Пшеничников Глеб Викторович, ВМК МГУ, 212 группа

Задача бинарной классификации (machine vs human).

Современные текстовые генеративные модели способны писать тексты, почти не отличимые от человеческих. Зачастую это может привести к плагиату, спаму, появлению ложной информации и неэтичным практикам. С развитием генеративных текстовых моделей эти проблемы только растут. Поэтому задача выявления текстов, сгенерированных искусственным интеллектом, стоит остро и требует качественного решения. В этой работе я представлю решение задачи бинарной классификации, построю модель, которая будет отделять тексты, написанные человеком, от сгенерированных текстов.

Основные идеи, методы, выкладки взяты из статей [1] и [2].

ВВЕДЕНИЕ

Текстовые генеративные модели непрерывно развиваются. Механизм само-внимания (позволяет работать модели внутри контекста, указывая, на какие фрагменты следует обратить внимание), большие объемы данных, методы дообучения моделей на основе трансформера (используются при генерации текста) позволяют добиться «осмысленности» текста, позволяют перефразировать, переводить документы. Однако, это не гарантирует правдивость полученной информации (так как основной метод при генерации текста основан на минимизации уровня перплексии).

Генеративные модели часто используются при написании научных работ. Чаще всего искусственно созданные текста используются во введении, основной части и заключении. При попытке детекции сгенерированных частей возникает проблема – анализируемый документ имеет слишком большой размер. При разбиение таких текстов на сегменты, которые в последующем будут анализироваться независимо, получается большое количество ложно-положительных срабатываний. В данной работе реализован метод, решающий эту проблему.

Цель работы состоит в построении бинарного классификатора, определяющего написан ли поданный на вход текст человеком или же сгенерирован искусственным интеллектом. Для реализации будет использован кодировщик трансформер, который будет дообучен на тренировочных данных. Также будет реализован метод классификации текстов обученной моделью с учетом проблемы больших размером документов, описанной выше.

ПОСТАНОВКА ЗАДАЧИ

Формально задачу можно описать следующим образом: необходимо разделить текст на сегменты меньшей длины, проанализировать каждый из них и вынести вердикт о сгенерированности/не сгенерированности всего текста. Математически эту задачу можно описать следующим образов:

W – алфавит, содержащий символы представленных текстов.

$\mathbb{D}$ – множество текстов; $\mathbb{D} = \left\{ \left[t_j \right]_{j=1}^{n} \;\middle|\; t_j \in W, n \in \mathbb{N} \right\}$

Для каждого текста необходимо задать модель ϕ, которая будет определять сгенерирован ли текст. Зададим ее в виде суперпозиции двух преобразований f и g. Преобразование f –

разделение текста на фрагменты. Преобразование g – классификатор текстовых последовательностей:

$$\phi = f \circ g$$

$$\phi: \mathbb{D} \to \mathbb{T}, \qquad \mathbb{T} = \left\{ \left[t_{s_j}, t_{f_j}, c_j \right]_{j=1}^{J} \;\middle|\; t_{s_j} = t_{f_{j-1}}, \qquad s_j \in \mathbb{N}_0, \qquad f_j \in \mathbb{N}, \qquad c_j \in \mathbb{C} \right\}$$

Где J – количество фрагментов после этапа фрагментации документа, t_{s_j}- стартовый индекс j-го фрагмента, t_{f_j}- завершающий индекс j-го фрагмента, c_j – класс (метка) j-го фрагмента. $\mathbb{C} = \{0,1\}$, где метка $c_j = 1$ означает, что текст машинно-сгенерирован, метка $c_j = 0$ означает, что текст написан человеком.

Первый элемент суперпозиции – преобразование, разделяющее текст на непересекающиеся фрагменты:

$$f: \mathbb{D} \to T^*, \;\; T^* = \left\{ \left[t_{s_j}, t_{f_j} \right]_{j=1}^{J} \;\middle|\; t_{s_j} = t_{f_{j-1}}, s_j \in \mathbb{N}_0, f_j \in \mathbb{N} \right\}$$

Где t_{s_j}- стартовый индекс j-го фрагмента, t_{f_j}- завершающий индекс j-го фрагмента. Следовательно, T^*- множество всех возможных непересекающихся фрагментов, полностью покрывающих документ.

Второй элемент суперпозиции – бинарная классификация каждого текстового фрагмента:

$$g: T^* \to \mathbb{C}$$

Далее по меткам сегмента определяется метка всего текста (например, при помощи мажоритарного голосования): $F\left(C_1, C_2, \dots, C_J\right) = C_0$, где $C_1, C_2, \dots, C_J$ – метки сегментов, C_0- метка текста целиком (документа).

Решение задачи можно разбить на несколько подзадач:

1. Создание датасета (для дообучения модели).
2. Дообучение модели на основе трансфера кодировщика на тренировочных данных.
3. Предложить метод классификации длинных текстов при помощи построенной модели и проверить насколько он применим.

При реализации каждой подзадачи будет рассматриваться несколько возможных методов (для реализации пункта 1 и 3) или моделей (для реализации пункта 2), будет проведено их сравнение и выбраны оптимальные из них.

СОЗДАНИЕ ДАТАСЕТА

В качестве данных для обучения и тестирования модели была взята выборка с соревнования SemEval 2024 [3], которая содержит два набора данных: на английском языке и на «разных» языках (арабский, русский, китайский, индонезийский, урду, болгарский, немецкий). Каждый набор включает в себя выборку для обучения и для тестирования.

Данные являются JSONL-файлами. Они имеют следующий формат

{

id -> identifier of the example,

label -> label (human text: 0, machine text: 1,),

text -> text generated by a machine or written by a human,

model -> model that generated the data,

source -> source (Wikipedia, Wikihow, Peerread, Reddit, Arxiv) on English or language (Arabic, Russian, Chinese, Indonesian, Urdu, Bulgarian, German)

}

Основными полями для обучения и тестирования бинарного классификатора являются «text» (содержит сам текст) и «label» (имеет значение 0, если текст написан человеком, и значение 1 – если сгенерирован).

Набор данных на английском языке содержит в тренировочной выборке 119.757 текстов, в тестовой выборке - 5.000. Набор данных на «разных» языках содержит в тренировочной выборке 172.417 текстов, в тестовой выборке - 4.000. Обе тренировочные выборки приблизительно сбалансированы по количеству текстов с «label» 0 и с «label» 1, а тестовая выборка точно сбалансирован по этому критерию.

Однако в силу того, что кодировщик трансформер может принимать на вход ограниченное количество термов, при обучении необходимо корректно разбить тренировочные текста на сегменты меньшей длины. При этом необходимо сохранить контекст. Для этого будем разбивать текста на сегменты длиной 256 токенов, при этом в каждом следующем сегменте первые 50 токенов будут из предыдущего сегмента (то есть получается накладывание следующего сегмента на предыдущий). Это обеспечит сохранение контекста в сегментах. Сопоставим полученным сегментам label такой же, как у текста, из которого они были созданы – в итоге будем иметь несколько текстов с приемлемой длиной. Применяя этот прием ко всему тренировочному датасету, получим набор текстов, каждый из которых может быть полностью воспринят трансформером кодировщиком. То есть, описывая этот метод математически, необходимо провести следующее преобразование r:

W – алфавит, содержащий символы представленных текстов.

$\mathbb{D}$ – множество текстов; $\mathbb{D} = \left\{ \left[t_j \right]_{j=1}^{n} \;\middle|\; t_j \in W, n \in \mathbb{N} \right\}$

N – количество текстов

Для каждого $D_i \in \mathbb{D}$ $(i = 1, 2, \dots, N)$ зададим преобразование r:

r: $N_i \to \{N_{ij}\}$ $(j = 1,2, \dots ,\ (len(N_i) - 1)\ //\ 256\ + 1\}$, где N_{i1} = {[0;256] токенов}, $N_{ij} = \{[(j-1) * 256 - 50; j * 256]$ токенов$\}$ для $j\ \neq 1$

ДООБУЧЕНИЕ МОДЕЛИ

Существуют несколько способов дообучить модель, основанную на архитектуре трансформера. В работе будут рассмотрены некоторые из них (взяты из [1]). Представленные методы относятся к датасету на английском языке.

- **Ручные признаки:** предлагается вручную выявить некоторое количество признаков (числовых статистик) у текстов и представить их в виде вектора (авторами статьи [1] было предложено 26 числовых статистик), тем самым создав признаковое пространство. Подход заключается в конкатенации признаков, собранных вручную, с признаками, полученными от кодировщика (BERT-подобные модели описывают поданный на вход текст при помощи 768 признаков на базе [CLS]).
- **Мультиязычное дообучение:** использование для обучения смешанных данные из двух разных языков.
- **Перевод текстов:** замена 10%, 20% и 50% сгенерированных текстов переводом на русский язык при помощм Google Translate [4]
- **Перефразирование текстов:** подмешивание в сгенерированные текста выборки перефразированных фрагментов (при помощи дообученной модели t5 sentence paraphaser [5]). Рассматриваются два способа перефразирования: в каждом сгенерированном тексте перефразируется от 0% до 100% предложений; в половине сгенерированных текстов перефразируется от 50% до 100% предложений.
- **T5:** использование модели T5. Рассматриваются два подхода: использование полной энкодер-декодер модели; от полной модели T5 взят энкодер и поверх него добавлен нейросетей классификатор.

Данные методы были протестированы в статье [1]:

Язык	Эксперимент	F1-score	Precision	Recall
en	базовое решение	0,796	0,855	0,802
	ручные признаки	0.801	0.856	0.807
	мультиязычное 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%	**0,825**	**0,868**	**0,830**
	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

Из таблицы видно, что лучшие результаты показал метод, основанный на переводе 50% сгенерированных текстов на русский язык. Применим к датасету на английском, на котором будет обучаться модель, реализуемая в данной работе, этот метод (сначала произведем перевод 50% сгенерированных текстов, а потом разделим текст на длины приемлемой длины, о чем говорилось выше).

Необходимо выбрать модель трансформера кодировщика, при помощи которой будет решаться задача бинарной классификации (эксперименты и выводы взяты из статьи [2], в которой сравнивались модели для выявление лучшей в задаче бинарной классификации английских текстов). Рассмотрим следующие три трансформера кодировщика:

1. **RoBERTa (Robustly Optimized BERT Pretraining Approach)** - имеет ту же архитектуру, что и BERT [5], но использует побайтовый BPE в качестве токенизатора. Для этой задачи

мы использовали XLM-RoBERTa (предварительно обучена на 2,5 ТБ отфильтрованных данных Common Crawl, содержащих 100 языков).

2. **DeBERTa (Decoding-enhanced BERT with Disentangled Attention)** - улучшает модели BERT и RoBERTa при помощи механизма разделенного внимания (содержание и позиция каждого слова кодируется двумя векторами соответственно) и улучшенного декодера масок. Будем рассматривать многоязычную версию mDeBERTa (предварительно обучена на данных 2.5T Common Crawl 100).

3. **MiniLM-L12-v2 (Multi-Head Self- Attention Relation Distillation for Compressing Pre-trained Transformers)** - обобщает дистилляцию глубокого самовнимания в MiniLM [7], использует отношения многоголового самовнимания для обучения ученика. В целом, это дистиллированная модель из учителей большого размера (BERT, RoBERTa, XLM-RoBERTa-large), которая использует реляционные знания. Рассмотрим чекпойнт этой модели из хаба Sentence Transformers - miniLM-L12-v2 (так как модель уже многоязычная).

Также в статье был сделан вывод, что производительность многоязычной версии больше, чем производительность одноязычной (разные языки позволяют улучшить качество векторных представлений).

Все модели были дообучены однообразно на одинаковом датасете. Результаты сравнения показаны в таблице. Сравнения проводились по мере F1.

Model	**Datasets**	
	Original Data	***Processed Data***
XLM-RoBERTa	86.86	88.75
mDeBERTa V3	90.42	**93.07**
MiniLM-L12-v2	89.63	90.49

Видно, что модель mDeBERTa показала лучший результат, ее будем использовать для решения задачи, поставленной в настоящей работе.

Далее необходимо определить оптимальные настройки сети для дообучения трансформера кодификатора (изначально значения взяты также из статьи [2], далее определялись более подходящие параметры; в качестве валидационных выборок использовалось 100 текстов на английском языке и на «разных» языках). Обучение производилось со следующими настройками:

- Определена головная часть для классификации (работает с эмбеддингом [CLS] на выходе кодировщика трансформера). Она состоит из трех полносвязных слоев.
- Функция активации GELU. Используется метод Dropout.
- Функция потерь: кросс-энтропия
- Использование батчей (размер батча 16)
- Оптимизатор: AdamW, learning rate = 1e-5

- Использование техники регуляризации сглаживания меток со значением 0.1 к функции потерь
- Использование планировщика MultiStepLR с milestones = [2, 4] и gamma = 0.5
- Наличие 5 эпох (для тонкой настройки): на 1 и 5 эпохах обучается только классификатор с замороженными весами, на 2–4 эпохах обучается полная модель

КЛАССИФИКАЦИЯ ТЕКСТОВ

Для бинарной классификации длинных текстов требуется сегментация, как было указано во введении. Необходимо предложить метод, который будет корректно оценивать сгенерирован ли текст или нет, анализируя сегменты (которые мы можем полностью подать на вход модели и получить вероятность «сгенерированности»). Методы, которые будут рассматриваться, взяты из статьи [1].

Введем две гипотезы. Нулевая гипотеза – текст написан человеком (классификатор выдает метку 0). Первая (альтернативная) гипотеза – текст сгенерирован (классификатор выдает метку 1).

$$H_0: \hat{g}(fragment) = 0$$

$$H_1: \hat{g}(fragment) = 1$$

При работе с фрагментами по-отдельности, при проверке большого числа гипотез, будут накапливаться ошибки первого рода (ложные отклонения гипотезы). Пусть уровень значимости $\alpha = 0.05$, m – количество фрагментов. Тогда верхняя оценка вероятности появления хотя бы одного ложноположительного результата:

$$P(false\ positive) = 1 - (1 - \alpha)^m$$

Эта величина велика даже при небольших значениях m. Поэтому требуется ввести меры контроля ошибки первого рода. Вводятся две такие меры: **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 ошибок первого рода, S – число истинно положительных результатов.

Для контроля FWER на уровне α предлагается метод Холма: сортировка p-value по возрастанию, уровень значимости при этом меняется в таком порядке:

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

Тогда если $p_1 \geq \alpha_1$, то все нулевые гипотезы не отвергаются, иначе отвергается первая и продолжаем.

Для контроля FDR вводится метод Бенджамини-Хохберга: сортировка p-value по возрастанию, уровень значимости при этом меняется по следующему закону:

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

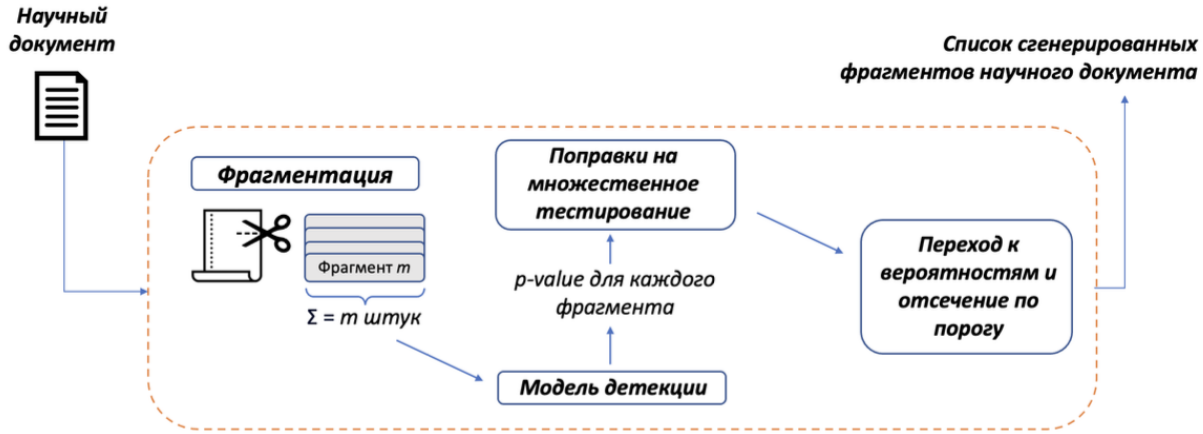
Тогда если $p\_m < \alpha_m$, то необходимо отвергнуть все гипотезы, иначе – не отвергать m-ую, и продолжить.

Замечание: для любой процедуры множественного тестирования гипотез FDR ≤ FWER.

То есть, в большинстве случаев метод Бенджамини-Хохберга оказывается мощнее метода Холма. Поэтому в данной работе будет использоваться метод Бенджамини-Хохберга для классификации текстов.

Используя метод Бенджамини-Хохберга мы сможем скорректировать метки сегментов (сгенерирован ли сегмент или нет). Далее агрегируя метки сегментов, найдем количество сгенерированных сегментов и количество не сгенерированных сегментов. Если будет больше сегментов, которых классификатор подсчитал сгенерированными, то текст будем считать сгенерированным, иначе – написанным человеком, то есть при помощи мажоритарного голосования.

Всю процедуру разделения текста, классификации сегментов можно представить в виде схемы (взято из статьи [1]).



РЕЗУЛЬТАТЫ ЭКСПЕРИМЕНТОВ

В силу отсутствия необходимой вычислительной мощности компьютера, на котором будут проводиться эксперименты, было решено укоротить тренировочную и тестовую выборки. Насколько была укорочена тренировочная выборка будет указываться в каждом эксперименте. Тестовая выборка была укорочена до 1000 тестов для всех проводимых экспериментов (в датасетах на английском и на «разных» языках). Укороченные тестовая и тренировочная выборки, как и в полной версии, сбалансированы по количеству примеров сгенерированных текстов и текстов, написанных человеком (в укороченных тренировочной и тестовой выборках на «разных» языка имеются все языки, имеющиеся в полной версии).

Сначала оценим качество модели, насколько она хорошо обучается. Будем обучать модель при помощи датасетов разных размеров (500, 1000, 2000 текстов) и сравнивать полученные меры F1, полноту и точность результатов. При этом при обучении на английском датасете тестировка будет проводиться на английском тестовом датасете, при обучении на датасете с «разными» языками – на тестовом датасете с «разными» языками. Также рассмотренный метод перевода текстов на русский язык будет применен только к английскому датасету. Результаты можно видеть в таблицах 1 и 2.

Кол-во текстов/Мера	F1	Полнота	Точность
500	0.3984	0.4008	0.3960
1000	0.5338	0.6080	0.4757
2000	0.5694	0.5856	0.5540

Таблица 1: оценка модели при обучении и тестировке на английских текстах

Кол-во текстов/Мера	F1	Полнота	Точность
500	0.5231	0.5200	0.5263
1000	0.5910	0.6140	0.5696
2000	0.7162	0.8380	0.6254

Таблица 2: оценка модели при обучении и тестировке на датасетах с «разными» языками

Определим, как влияет дообучение модели на датасете с «разными» языками на классификацию английских текстов и как влияет дообучение модели на английском датасете на классификацию текстов с «разными» языками. Для этого:

1. Обучим модель на английских текстах (датасет с 1000 текстами), а протестируем на датасете с «разными» языками.
2. Обучим модель на текстах с «разными» языками (датасет с 1000 текстами), а протестируем на датасете с английскими текстами.
3. Обучим модель на датасете, являющимся объединением двух датасетов: датасета из английских текстов (датасет с 500 текстами) и датасета с «разными» языками (датасет с 500 текстами). Протестируем для английских текстов и для текстов с «разными» языками.

Для составления таблицы также были взяты значения из Таблиц 1 и 2. Сравнение проводится на основе F1 мера. Результаты можно видеть в таблице 3.

Тесты/Обучение	Английские текста	«Разные» текста	Английские + «разные текста»
Английские текста	0.5338	0.4163	0.4902
«Разные» текста	0.9031	0.5910	0.4933

Таблица 3: значение F1 меры при комбинации английских/мультиязычных датасетов для обучения/тестировки.

Так же проверим влияет ли метод Бенджамини-Хохберга на результаты модели. Протестируем обученную на английских датасетах модель без применения метода

Бенджамини-Хохберга для корректировки p-values. Сравним количество текстов, которые классификатор посчитал сгенерированными. Результаты можно видеть в таблице 4.

Кол-во текстов при обучении/Тип проверки текста на генерированность	С использованием метода Бенджамини-Хохберга	Без использования метода Бенджамини-Хохберга
500	494	988
1000	567	962
2000	473	956

Таблица 4: оценка качества метода Бенджамини-Хохберга

Также оценим качество метода перевода 50% сгенерированных текстов для английского датасета. Обучим модель с применением этого метода и без. Сравнения будут проводиться по мере F1. Результаты можно видеть в таблице 5 (для второго столбца значения были взяты из таблицы 1).

Кол-во текстов при обучении/Метод подготовки датасета	Перевод 50% сгенерированных текстов	Без перевода
500	0.4868	0.4641
1000	0.5338	0.5228
2000	0.5694	0.5535

Таблица 5: оценка качества метода перевода 50% сгенерированных английских текстов на русский

ВЫВОДЫ

Из проведенных экспериментов можно сделать следующие выводы:

1. Построенная модель корректно работает, с увеличением количества тренировочных данных увеличивается качество предсказаний модели о том, сгенерирован ли текст или нет (повышается F1 мера, полнота и точность) (выводы из таблиц 1 и 2).
2. По значениям таблицы 3 можно сделать вывод, что обучение модели на английском датасете, позволяет модели более качественно классифицировать как английские текста, так и текста на «разных» языках. Вместе с тем обучение на смешанном датасете (столбец 4 таблицы 3) не показал сильного улучшения качеств модели.
3. Эксперименты показали, что классификатор на основе трансофрмера кодировщика mDeBERTa обучается лучше и быстрее для текстов на «разных» языках. Это может быть связанно с тем, что вектор-признаки строятся лучше для датасета с «разными» языками.
4. Использование метода Бенджамини-Хохберга действительно улучшает качество предсказаний. Без использования этой меры модель почти все тексты классифицирует как сгенерированные – значение близко к 1000 (при том, что в тестовой выборке только 500 текстов являются сгенерированными). А при

использование этого метода, модель более корректно анализирует сегменты и выносит более точный вердикт о сгенерированности всего текста (выводы из таблицы 4).

5. Метод перевода 50% действительно повышает качество модели, об этом говорит более высокая мера F1 при использовании перевода, чем при его отсутствии (выводы из таблицы 5). Разница небольшая, но это связано с небольшой (по сравнению с исходной) выборкой тренировочных данных.

Таким образом лучшей стратегией построения бинарного классификатора является: дообучение на английском датасете (где 50% сгенерированных текстов переведены на русский), использование метода Бенджамини-Хохберга.

ПРЕДЛОЖЕНИЯ ДЛЯ УЛУЧШЕНИЯ МОДЕЛИ

Предлагаю несколько улучшений для модели бинарной классификации, которые позволят повысить уровень точности и корректности классификации.

- Провести предварительную обработку текста (для тренировочных и тестовых данных), заключающуюся в удалении стоп-слов, удаление неизвестных модели символов. Это приведет входные данные к единому формату, что позволит модели более стандартизованно обучаться и, как следствие, более точно делать предсказания.
- Добавить вектор-признак, отвечающий к какому типу относится поданный на вход документ (научная статья, рассказ, письмо другу), то есть добавит стилистический признак. Такое предложение обосновано тем, что в зависимости от типа документа, человек пишет по-разному, в разном стиле, с разными эмоциями, чего может не быть у генеративных текстов. Это может стать еще одним критерием для классификации.
- Добавить вектор-признак, контролирующий пунктуацию. Человек иногда может допускать ошибки в пунктуации (особенно если текст пишется неформально), забывает ставить запятые, тире и другие знаки препинания. Генеративная модель, в свою очередь, таких ошибок обычно не допускает.
- По аналогии с пунктуацией человек иногда может допускать ошибки в написании слов (ошибки в правописании), а машина обычно их не делает. Можно создать базу слов (словарь), в котором указаны слова с правильным написанием, и по ней построить дополнительный вектор признак правописания слов (также уместно будет применить лемматизацию).
- Также можно классифицировать текст при помощи моделей, построенных на архитектуре разных трансформеров кодировщиков, и после анализировать результаты каждого и выносить вердикт о сгенерированности текста исходя из совокупности выводов моделей.

Источники

[1] ПОИСК ИСКУССТВЕННО СГЕНЕРИРОВАННЫХ ТЕКСТОВЫХ ФРАГМЕНТОВ В НАУЧНЫХ ДОКУМЕНТАХ, 2023 г. Г. М. Грицай, А. В. Грабовой, А. С. Кильдяков, Ю. В. Чехович

[2] Automated Text Identification: Multilingual Transformer-based Models Approach, German Gritsay1, Andrey Grabovoy, Aleksandr Kildyakov and Yury Chekhovich

[3] Dataset: https://github.com/mbzuai-nlp/SemEval2024-task8

[4] Google Translate: https://translate.google.com/?hl=ru

[5] Paraphraser: https://huggingface.co/ramsrigouthamg/t5_sentence_ paraphraser

[6] J. Devlin, M.-W. Chang, K. Lee, K. Toutanova, Bert: Pre-training of deep bidirectional transformers for language understanding, 2019

[7] W.Wang,F.Wei,L.Dong,H.Bao,N.Yang,M.Zhou,Minilm:Deepself-attentiondistillation for task-agnostic compression of pre-trained transformers, 2020.