\mathbf{C}_t| > 2$ ) применяется кросс-энтропия:

$$\mathcal{L}_{\text{CE}}^{(t)}(\theta, \mathbb{D}) = -\frac{1}{|\mathbb{D}|} \sum_{x_j \in \mathbb{D}} \sum_{k=1}^{|\mathbf{C}_t|} c_{t,k}^j \cdot \log(\text{softmax}_k(g_{\theta_t} \circ h_{\theta_s}(x_j))).$$

## 4.2. Мультизадачное обучение с архитектурой трансформер

Задача детекции машинно-сгенерированных текстов обычно формулируется как задача бинарной классификации текстов. Наиболее распространёнными подходами к её решению являются дообучение моделей на основе трансформеров [23] или использование zero-shot методов, опирающихся на внутренние статистические характеристики текста. Эти методы демонстрируют высокую эффективность в рамках задач, ограниченных исходным доменом, они не обладают устойчивостью к изменениям домена, модели генератора или языка текстов

[56]. Между тем, в реальных сценариях обнаружения машинно-сгенерированного контента такие изменения, напротив, представляют собой более актуальную постановку задачи.

Кроме того, зачастую, данные, которые используются для решения задачи детекции, могут содержать шум или быть низкокачественными, что дополнительно усложняет проблему [64]. Ключевой задачей является разработка модели, устойчивой к наличию зашумлённых и низкокачественных данных, а также способной адаптироваться к новым доменам и языкам.

Высокая когерентность и качество текстов, генерируемых современными языковыми моделями, затрудняет выделение качественных признаков, по которым можно было бы построить разделяющую гиперплоскость в пространстве текстовых последовательностей для разделения машинно-генерированных и человеческих текстов. Одним из способов улучшения представлений, получаемых однозадачными архитектурами, является применение многозадачного обучения (MTL) [61].

В данной главе предлагается метод отражающий, что с использованием дополнительного анализа внутренних данных и выравнивания эмбеддингов в рамках MTL возможно достичь высокой эффективности обнаружения фрагментов в условиях кросс-доменных и кросс-генераторных сценариев для текстов, сгенерированных продвинутыми LLM. За счет формирования внимания модели на различных доменах удалось сформировать кластерную структуру представлений текстов в векторном пространстве. Помимо этого, многозадачное обучение предполагает совместное обучение на нескольких задачах, что позволяет повысить качество решения целевой задачи за счет выделения общих информативных паттернов между задачами. Такой подход эффективен, если задачи связаны между собой, а также может быть полезен даже для одной задачи, если имеются дополнительные данные, позволяющие улучшить представление текста в векторном пространстве.

**Оценка обобщающей способности модели.** Для анализа способности модели аппроксимировать истинное распределение данных и оценки её обобщающей способности используется изучение границ обобщающей ошибки. Эти границы определяют доверительные интервалы для истинного риска модели. Рассмотрим формальную постановку: пусть заданы  $n$  независимых и одинаково распределенных (i.i.d.) примеров  $\{(X_1, Y_1), \dots, (X_n, Y_n)\}$ , сгенерированных из неизвестного распределения  $P$ . Алгоритм обучения строит функцию  $f : X \rightarrow Y$ , которая минимизирует ожидаемые потери (риск):

$$\mathbb{E}_{(X,Y) \sim P} [\ell(f(X), Y)].$$

Поскольку истинное распределение  $P$  неизвестно, риск оценивается через его эмпирический аналог:

$$\mathcal{L}_{\text{emp}}(f) = \frac{1}{n} \sum_{i=1}^n \ell(f(X_i), Y_i).$$

Типичная форма границы обобщающей ошибки выглядит следующим образом:

$$\mathbb{E}_{(X,Y) \sim P} [\ell(f(X), Y)] \leq \mathcal{L}_{\text{emp}}(f) + h \text{ (сложность класса } F, n \text{)},$$

где  $h$  — функция, зависящая от сложности класса  $F$  и объема выборки  $n$ .

**Сложность Радемахера.** Для количественной оценки сложности класса функций  $F$  используется сложность Радемахера, которая характеризует способность функций из  $F$  коррелировать со случайным шумом. Эта мера позволяет оценить, насколько богатым является множество гипотез  $G$ .

**Определение 1** (Сложность Радемахера). Пусть задан класс гипотез  $G := \{g : Z \rightarrow \mathbb{R}\}$  и выборка  $S := \{z_1, \dots, z_n\}$ , сгенерированная из распределения  $P$ . Эмпирическая сложность Радемахера класса  $G$  определяется как:

$$\widehat{R}(G) := \mathbb{E}_\sigma \left[ \sup_{g \in G} \frac{1}{n} \sum_{i=1}^n \sigma_i g(z_i) \right],$$

где  $\sigma_i$  — независимые случайные величины, равномерно распределенные на множестве  $\{\pm 1\}$ .

Теоретическая сложность Радемахера вычисляется как математическое ожидание эмпирической сложности по всем возможным выборкам объема  $n$ :

$$R(G) := \mathbb{E}_{S \sim P^n} [\widehat{R}(G)].$$

**Мультизадачность с архитектурой трансформеров.** Рассмотрим модель классификации, основанную на архитектуре трансформера, с энкодером и дополнительной головой в случае однозадачного обучения (STL) и общим энкодером и  $T$  головами в случае многозадачного обучения. Для STL предполагается наличие  $nT$  примеров на задачу, для MTL —  $n$  примеров на каждую из  $T$  задач. Каждая задача имеет  $n$  независимых одинаково распределенных (i.i.d.) выборок. Тогда множества гипотез можно задать следующим образом:

$$\begin{aligned} \mathcal{F}_{\text{STL}} &= \left\{ x \mapsto w_{\text{head}}^\top \phi(x; w_{\text{enc}}) \mid w_{\text{enc}} \in \mathcal{W}_{\text{enc}}, w_{\text{head}} \in \mathcal{W}_{\text{head}} \right\}, \\ \mathcal{F}_{\text{MTL}} &= \left\{ \left( x \mapsto w_t^\top \phi(x; w_{\text{shared}}) \right)_{t=1}^T \mid w_{\text{shared}} \in \mathcal{W}_{\text{shared}}, w_t \in \mathcal{W}_{\text{head}} \right\}. \end{aligned}$$

**Теорема 1** (Сравнение сложности Радемахера для MTL и STL). Пусть:

- В подходе STL используется трансформерный энкодер  $\phi(\cdot; w_{\text{enc}})$  с параметрами  $w_{\text{enc}} \in \mathcal{W}_{\text{enc}}$  и линейная голова  $w_{\text{head}} \in \mathcal{W}_{\text{head}}$ , обученная на выборке мощности  $nT$ .

- В подходе MTL используется общий трансформерный энкодер  $\phi(\cdot; w_{\text{shared}})$  с  $w_{\text{shared}} \in \mathcal{W}_{\text{shared}} \subsetneq \mathcal{W}_{\text{enc}}$  и  $T$  линейных голов  $\{w_t\}_{t=1}^T \in \mathcal{W}_{\text{head}}^T$ , каждая из которых обучается на выборке объема  $n$ .

Предположим выполнение следующих условий:

1. *Ограничения на нормы:*

- Энкодер:  $\|w_{enc}\| \leq B_{enc}$ ,  $\|w_{shared}\| \leq B_{shared} \leq \frac{B_{enc}}{\sqrt{T}}$ .
- Головы:  $\|w_{head}\| \leq B_{head}$ ,  $\|w_t\| \leq B_{head} \forall t \in [T]$ .

2. *Ограниченные нормы признаков:* Выход энкодера трансформера удовлетворяет:

$$\|\phi(x; w)\|_2 \leq L \cdot \|w\| \cdot \|x\|_2 \quad (\text{свойство Липшица}),$$

где  $L > 0$  — константа, зависящая от архитектуры энкодера.

3. *Ограниченные входы:* Входные векторы ограничены:  $\|x\|_2 \leq R$  для всех  $x \in \mathcal{X}$ .

Тогда эмпирическая сложность Радемахера для любой задачи  $t$  в MTL удовлетворяет:

$$\widehat{\mathfrak{R}}_{MTL}^{(1)}(n) \leq \widehat{\mathfrak{R}}_{STL}^{(1)}(nT).$$

Кроме того, если  $B_{shared} < \frac{B_{enc}}{\sqrt{T}}$ , неравенство становится строгим:

$$\widehat{\mathfrak{R}}_{MTL}^{(1)}(n) < \widehat{\mathfrak{R}}_{STL}^{(1)}(nT).$$

*Доказательство.* Для энкодера трансформера  $\phi(x; w)$  свойство Липшица гарантирует:

$$\|\phi(x; w)\|_2 \leq L \cdot \|w\| \cdot \|x\|_2 \leq LB_R,$$

где  $B = B_{enc}$  или  $B_{shared}$ .

1. **Сложность STL:** Гипотеза  $f(x) = w_{head}^\top \phi(x; w_{enc})$  имеет выходы, ограниченные:

$$|f(x)| \leq \|w_{head}\| \cdot \|\phi(x; w_{enc})\| \leq B_{head} \cdot LB_{enc}R.$$

Эмпирическая сложность Радемахера для  $nT$  выборок оценивается как:

$$\widehat{\mathfrak{R}}_{STL}^{(1)}(nT) \leq \frac{LB_{enc}B_{head}R}{\sqrt{nT}}.$$

2. **Сложность MTL (на задачу):** Для задачи  $t$  гипотеза  $f_t(x) = w_t^\top \phi(x; w_{shared})$  удовлетворяет:

$$|f_t(x)| \leq B_{head} \cdot LB_{shared}R.$$

С  $n$  выборками на задачу:

$$\widehat{\mathfrak{R}}_{MTL}^{(t)}(n) \leq \frac{LB_{shared}B_{head}R}{\sqrt{n}}.$$

3. Подставим  $B_{\text{shared}} \leq \frac{B_{\text{enc}}}{\sqrt{T}}$  в MTL:

$$\widehat{\mathfrak{R}}_{\text{MTL}}^{(t)}(n) \leq \frac{LB_{\text{head}}R}{\sqrt{n}} \cdot \frac{B_{\text{enc}}}{\sqrt{T}} = \frac{LB_{\text{enc}}B_{\text{head}}R}{\sqrt{nT}} = \widehat{\mathfrak{R}}_{\text{STL}}^{(1)}(nT).$$

Если  $B_{\text{shared}} < \frac{B_{\text{enc}}}{\sqrt{T}}$ , неравенство становится строгим:

$$\widehat{\mathfrak{R}}_{\text{MTL}}^{(1)}(n) < \widehat{\mathfrak{R}}_{\text{STL}}^{(1)}(nT).$$

□

**Набор данных.** В данной работе для проведения экспериментов и отражения результатов был выбран набор данных COLING. Набор данных, помимо методов бинарной классификации, содержит также дополнительную информацию о текстах, такую как модель генерации, источник примера (его исходный набор данных) и домен. Эта информация поможет модели мультизадачного обучения вносить поправки в отображение текстовых примеров в векторном пространстве модели, корректируя на основе дополнительных свойств их расположение.

### 4.3. Анализ применения мультизадачности для задачи детекции фрагментов

В данной главе предлагается архитектура MTL с жестким разделением параметров, которая изображена на Рисунке 4.1. В HPS общий трансформерный кодировщик используется для нескольких задач. После нескольких вариаций набора параллельных линейных слоев выбор остановился на трех пользовательских классификационных головах (ССН) для одновременного обучения:

- Бинарная кастомная голова классификации (ССН) для решения начальной бинарной подзадачи [2 класса];
- Мультиклассовая ССН для определения подисточника в рамках источника НСЗ [5 классов];
- Многоклассовый ССН для определения суб-источника в источнике M4GT [6 классов].

Обучение модели проводилось в два этапа: тонкая настройка выбранных классификаторов с замороженными весами общих кодировщиков и тонкая настройка полной модели с размороженными весами. На этапе вывода для окончательной классификации использовались только бинарные предсказания ССН.

В качестве моделей для сравнения был выбран классификатор Logistic Regression с TF-IDF признаками на n-граммах слов, а также DeBERTa-v3, настроенный с двухэтапным режимом обучения, описанном ранее, но в однозадачной постановке. В подходе MTL сравнивались контрольные точки на разных этапах, а также исследовали влияние добавления пороговых значений на выход итогового классификатора. Была выбрана DeBERTa-v3-base в качестве базовой и основной в предлагаемой системе, так как в настоящее время модель показывает высокое качество на задачах классификации [57].

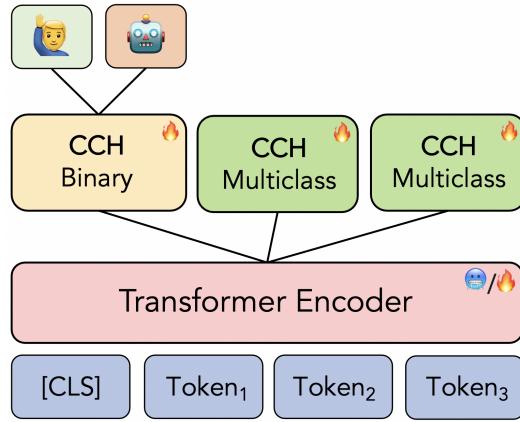


Рис. 4.1: Обзор предложенной многозадачной архитектуры. Модули, отмеченные только символом огня, поддаются обучению на всех этапах. Веса трансформерного энкодера замораживаются на первом этапе обучения и обучаются на втором. Для предсказаний используется кастомная голова классификации (CCH).

Модель	Development	Test
TF-IDF с LogReg	63.53	60.93
DeBERTaV3 base	82.56	78.52
MTL: 1 стадия	80.51	78.67
MTL: 2 стадия	87.33	81.55
MTL: 2 стадия + порог	<b>87.96</b>	<b>83.07</b>

Таблица 4.1: Результаты сравнения показателей дообученной модели на тестовом и валидационном наборе данных.

Результаты представлены в таблице 4.1. Видно, что существует слабая корреляция между разрывом в предсказаниях на подвыборках dev и test. Например, наличие порога после финального слоя незначительно повлияло на результат dev, но в то же время позволило добиться выигрышного результата на тестовом множестве.

**Кластерная структура.** В работе проведено сравнение текстовых эмбеддингов после этапов дообучения. Выборки из валидационной части данных были переданы через трансформерный энкодер, а векторы [CLS] извлечены в качестве выходных представлений. Для визуализации этих векторов использован метод главных компонент (PCA), представленный на рисунке 4.2. Анализ показывает, что выравнивание представлений, описанное ранее, приводит к формированию кластерной структуры, зависящей от домена. Хотя кластеры не являются идеально разделимыми, можно наблюдать значимое отличие между стандартной моделью типа BERT и подходами с многозадачным обучением (MTL).

**Эксперименты с исключением.** Для углублённого анализа многозадачной настройки проведены эксперименты с изменением числа многоклассовых ССН-модулей (Multiclass CCH). Базовая система, представленная в таблице, использует 2 многоклассовых ССН. В рамках экспериментов с исключением были протестированы конфигурации с 1 и 3 ССН. Результаты представлены в таблице 4.2. Конфигурации с 1 и 3 ССН показали лучшие показатели на валидационной выборке, однако их эффективность на тестовой выборке ухудшалась по сравнению с системой, использующей 2 ССН. Интересно, что результаты, полученные на ССН, обученных на данных НСЗ, оказались сопоставимыми с результатами ССН, обученных на M4GT, несмотря на то, что объём обучающей выборки M4GT в 10 раз превышает размер НСЗ. Дополнительно исследованы пороговые значения для всех конфигураций. Графики представлены на рисунке 4.3, которые подтверждают выбор финальной системы и её параметров.

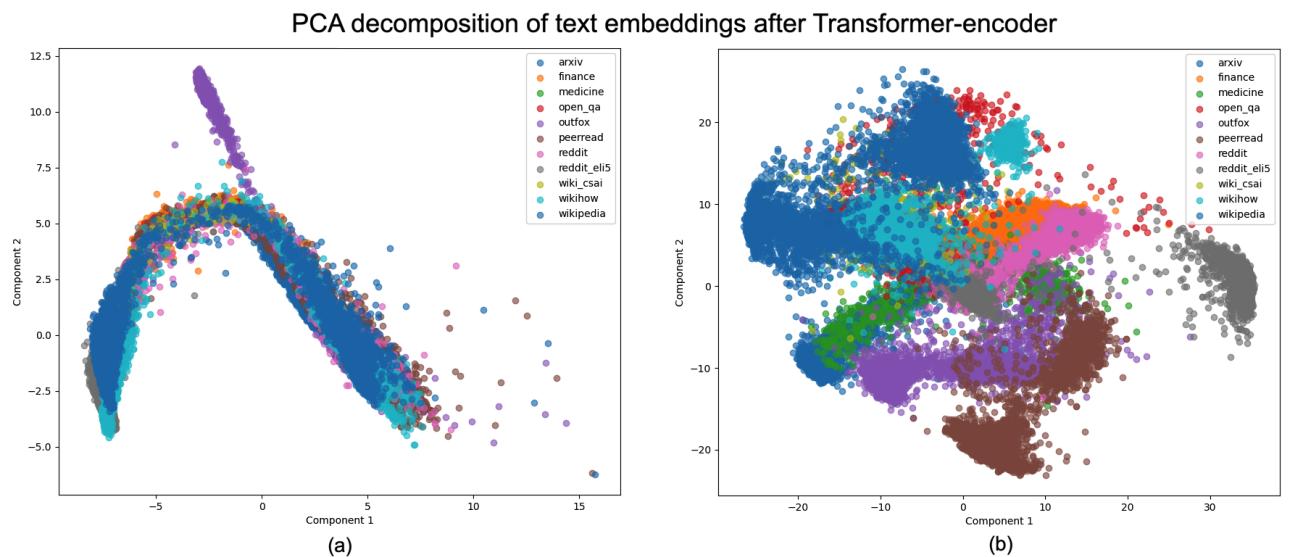


Рис. 4.2: Два главных компонента разложения PCA для текстов из валидационной выборки. На рисунке (а) показана структура векторного пространства для модели `deberta-v3-base`, настроенной в однозадачном режиме, а на рисунке (б) - та же модель, но настроенная в режиме MTL с двумя дополнительными пользовательскими классификационными головами.

В данной работе описана система детекции машинно-сгенерированных текстов, которая основана на применении подхода мультизадачного обучения. Используется общий трансформерный кодировщик и композиция кастомных голов классификации (1 бинарная и 2 многоклассовых). Предложенная система показала лучшие результаты в официальном рейтинге оценок для данного набора, обойдя базовую систему на 10%. Добавление задач для параллельного обучения показало формирование кластерной структуры в векторном пространстве, что помогло добиться высоких результатов, несмотря на наличие большого количества зашумленных данных. Также было показано, что обучение аналогичной модели, но в однозадачном режиме, проигрывает предложенному подходу,

Голова задачи	Development	Test
HC3	92.27	82.70
M4GT	91.70	81.07
MTL (HC3 + M4GT)	87.96	83.07
HC3 + M4GT + MAGE	91.43	79.23

Таблица 4.2: Сравнение различных конфигураций голов и задач, обучаемых одновременно в архитектуре MTL. Выделенная метрика - макро  $F_1$ -score (%).

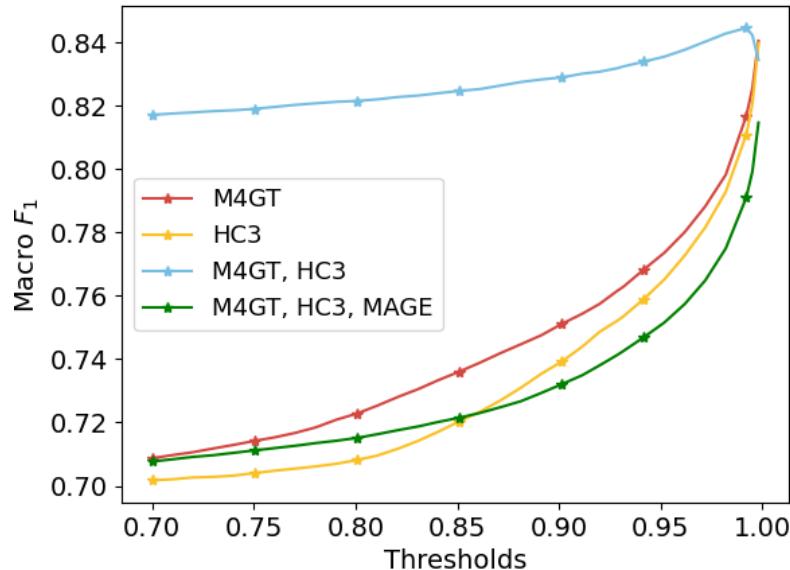


Рис. 4.3: Анализ пороговых значений.

а конфигурации с одной или тремя мультиклассовыми головами также показывают худшие результаты, чем полученная итоговая система. Помимо этого, приведено теоретическое обоснование использования данного подхода. Доказана теорема о снижении сложности модели по Радемахеру при выборе мультизадачного подхода решения.

## **Заключение**

Основные результаты диссертационной работы заключаются в следующем.

В главе 1 введены основные понятия, поставлена задача детекции машинно-сгенерированных текстовых последовательностей в терминах классификации. Исследованы методы формирования признакового пространства для текстовых фрагментов, а также предложены подходы к решению задачи. Один из них основан на дообучении предобученных моделей, использующих механизм внимания, с акцентом на параметры, влияющие на точность отражения паттернов исходного текста в векторных представлениях. Дополнительно разработан метод, опирающийся на статистические языковые модели для аппроксимации значения перплексии текста. Проведены вычислительные эксперименты, демонстрирующие зависимость качества классификации от длины входной последовательности при использовании трансформерной архитектуры. Также выполнено сравнение производительности, подтверждающее преимущество скорости работы статистических моделей по сравнению с аналогами, использующими архитектуру трансформер.

В главе 2 были предложены методы детекции машинно-сгенерированных текстовых фрагментов в составе документов. Исследованы подходы к распознаванию фрагментов с фиксированной и переменной длиной, включая алгоритмы разбиения текстового документа на непересекающиеся сегменты заданного размера. Для учета статистической зависимости, возникающей при множественном тестировании из-за высокой размерности множества сегментов, предложено применение корректировок уровня значимости. Исследована постановка задачи детекции машинно-сгенерированных фрагментов в виде задачи классификации токенов с использованием мультиголовой модели, основанной на механизме внимания. В рамках экспериментального анализа сформированы наборы данных, позволяющие повысить обобщающую способность моделей за счет искусственных искажений текста при генерации. Полученные результаты демонстрируют высокую полноту обнаружения сгенерированных символов.

В главе 3 предложены подходы агрегации методов детекции машинно-сгенерированных текстов. Исследованы подходы ансамблирования и агрегации дообученных моделей, основанных на архитектуре трансформер. Рассмотрены методы вычислительно эффективного обучения больших языковых моделей с использованием легковесных адаптеров, позволяющих минимизировать ресурсные затраты. Отражена стратегия дообучения адаптеров для покрытия распределений различных семейств больших языковых моделей. Предложен подход, основанный на интеграции выходов нескольких дообученных моделей, что обеспечивает охват языкового разнообразия в данных и повышает точность отображения текстов в векторном пространстве. Экспериментально подтверждена эффективность предложенных решений в условиях мультиязычности и вариативности генеративных моделей.

В главе 4 представлен анализ и разработан метод детекции машинно-

сгенерированных текстовых фрагментов, основанный на мультизадачном обучении, направленный на повышение обобщающей способности модели. Рассмотрены различные постановки задачи детекции в рамках MTL, включая сравнение эффективности одно- и мультизадачных подходов. Теоретический анализ сформулирован в виде теоремы, сравнивающей эмпирическую сложность модели по Радемахеру для одно- и многозадачных сценариев. Теорема демонстрирует снижение сложности модели при обучении в многозадачном режиме, что теоретически обосновывает преимущества данного подхода для целевой задачи. Проведены вычислительные эксперименты, направленные на исследование влияния отдельных компонентов системы путём их последовательного исключения. Показано, что обучение в многозадачной постановке способствует формированию кластерной структуры текстовых представлений в векторном пространстве, что выступает в роли неявной регуляризации и улучшает устойчивость модели к вариациям доменов и генеративных моделей.

## Список иллюстраций

1.1	Цикл работы предлагаемого метода, основанного на использовании статистических языковых моделей. На первом этапе выполняется предварительная настройка, для которой требуются тексты из разных LLM. На их основе строятся Агра-словари. Агра-словари используются для инициализации статистических языковых моделей KenLM. На втором этапе представлен алгоритм обнаружения. Подозрительный текст представляется в числовом виде на основе признаков от предварительно собранных KenLM. После этого векторы подаются на вход предварительно обученного LightGBM-классификатора, который и производит окончательную классификацию. . . . .	12
1.2	Точность детекции для английского языка растет по мере увеличения длины последовательностей, используемых для обучения дискриминатора. После $n=512$ метрика достигает плато. Красная линия показывает границу предыдущих экспериментов с увеличением длины. . . . .	14
1.3	Большинство образцов данных RuATD имеют длину до 200 токенов, при таком количестве токенов модели будет сложно понять длинный контекст. . . . .	15
1.4	Точность детекции на русском языке возрастает по мере увеличения длины последовательностей, используемых для обучения дискриминатора, для всех представленных методов выборки. После $n=256$ метрика достигает плато. Коричневая линия показывает среднюю длину лексем в наборе данных RuATD. . . . .	16
1.5	Сравнение статистики N-грамм Агра-словарей, построенных на словах и токенах. Можно заметить, что использование токенов при построении позволяет уменьшить количество уникальных N-грамм, но порядок значений остается прежним. . . . .	16
1.6	Сравнение метрик качества классификации в зависимости от количества статистических языковых моделей, используемых для построения числового представления подозрительного текста. Можно заметить, что качество становится выше при использовании большего числа моделей KenLM. . . . .	20
1.7	Графики изменения порогового значения классификации. Левый график показывает значение полноты обнаружения при изменении порога, а правый — уровень ложных срабатываний на документах, написанных человеком, при изменении порога. . . . .	21
2.1	Полный цикл работы алгоритма детекции сгенерированных фрагментов в научных документах. . . . .	26

2.2	На рисунке (а) представлено количественное разбиение текстов русской части данных по частям речи, а на рисунке (б) для английского. Здесь NOUN — существительное, ADJ(F) — прилагательное (полное), PREP — предлог, VERB — глагол, CONJ — союз, NPRO — местоимение, ADV(B) — наречие, PRCL — частица, INFN — инфинитив, ADP — дополнения (предлоги), DET — артикли, числительные, PRON — местоимения, PRT — частицы. . . . .	27
2.3	Многозадачная архитектура для решения задачи классификации токенов. Каждый токен, после получения векторного представления текста с помощью кодировщика, классифицируется с помощью двух голов: линейная А — бинарная классификация, линейная В — мультиклассовая классификация. . . . .	30
4.1	Обзор предложенной многозадачной архитектуры. Модули, отмеченные только символом огня, поддаются обучению на всех этапах. Веса трансформерного энкодера замораживаются на первом этапе обучения и обучаются на втором. Для предсказаний используется кастомная голова классификации (ССН). . . . .	49
4.2	Два главных компонента разложения РСА для текстов из валидационной выборки. На рисунке (а) показана структура векторного пространства для модели <code>deberta-v3-base</code> , настроенной в однозадачном режиме, а на рисунке (б) - та же модель, но настроенная в режиме MTL с двумя дополнительными пользовательскими классификационными головами. . . . .	50
4.3	Анализ пороговых значений. . . . .	51

## Список таблиц

1.1	Результаты экспериментов со статистическими языковыми моделями KenLM. Рассматриваются различные варианты создания Агра-словаря, а также их объединение в ансамбль. Значение в каждой ячейке представляет собой метрику полноты обнаружения искусственно сгенерированного фрагмента. . . . .	17
1.2	Результаты сравнения предложенного метода с уже известными подходами на трех разных наборах данных. Собранный вручную набор данных с публично доступными выходами различных моделей содержит 2 тыс. сбалансированных примеров, а НС3 и Binoculars - по 1 тыс. примеров. Можно заметить, что производительность предложенного метода значительно выше, чем у аналогов. . . . .	18
2.1	Пара примеров представления строк в предоставленном наборе данных. Каждый пример представлен текстовой строкой, ее разметкой, разбиением на токены и метками для каждого токена. . .	29
2.2	Сводная таблица результатов вычислительного эксперимента. . .	33
2.3	Результаты на валидационном наборе данных с различными настройками для предложенной многозадачной архитектуры. . . .	34
3.1	Структура обновления весов при использовании адаптера LoRA .	38
3.2	Результаты предложенного метода с сравнении с иными методами детекции. Отражены метрики качества ROC-AUC, Brier, C@1, F <sub>1</sub> , F <sub>0.5u</sub> , а также их среднее значение. . . . .	41
3.3	Сравнение метрик моделей на отложенной выборке данных. Столбец «без» означает, что для языков, на которых обучалась модель, используется только указанный в строке ответ модели, а для остальных примеров используется ответ BLOOM. В столбце «с» указаны метрики агрегирования, когда ответ модели сочетается с выходом BLOOM. . . . .	42
3.4	Сравнение показателей качества детекции для двух отдельных кодировщиков. Можно заметить, что качество на низкоресурсных и высокоресурсных языках сильно различается. . . . .	42
4.1	Результаты сравнения показателей дообученной модели на тестовом и валидационном наборе данных. . . . .	49
4.2	Сравнение различных конфигураций голов и задач, обучаемых одновременно в архитектуре MTL. Выделенная метрика - макро F <sub>1</sub> -score (%). . . . .	51

## Список литературы

1. *Gritsai G., Khabutdinov I., Grabovoy A.* Multi-head Span-based Detector for AI-generated Fragments in Scientific Papers // *Proceedings of the Fourth Workshop on Scholarly Document Processing (SDP 2024)*. — 2024.
2. Real or Fake Text?: Investigating Human Ability to Detect Boundaries between Human-Written and Machine-Generated Text / L. Dugan, D. Ippolito, A. Kirubarajan et al. // *Proceedings of the AAAI Conference on Artificial Intelligence*. — 2023.
3. *Ray Partha Pratim.* ChatGPT: A Comprehensive Review on Background, Applications, Key Challenges, Bias, Ethics, Limitations and Future Scope // *Internet of Things and Cyber-Physical Systems*. — 2023.
4. *Touvron H., Lavril T., Izacard G. et al.* LLaMA: Open and Efficient Foundation Language Models. — 2023.
5. *Jiang A., Sablayrolles A., Mensch A. et al.* Mistral 7B. — 2023.
6. *GigaChat by SberDevices*. — Accessed: 2025-04-05. <https://giga.chat/>.
7. *YaGPT by Yandex*. — Accessed: 2025-04-05. <https://yandex.ru/project/alice/yagpt>.
8. *Chen C., Shu K.* Can LLM-Generated Misinformation Be Detected? — 2024.
9. *Chekhovich Yu., Grabovoy A., Gritsai G.* Generative AI Models with Their Full Reveal // *2024 4th International Conference on Technology Enhanced Learning in Higher Education (TELE)*. — 2024.
10. *Gritsay G.M., Grabovoy A.V., Kildyakov A.S. et al.* Artificially Generated Text Fragments Search in Academic Documents // *Doklady Mathematics*. — 2024.
11. *Grashchenkov K., Grabovoy A., Khabutdinov I.* A Method of Multilingual Summarization For Scientific Documents // *2022 Ivannikov Ispras Open Conference (ISPRAS)*. — 2022.
12. *Boeva G., Gritsai G., Grabovoy A.* Team ap-team at PAN: LLM Adapters for Various Datasets // *CLEF 2024: Conference and Labs of the Evaluation Forum*. — 2024.
13. *Bakhteev O., Ogaltsov A., Ostroukhov P.* Fake News Spreader Detection Using Neural Tweet Aggregation—Notebook for PAN at CLEF 2020 // *CLEF 2020 Labs and Workshops, Notebook Papers*. — 2020.
14. *Avetisyan K., Gritsay G., Grabovoy A.* Cross-Lingual Plagiarism Detection: Two Are Better Than One // *Programming and Computer Software*. — 2023.
15. *Bakhteev O. et al.* Cross-language plagiarism detection: A Case Study of European Universities Academic Works // *Academic Integrity: Broadening Practices, Technologies, and the Role of Students*. — 2022.
16. *Casal J., Kessler M.* Can Linguists Distinguish Between ChatGPT/AI and Human Writing? A Study of Research Ethics and Academic Publishing // *Research Methods in Applied Linguistics*. — 2023.

17. *Liang W., Zhang Ya., Wu Z. et al.* Mapping the Increasing Use of LLMs in Scientific Papers. — 2024.
18. *Gray A.* ChatGPT "contamination": Estimating the Prevalence of LLMs in the Scholarly Literature. — 2024.
19. Authorship Attribution for Neural Text Generation / A. Uchendu, T. Le, K. Shu, D. Lee // *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. — 2020.
20. *Zellers R., Holtzman A., Rashkin H. et al.* Defending Against Neural Fake News. — 2019.
21. *Zhu E., Zhang J., Yan J. et al.* N-gram MalGAN: Evading Machine Learning Detection via Feature n-gram // *Digital Communications and Networks*. — 2021.
22. *Nguyen T., Hatua A., Sung A.* How to Detect AI-Generated Texts? // *2023 IEEE 14th Annual Ubiquitous Computing, Electronics and Mobile Communication Conference (UEMCON)*. — 2023.
23. *Jawahar G., Abdul-Mageed M., Lakshmanan L.* Automatic Detection of Machine Generated Text: A Critical Survey. — 2020.
24. *Wang H., Li J., Li Zh.* AI-Generated Text Detection and Classification Based on BERT Deep Learning Algorithm. — 2024.
25. Automatic Detection of Generated Text is Easiest when Humans are Fooled / D. Ippolito, D. Duckworth, C. Callison-Burch, D. Eck. — 2019.
26. Automated Text Identification: Multilingual Transformer-based Models Approach / German Gritsay, Andrey Grabovoy, Aleksandr Kildyakov, Yury Chekhovich // *IberLEF@ SEPLN*. — 2023.
27. DetectGPT: Zero-Shot Machine-Generated Text Detection using Probability Curvature / E. Mitchell, Yo. Lee, A. Khazatsky et al. — 2023.
28. Spotting LLMs With Binoculars: Zero-Shot Detection of Machine-Generated Text / A. Hans, A. Schwarzschild, V. Cherepanova, H. Kazem. — 2024.
29. *Li Y., Li Q., Cui L. et al.* MAGE: Machine-generated Text Detection in the Wild. — 2024.
30. Fast-DetectGPT: Efficient Zero-Shot Detection of Machine-Generated Text via Conditional Probability Curvature / G. Bao, Ya. Zhao, Z. Teng et al. — 2024.
31. Ghostbuster: Detecting Text Ghostwritten by Large Language Models / V. Verma, E. Fleisig, N. Tomlin, D. Klein. — 2024.
32. *Gritsai German M., Khabutdinov Ildar Ayratovich, Grabovoy Andrey Valerievich.* Stack More LLM's: Efficient Detection of Machine-Generated Texts via Perplexity Approximation // *Doklady Mathematics*. — 2024. — Vol. 110, no. Suppl 1. — Pp. S203–S211.
33. *Heafield K.* KenLM: Faster and Smaller Language Model Queries // *Proceedings of the Sixth Workshop on Statistical Machine Translation*. — 2011.

34. *Gritsay G., Grabovoy A., Chekhovich Yu.* Automatic Detection of Machine Generated Texts: Need More Tokens // *2022 Ivannikov Memorial Workshop (IVMEM)*. — 2022.
35. TURINGBENCH: A Benchmark Environment for Turing Test in the Age of Neural Text Generation / Adaku Uchendu, Zeyu Ma, Thai Le et al. // Findings of the Association for Computational Linguistics: EMNLP 2021. — Punta Cana, Dominican Republic: Association for Computational Linguistics, 2021. — . — Pp. 2001–2016. <https://aclanthology.org/2021.findings-emnlp.172>.
36. *Maloyan Narek, Nutfullin Bulat, Ilyushin Eugene.* DIALOG-22 RuATD Generated Text Detection. — 2022. — 06. <https://doi.org/10.48550/arXiv.2206.08029>.
37. *Dataset with GPT-2 Output.* — Accessed: 2025-04-05. <https://github.com/openai/gpt-2-output-dataset>.
38. *Guo B., Zhang X., Wang Z. et al.* How Close is ChatGPT to Human Experts? Comparison Corpus, Evaluation, and Detection. — 2023.
39. *Wang Yu., Mansurov J., Ivanov P. et al.* SemEval-2024 Task 8: Multidomain, Multimodel and Multilingual Machine-Generated Text Detection. — 2024.
40. *Ke G., Meng Q., Finley Th. et al.* LightGBM: A Highly Efficient Gradient Boosting Decision Tree // *Neural Information Processing Systems*. — 2017.
41. RoFormer: Enhanced Transformer with Rotary Position Embedding / Jianlin Su, Yu Lu, Shengfeng Pan et al. // *CoRR*. — 2021. — Vol. abs/2104.09864. <https://arxiv.org/abs/2104.09864>.
42. *Vasilatos Ch., Alam M., Rahwan T. et al.* HowkGPT: Investigating the Detection of ChatGPT-generated University Student Homework through Context-Aware Perplexity Analysis. — 2023.
43. *Keskar N., Muszkiewicz A., Ribeiro A.* On the Role of Text Generation in Language Model Evaluation. — 2022.
44. *Zhang H., Chiang D.* Kneser-Ney Smoothing on Expected Counts // *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. — 2014.
45. Overview of the DAGPap22 Shared Task on Detecting Automatically Generated Scientific Papers / Yury Kashnitsky, Drahomira Herrmannova, Anita de Waard et al. // Proceedings of the Third Workshop on Scholarly Document Processing / Ed. by Arman Cohan, Guy Feigenblat, Dayne Freitag et al. — Gyeongju, Republic of Korea: Association for Computational Linguistics, 2022. — . — Pp. 210–213. <https://aclanthology.org/2022.sdp-1.26>.
46. Deepfake Text Detection in the Wild / Yafu Li, Qintong Li, Leyang Cui et al. // *arXiv preprint arXiv:2305.18653*. — 2023.
47. *Holm Sture.* A Simple Sequentially Rejective Multiple Test Procedure // *Scandinavian Journal of Statistics*. — 1979. — Vol. 6.

48. *Benjamini Yoav, Hochberg Yosef.* Controlling the False Discovery Rate: A Practical and Powerful Approach to Multiple Testing // *Journal of the Royal Statistical Society. Series B (Methodological)*. — 1995. — Vol. 57.
49. *Chen Yu., Kang H., Zhai V. et al.* Token Prediction as Implicit Classification to Identify LLM-Generated Text // *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*. — 2023.
50. Grammatical Error Correction: A Survey of the State of the Art / Christopher Bryant, Zheng Yuan, Muhammad Reza Qorib et al. // *Computational Linguistics*. — 2023. — 09. — Vol. 49, no. 3. — Pp. 643–701. [https://doi.org/10.1162/coli\\_a\\_00478](https://doi.org/10.1162/coli_a_00478).
51. GECToR – Grammatical Error Correction: Tag, Not Rewrite / Kostiantyn Omelianchuk, Vitaliy Atrasevych, Artem Chernodub, Oleksandr Skurzhanskyi // Proceedings of the Fifteenth Workshop on Innovative Use of NLP for Building Educational Applications / Ed. by Jill Burstein, Ekaterina Kochmar, Claudia Leacock et al. — Seattle, WA, USA → Online: Association for Computational Linguistics, 2020. — . — Pp. 163–170. <https://aclanthology.org/2020.bea-1.16>.
52. *Ando Rie Kubota, Zhang Tong.* A Framework for Learning Predictive Structures from Multiple Tasks and Unlabeled Data // *Journal of Machine Learning Research*. — 2005. — . — Vol. 6. — Pp. 1817–1853.
53. *Beltagy Iz, Lo Kyle, Cohan Arman.* SciBERT: A Pretrained Language Model for Scientific Text // *arXiv preprint arXiv:1903.10676*. — 2019.
54. XLNet: Generalized Autoregressive Pretraining for Language Understanding / Zhilin Yang, Zihang Dai, Yiming Yang et al. // *arXiv preprint arXiv:1906.08237*. — 2019.
55. QLoRA: Efficient Finetuning of Quantized Language Models / Tim Dettmers, Mike Lewis, Samuel Lenc et al. // *arXiv preprint arXiv:2305.14314*. — 2023.
56. RoBERTa: A Robustly Optimized BERT Pretraining Approach / Yinhan Liu, Myle Ott, Naman Goyal et al. // *arXiv preprint arXiv:1907.11692*. — 2019.
57. DeBERTa: Decoding-enhanced BERT with Disentangled Attention / Pengcheng He, Xiaodong Liu, Jianfeng Gao, Weizhu Chen // *arXiv preprint arXiv:2006.03654*. — 2020.
58. *Gritsay G., Grabovoy A.* Automated Text Identification on Languages of the Iberian Peninsula: LLM and BERT-based Models Aggregation // *IberLEF@SEPLN*. — 2024.
59. LoRA: Low-Rank Adaptation of Large Language Models / Edward J. Hu, Yelong Shen, Phillip Wallis et al. // *arXiv preprint arXiv:2106.09685*. — 2021.
60. BLOOM: A 176B-Parameter Open-Access Multilingual Language Model / Teven Le Scao, Amanpreet Fan, Christopher Akiki et al. // *Proceedings of the Joint Conference of the 60th Annual Meeting of the ACL and the 10th International Joint Conference on Natural Language Processing (ACL-IJCNLP 2022)*. — 2022. — Pp. 8547–8567.

61. *Crawshaw Michael*. Multi-Task Learning with Deep Neural Networks: A Survey. — 2020. <https://arxiv.org/abs/2009.09796>.
62. *Gritsai German, Voznyuk Anastasia, Khabutdinov Ildar, Grabovoy Andrey*. Advacheck at GenAI Detection Task 1: AI Detection Powered by Domain-Aware Multi-Tasking. — 2024. <https://arxiv.org/abs/2411.11736>.
63. GenAI Content Detection Task 1: English and Multilingual Machine-generated Text Detection: AI vs. Human / Yuxia Wang, Artem Shelmanov, Jonibek Mansurov et al. // Proceedings of the 1st Workshop on GenAI Content Detection (GenAIDetect). — Abu Dhabi, UAE: International Conference on Computational Linguistics, 2025. — January.
64. Are AI Detectors Good Enough? A Survey on Quality of Datasets With Machine-Generated Texts / German Gritsai, Anastasia Voznyuk, Andrey Grabovoy, Yury Chekhovich. — 2024.