Детекция галлюцинаций больших языковых моделей

A Preprint

Левыкин Александр Михайлович
Факультет вычислительной математики и кибернетики
МГУ им. Ломоносова
s02210450@gmail.com

Воронцов Константин Вячеславович
Факультет вычислительной математики и кибернетики
МГУ им. Ломоносова
vokov@forecsys.ru

Abstract

В данной работе рассматривается задача детекции галлюцинаций больших языковых моделей. Основное внимание уделяется решению задачи в token classification постановке, в которой требуется классифицировать на наличие галлюцинаций каждый токен ответа модели. Кроме того, фокус направлен на детекцию фактологических галлюцинаций для задачи в reference-free постановке и подход со сведением её к reference-based. Анализируются подходы instruction-based, дообучение NER моделей, а также анализ временных рядов.

1 Введение

Задача детекции галлюцинаций в больших языковых моделях (Large Language Models, LLM) является одной из подзадач обработки естественного языка (NLP), которая направлена на выявление ложных или недостоверных ответов, генерируемых моделями. Галлюцинации могут проявляться в виде неправдивых утверждений, вымышленных данных и фактов, что ограничивает точность и применимость LLM в ряде критически важных задач. Детекция таких ошибок критически важна в областях, где на основе необходима для обеспечения надежности и повышения доверия к моделям в реальных приложениях.

Применение детекции галлюцинаций может быть полезно в следующих областях:

- В создании автоматических систем поддержки решений (например, в медицине или праве) идентификация ложных данных в ответах модели помогает минимизировать риски и избегать принятия решений на основе недостоверной информации [1].
- В вопрос-ответных системах, чат-ботах применение детекции галлюцинаций позволяет улучшить качество и достоверность предоставляемых ответов, помогая пользователям получать более точную информацию [3, 7].
- В широком спектре задач, например, data-to-text generation [13] и суммаризации документов [2].

Галлюцинация LLM — это ответ (или его часть), сгенерированный моделью, который не соответствует входным данным (промпту) или ранее сгенерированному контексту, либо противоречит общеизвестным фактам о мире.

Задачи детекции галлюцинаций в больших языковых моделях можно классифицировать по нескольким основным признакам, включая подходы к детекции, использование эталонных данных и уровень детализации анализа.

1. Детекция галлюцинаций с использованием эталонных данных (reference-based hallucination detection) Одним из распространённых методов является эталонная (reference-based) детекция галлюцинаций, при которой сгенерированный текст сравнивается с имеющимся эталоном. Этот подход успешно применяется в таких задачах, как суммаризации документов [2], машинный перевод [19], генерация текста на основе данных [13] и создание подписей к изображениям. Однако для многих задач генерации текста общего формата эталонные данные могут быть недоступны, что ограничивает применимость таких методов. Например, в диалоговых системах реального времени, таких как чат-боты, модель часто генерирует ответы без возможности сравнения с эталоном.
2. Безэталонная детекция галлюцинаций (reference-free hallucination detection) В условиях, где эталонные данные отсутствуют, используются методы безэталонной детекции галлюцинаций, которые основываются только на контексте и правилах, встроенных в модель [10]. Такие подходы становятся актуальными в системах, работающих в реальном времени (например, чат-боты [3, 7]), где сравнение с эталоном затруднено из-за невозможности получить эталонные данные. Эти методы направлены на определение неконсистентных и потенциально ложных утверждений исключительно по информации, доступной модели в контексте текущего сеанса взаимодействия.

Галлюцинации можно классифицировать по следующей структуре:

- Intrinsic галлюцинации (внутренние галлюцинации) внутри себя делятся на:
  - Input-conflict галлюцинации — это случай, когда генерируемый текст не соответствует запросу пользователя.
  - Context-conflict галлюцинации - случай, когда модель генерирует высказывание, противоречащее ранее сгенерированной информации.
- Extrinsic (fact-conflict) галлюцинации (внешние галлюцинации) — эти галлюцинации определяются различно в зависимости от задачи:
  - генерируемый текст не соответствует фактам реального мира (для referenced-free постановки задачи),
  - генерируемый текст не может быть подтвержден информацией, содержащейся во входных данных - эталонном документе (для reference-based постановки).

Кроме того, подходы к решению задачи детекции галлюцинации можно поделить на два вида по степени детальности обнаружения:

- Уровень предложений или документов: Многие подходы анализируют галлюцинации на уровне предложения или всего документа [20]. Однако такой подход может быть недостаточно точным для выявления конкретных ложных утверждений внутри текста, так как он классифицирует весь текст в целом.
- Детекция на уровне токенов: В альтернативных методах детекция галлюцинаций осуществляется на уровне отдельных токенов [10, 16]. Такой подход позволяет более точно идентифицировать ложные утверждения в тексте и, в некоторых случаях, корректировать сгенерированный текст в реальном времени (например, управляя вероятностью появления тех или иных токенов). Этот метод более эффективен для задач, требующих высокую точность детекции и исправления, таких как системы генерации текста реального времени.

1.1 Цель работы

Reference-free постановка задачи являются более сложной с той точки зрения, что в ней не задан документ-образец, относительно которого идёт поиск галлюцинаций. Из-за этого качество таких моделей ниже. В данной работе исследуются различные решения для reference-free задач (Question Answering и детекция фактологических ошибок) и предлагается подход со сведением её к reference-based постановке. Кроме того, задача решается на уровне токенов, в token-classification постановке, что дополнительно усложняет решение задачи и делает исследование особенно ценным и комплексным.

2 Постановка задачи

Пусть заданы:

- Запрос пользователя (query, промпт, задача) - последовательность токенов $Q = \{q_1, ..., q_m\}$, где m - длина последовательности.
- Ответ модели - последовательность токенов $X = \{x_1, x_2, ..., x_n\}$, где n - длина последовательности.

Задача детекции галлюцинаций заключается в поиске всех пар вида (i_k, j_k), где i_k и j_k, такие что $i_k \leqslant j_k$ - индексы начала и конца текстового фрагмента $\overline{x_{i_k} x_{i_k+1} ... x_{j_k}}$, являющегося галлюцинацией.

Предполагается отсутствие вложенных сущностей (плоская задача NER), поэтому постановка выше эквивалентна следующей:
Задача детекции галлюцинаций заключается в классификации каждого токена ответа $X = \{x_1, x_2, \ldots, x_n\}$ на один из 3-х классов $y_i \in \{B, I, O\}$, где B обозначает начало галлюцинации, I — продолжение галлюцинации, а O — токен, не являющийся частью галлюцинации.

3 Предложенный метод

3.1 Instruction-based подход

В предложенном подходе для детекции галлюцинаций используется instruction-based метод, при котором большая языковая модель (LLM) генерирует текст, явно указывая на галлюцинации в ответе. В данном эксперименте вместо вывода вероятностей или меток на уровне токенов модель на естественном языке выделяет фрагменты текста, которые, по её мнению, являются галлюцинациями.

Одним из ключевых свойств крупных языковых моделей является способность решать задачи, на которые они не были специально обучены, используя инструкции (промпты), формирующие поведение модели в ходе генерации текста.
Исследования по подбору промпта широко развились [14, 9] и получили термин “prompt engineering”, а способность генеративных моделей проявлять умения, которые от них не требовались при обучении, получила название эмерджентным поведением (emergent behavior) и повлекло множество исследований [18, 12].

Формирование промпта для явной индикации галлюцинаций: Данные в задаче представляют из себя наборы пар: запрос пользователя $Q = \{q_1, q_2, \ldots, q_m\}$ и ответ модели $X = \{x_1, x_2, \ldots, x_n\}$. Задача состоит в создании промпта P, который направит модель на явное указание фрагментов, являющихся галлюцинациями, и покажет наилучшее качество по описанным ниже метрикам.

Преимущества и недостатки instruction-based метода

- Преимущества:
  - Адаптивность: позволяет модели решать задачи, на которые она не была специально обучена, благодаря использованию промптов.
  - Экономичность: отсутствует необходимость в дообучении модели, что снижает затраты на дополнительные данные и вычислительные ресурсы.
  - Универсальность: подходит для широкого спектра задач, где можно управлять поведением модели через промпты.
  - Интерпретируемость: модель может выдавать галлюцинации в явном виде, делая выводы более прозрачными для пользователей.
- Недостатки:
  - Чувствительность к формулировке промпта: результат может сильно зависеть от точной формулировки запроса, что требует времени на подбор оптимальных инструкций.
  - Отсутствие гарантий точности: модель может ошибаться в детекции галлюцинаций, так как метод не обеспечивает вероятностных оценок надёжности.
  - Ограниченная гибкость: сложно управлять степенью уверенности модели в предсказаниях без дополнительных вероятностных данных.
  - Зависимость от масштабов модели: для эффективного выполнения задачи могут требоваться крупные модели, что увеличивает вычислительные затраты.

3.2 Обучение рекуррентной нейросети LSTM

Для данных, в которых помимо сгенерированного ответа модели известны логиты (или вероятности) каждого токена, можно провести эксперименты с обучением моделей, анализирующих лишь временной ряд логитов. В данном случае данные представляют собой последовательность логитов и соответствующую последовательность истинных меток, которые модель должна предсказать.

Архитектура модели включает слой LSTM, после которого добавляется линейный слой, понижающий размерность до 1, что позволяет получить вероятность для каждого токена. Модель обучается с использованием функции потерь BCE (Binary Cross-Entropy Loss), которая минимизируется по следующей формуле:

$$\text{BCE Loss} = -\frac{1}{N}\sum_{i=1}^{N}\left(y_i \log(\hat{y}_i) + (1 - y_i)\log(1 - \hat{y}_i)\right)$$

где:

- N — количество токенов в обучающей выборке,
- y_i — истинная метка для токена i (1 для галлюцинации, 0 для корректного токена),
- $\hat{y}_i$ — предсказанная моделью вероятность того, что токен i является галлюцинацией.

Таким образом, модель на базе LSTM обучается на последовательности логитов для детекции галлюцинаций, опираясь на временные зависимости между логитами.

Плюсы и минусы подхода
Плюсы:

- Использование только логитов позволяет модели анализировать временные зависимости без учёта лексического содержания, что снижает влияние языковых особенностей и позволяет концентрироваться на статистических особенностях.
- Архитектура LSTM позволяет учитывать долгосрочные зависимости в последовательностях, что полезно для обнаружения скрытых закономерностей в логитах, связанных с галлюцинациями.

Минусы:

- Отсутствие информации о самих токенах может ограничивать точность, так как модель не учитывает семантический контекст слов.
- Обучение на логитах может быть более подвержено шуму и нестабильности, особенно если логиты не отражают чёткой разницы между галлюцинациями и корректными ответами.
- Для эффективного обучения может потребоваться большой объём данных, так как модель анализирует только числовые представления (логиты) без дополнительных подсказок из текста.

Модель LSTM была представлена в 1997 году [4] и получила широкую известность и область применения. Схема её архитектуры и основные слои представлены на рис. 1

3.3 Дообучение NER моделей

Задача выделения фрагментов галлюцинаций в тексте представляет собой задачу классификации токенов, в которой каждому токену присваивается метка, указывающая, является ли он частью галлюцинации или нет. Для решения этой задачи используется формат BIO-разметки (отмечающий начало, середину и конец фрагментов, содержащих галлюцинации). Формат BIO позволяет более точно определять начало и границы фрагментов и показывает лучшие результаты в сравнении с форматом бинарной классификации каждого токена. В данном разделе мы подробно опишем задачу обучения модели и введем основные обозначения.

Формализация задачи

Рассмотрим текст, состоящий из N токенов, представленных последовательностью $X = (x_1, x_2, \ldots, x_N)$. Модель должна классифицировать каждый токен x_i как принадлежащий галлюцинации или нет, используя BIO-формат, где:

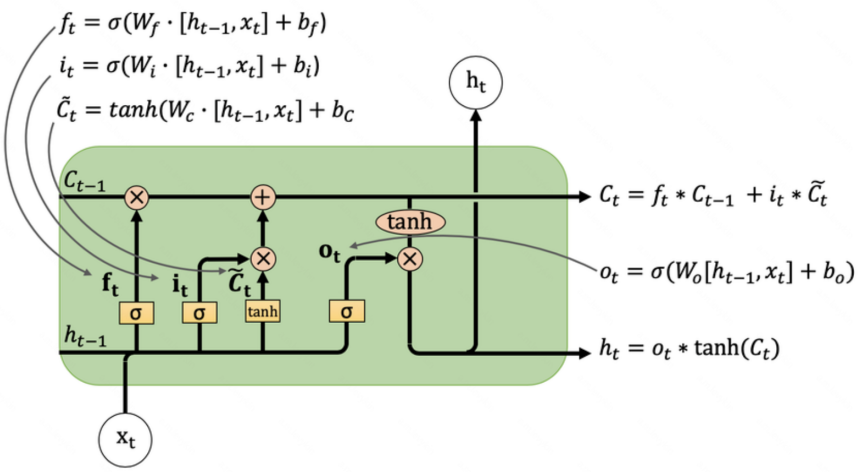

Рис. 1: Архитектура нейросети LSTM

- B обозначает начало фрагмента галлюцинации,
- I обозначает токен, продолжающий галлюцинацию,
- O обозначает токен, не связанный с галлюцинацией.

Для этого модель обучается предсказывать вероятности $\hat{S}$ для каждого токена текста.

Обозначения матриц меток и предсказаний:

$S \in \{0,1\}^{N \times 3}$: бинарная матрица, содержащая истинные метки для каждого токена, где:

- $S_{i,0} = 1$, если токен x_i отмечен как B,
- $S_{i,1} = 1$, если токен x_i отмечен как I,
- $S_{i,2} = 1$, если токен x_i отмечен как O.

$\hat{S} \in [0,1]^{N \times 3}$: матрица предсказаний модели, где $\hat{S}_{i,t}$ обозначает вероятность того, что токен x_i относится к классу $t \in \{B, I, O\}$.

Ограничения нормировки:

Для каждого токена модель должна предсказывать распределение вероятностей по трем меткам, что выражается следующим условием:

$$\sum_t \hat{S}_{i,t} = 1, \quad \forall i \in \{1, \ldots, N\}$$

Функция потерь — кросс-энтропия:

Для оптимизации параметров span-detection моделей используется функция потерь на основе кросс-энтропии, которая минимизируется по параметрам модели θ':

$$CE(S, \hat{S}) = -\frac{1}{N} \sum_{i=1}^{N} \sum_{t \in \{B,I,O\}} S_{i,t} \log(\hat{S}_{i,t})$$

Здесь: - $S_{i,t}$ — бинарный индикатор для истинной метки токена x_i по классу t, - $\hat{S}_{i,t}$ — вероятность, предсказанная моделью для метки t токена x_i.

И задача обучения модели заключается в поиске параметров, доставляющих минимум функции потерь:

$$CE(S, p(S \mid \theta)) = CE(S, \hat{S}) = -\frac{1}{N} \sum_{i=1}^{N} \sum_{t \in \{B,I,O\}} S_{i,t} \log(\hat{S}_{i,t}) \longrightarrow \min_{\theta}$$

3.4 Методика оценивания модели

Обозначения для i-го класса:

- True Positive (TP_i): количество токенов или фрагментов, которые модель правильно классифицировала как принадлежащие классу i. Определяется как количество случаев, в которых предсказанный класс и истинный класс равны i:

$$TP_i = \sum_{j=1}^{N} \mathbb{I}(\hat{y}_j = i \wedge y_j = i)$$

  где N — общее количество токенов, $\hat{y}_j$ — предсказанный класс для токена j, y_j — истинный класс для токена j, а $\mathbb{I}(\cdot)$ — индикаторная функция, принимающая значение 1, если условие выполняется, и 0 в противном случае.

- False Positive (FP_i): количество токенов или фрагментов, которые модель ошибочно классифицировала как принадлежащие классу i. Определяется как количество случаев, в которых предсказанный класс равен i, но истинный класс не равен i:

$$FP_i = \sum_{j=1}^{N} \mathbb{I}(\hat{y}_j = i \wedge y_j \neq i)$$

- False Negative (FN_i): количество токенов или фрагментов, которые модель ошибочно не отнесла к классу i. Определяется как количество случаев, в которых истинный класс равен i, но предсказанный класс не равен i:

$$FN_i = \sum_{j=1}^{N} \mathbb{I}(\hat{y}_j \neq i \wedge y_j = i)$$

Метрики для i-го класса:

1. Точность (Precision) для i-го класса: Точность показывает, какая доля токенов, предсказанных моделью как относящиеся к классу i, действительно относятся к этому классу.

$$\text{Precision}_i = \frac{TP_i}{TP_i + FP_i}$$

2. Полнота (Recall) для i-го класса: Полнота показывает, какую долю токенов класса i модель правильно предсказала, по отношению к общему числу токенов данного класса в данных.

$$\text{Recall}_i = \frac{TP_i}{TP_i + FN_i}$$

3. F1-score для i-го класса: F1-score является гармоническим средним точности и полноты и даёт общую оценку качества предсказания для класса i.

$$\text{F1}_i = 2 \cdot \frac{\text{Precision}_i \cdot \text{Recall}_i}{\text{Precision}_i + \text{Recall}_i}$$

Общие метрики для всех классов:

4. F1-macro (макроусреднённый F1-score): F1-macro вычисляется как среднее значение F1-score по всем классам, где каждый класс имеет одинаковый вес. Это полезно, когда важно учитывать качество предсказаний для каждого класса независимо от его размера.

$$\text{F1-macro} = \frac{1}{C} \sum_{i=1}^{C} \text{F1}_i$$

где C — количество классов.

5. F1-micro (микроусреднённый F1-score): F1-micro учитывает общие суммы TP, FP и FN по всем классам и вычисляет F1 на основе общей точности и полноты. Он полезен, когда важен вклад каждого примера, вне зависимости от размера класса.

$$\text{F1-micro} = \frac{2 \cdot \sum_{i=1}^{C} TP_i}{2 \cdot \sum_{i=1}^{C} TP_i + \sum_{i=1}^{C} FP_i + \sum_{i=1}^{C} FN_i}$$

Критерий IoU (Intersection over Union, пересечение над объединением) используется для оценки схожести предсказанных и истинных фрагментов. IoU измеряет долю пересечения между предсказанным фрагментом и истинным фрагментом от их объединения. Формула IoU определяется как:

$$\text{IoU} = \frac{|A \cap B|}{|A \cup B|}$$

где:

- A — множество токенов, принадлежащих истинному фрагменту галлюцинации.
- B — множество токенов, принадлежащих предсказанному фрагменту галлюцинации.
- $|A \cap B|$ — количество токенов, которые находятся одновременно в истинном и предсказанном фрагментах.
- $|A \cup B|$ — количество токенов, которые находятся хотя бы в одном из фрагментов (их объединение).

Для задачи с несколькими фрагментами галлюцинаций метрика IoU может быть усреднена по всем парам предсказанных и истинных фрагментов, чтобы получить средний IoU.

4 Вычислительный эксперимент

4.1 Данные

4.1.1 SemEval-2025

В рамках активного соревнования Semeval-2025 (https://helsinki-nlp.github.io/shroom/) на момент написания статьи доступна размеченная валидационная выборка. Она состоит из 10 файлов на разных языках. В каждом файле 50 объектов-троек: вопрос-ответ-размеченные галлюцинации.

4.1.2 HaluEval

Датасет HaluEval был дополнительно переразмечен в формат классификации токенов с помощью GPT-4. В итоговом датасете содержится 10 000 объектов, представляющих собой пары (вопрос, ответ, фрагменты галлюцинации).

4.2 Instruction-based подход

Для проведения эксперимента выбрана модель Meta-Llama-3.1-8B-Instruct. Модель была представлена в 2024 году, обучалась на данных до декабря 2023 года и содержит одни из самых актуальных знаний среди больших языковых моделей, а также показывает высокие результаты на бенчмарках. Кроме того, платформа kaggle с видеокартой Nvidia P100 с 16 Gb видеопамяти добавляет ограничения на размер модели, и Llama-3.1-8B занимает всю доступную видеопамять. Так, для экспериментов была выбрана эта модель как наиболее актуальная и мощная из возможных для запуска на kaggle.

Для эксперимента взяты 50 объектов на английском языке из валидационной выборки соревнования SemEval-2025 и проведено множество экспериментов с промптами. Наилучший результат достигнут для следующего промпта:

```
messages = [ {‘‘role’’: ‘‘system’’, ‘‘content’’: ‘‘You are a fact-checking assistant. Your task
is to identify fragments of the response that are hallucinations - parts of the text
that are factually incorrect or made up by model. Pay attention to facts, dates, numbers,
places. Detect only hallucination words, without neighbour words. Give me only a list
```

`of fragments-hallucinations you found in model output.''}, {``role'': ``user'', ``content'': formatted_str},]`

В `formatted_str` подавалась строка следующего вида:
`formatted_str = ``query'':``{data[``model_input'']}'' ``model_output'': ``{data[``model_output_text'']}''`

Для оценки качества по регламенту конкурса используется метрика IoU.

Результаты: распределение IoU на объектах в данном эксперименте представлено на рис. 2. Средний IoU составил 39.7%.

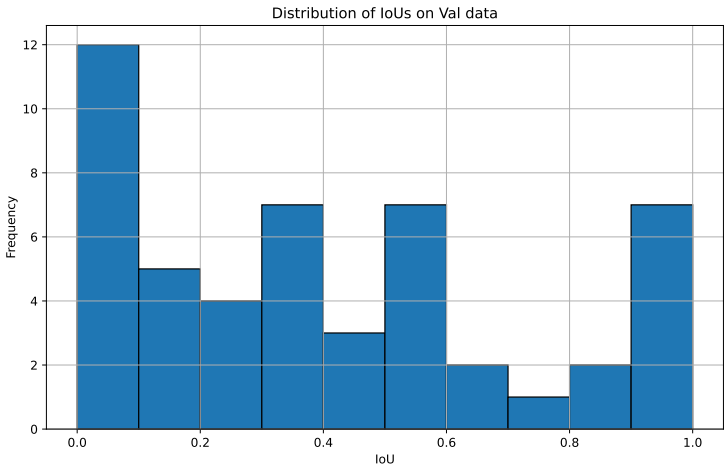

Рис. 2: распределение IoU на объектах SemEval-2025.

4.3 Обучение рекуррентной нейросети LSTM

Для проведения эксперимента была выбрана реализация модели LSTM из библиотеки torch: `torch.nn.LSTM`. На выходы LSTM добавляется линейный слой `torch.nn.Linear`, понижающий размерность до 1. Аналогично предыдущему эксперименту, использовалась платформа Kaggle.

В качестве данных взята валидационная выборка на всех языках от конкурса SemEval-2025. Выборка составила 500 объектов. Выборка поделена на обучающую и тестовую в соотношении 9:1. Обучение длилось 50 эпох.

Кривые процесса обучения представлены на графиках 3, 5. Качество обученной модели на валидационной выборке IoU = 15.5%.

4.4 Дообучение NER модели

В качестве данных использовался набор HaluEval, переразмеченный в формат классификации токенов с помощью GPT-4. В итоговом датасете содержится 10 000 объектов, представляющих собой пары (вопрос, ответ, фрагменты галлюцинации). Данные были разделены на обучающую и тестовую выборки в соотношении 9:1.

Для обучения модели `distilbert-base-uncased` использовались следующие гиперпараметры:

- Количество эпох: `num_train_epochs` = 3
- Количество шагов разогрева: `warmup_steps` = 500
- Коэффициент затухания весов: `weight_decay` = 0.01
- Размер батча на устройство: `per_device_train_batch_size` = 16

Качество обученной модели было оценено на тестовой выборке. Основные метрики приведены ниже:

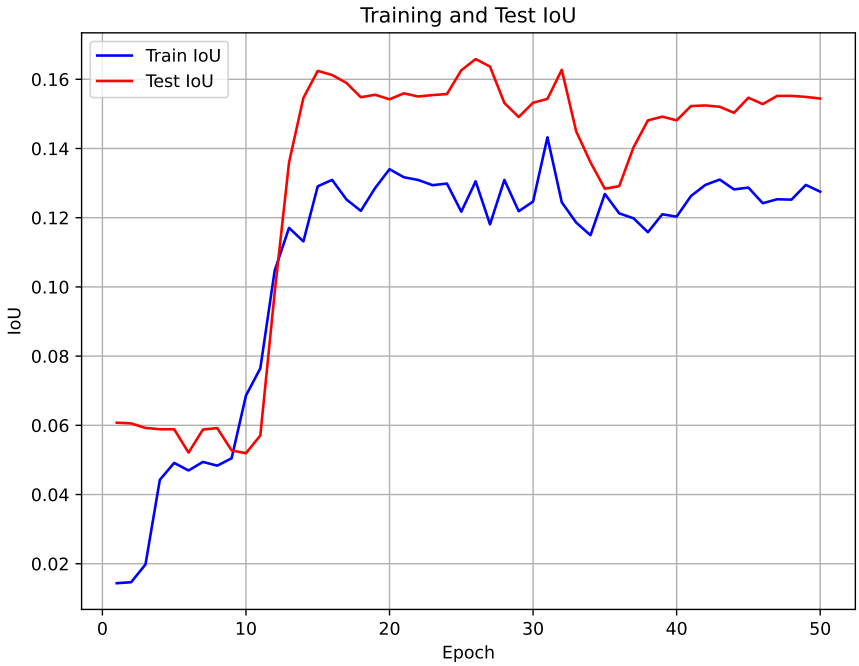

Рис. 3: График зависимости IoU от эпохи на обучающей и тестовой выборках.

- Accuracy: 0.94
- F1 Macro: 0.810
- F1 Micro: 0.939
- IoU: 0.711

Процесс обучения представлен на графиках. На Рисунке 5 показана динамика функции потерь BCE на обучающей выборке, а на Рисунке 6 — на валидационной выборке.

Для более детального анализа работы модели был сформирован классификационный отчёт, представленный в Таблице 1.

	Precision	Recall	F1-Score	Support
0	0.95	0.98	0.97	34944
1	0.77	0.57	0.65	3911
Macro Avg	0.86	0.77	0.81	38855
Weighted Avg	0.93	0.94	0.94	38855

Таблица 1: Классификационный отчёт

4.5 Сравнение рассмотренных методов

В таблице 2 представлены результаты работы моделей LSTM, Llama-3.1-8B и DistilBERT на различных датасетах. Для оценки использованы метрики F1 Macro и IoU.

Анализ таблицы 2 позволяет сделать следующие выводы:

- Устойчивость данных: Модель Llama-3.1-8B показала наилучшую устойчивость к различным форматам данных, превосходя остальные модели на датасетах SemEval-500 и SemEval-50.

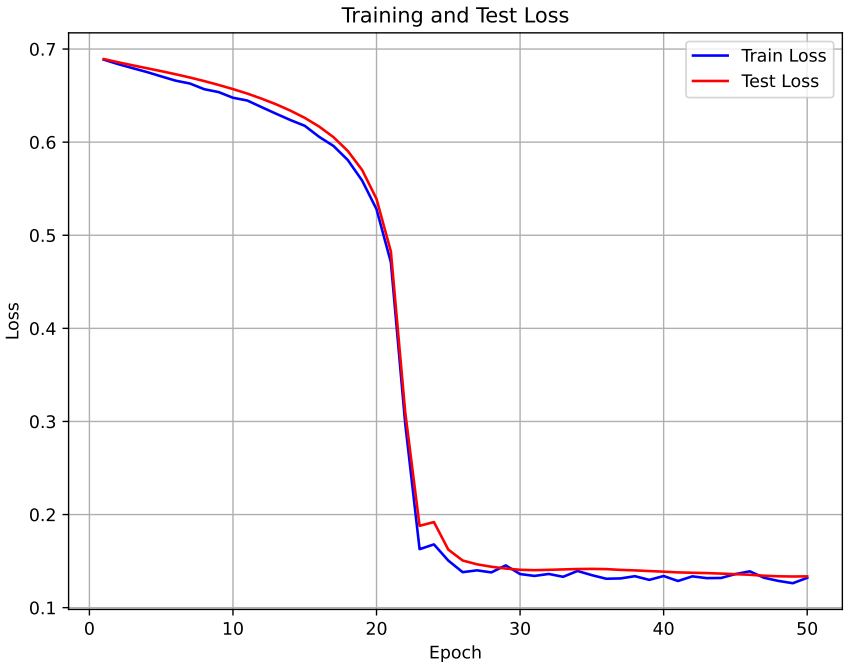

Рис. 4: График зависимости BCE Loss от эпохи на обучающей и тестовой выборках.

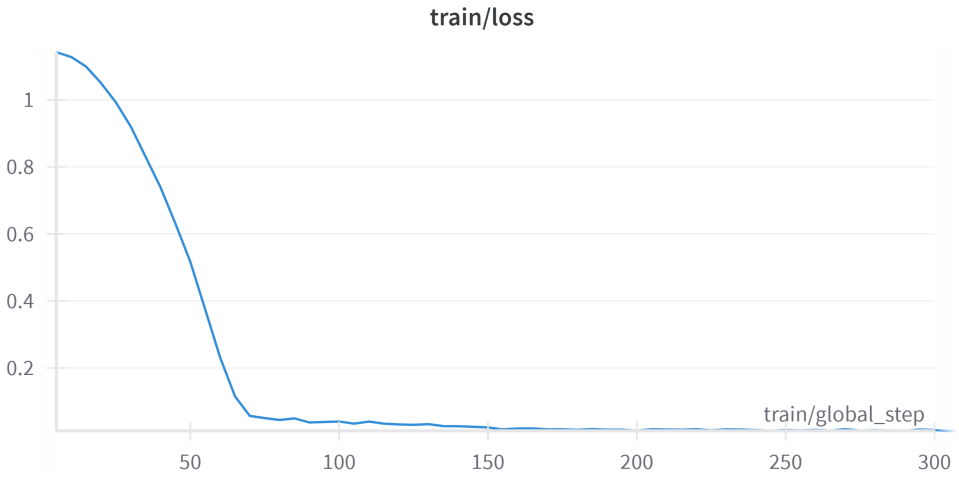

Рис. 5: BCE Loss на обучающей выборке.

- Специализация на обучающей выборке: Модель DistilBERT превосходит Llama-3.1-8B по метрикам на датасете HaluEval, схожем с обучающей выборкой, но демонстрирует худшие результаты на других датасетах, что указывает на меньшую способность к обобщению.

- Производительность LSTM: Модель LSTM существенно уступает современным подходам по качеству, что делает её использование нецелесообразным.

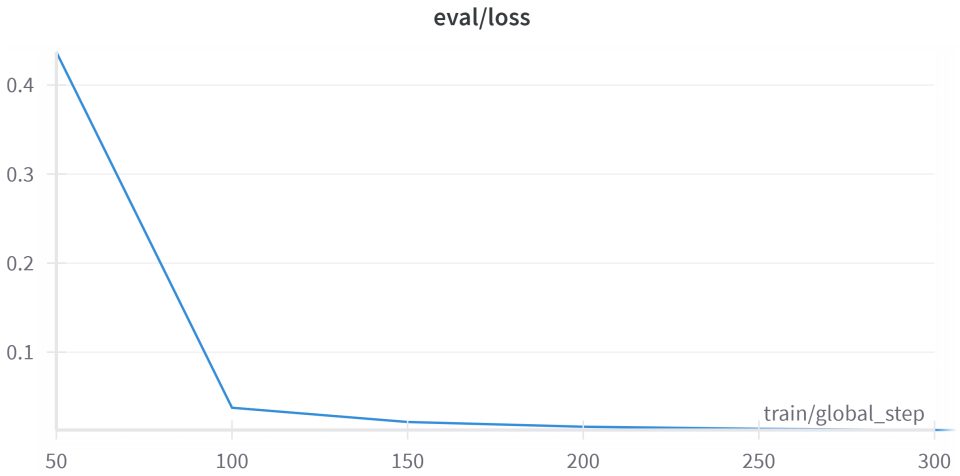

Рис. 6: BCE Loss на валидационной выборке.

Модель	HaluEval (val)	SemEval-500	SemEval-50 (Eng)
LSTM	–	0.23 / 0.155	0.25 / 0.19
Llama-3.1-8B	0.722 / 0.693	0.398 / 0.375	0.421 / 0.397
DistilBERT	0.810 / 0.711	0.303 / 0.281	0.361 / 0.332

Таблица 2: F1 Macro и IoU для всех моделей на различных датасетах

5 Заключение

В данной работе проведено исследование методов детекции галлюцинаций в token classification постановке с акцентом на фактологические галлюцинации в reference-free задачах. В рамках исследования были проанализированы и сравнены три подхода: instruction-based использование больших языковых моделей (LLM), дообучение моделей NER, а также анализ временных рядов. Сравнительный анализ подтвердил, что успешное решение reference-free задач требует компромисса между качеством детекции, вычислительной эффективностью и способностью модели к обобщению. Instruction-based подходы предоставляют универсальность и высокое качество, однако требуют значительных ресурсов. В то же время компактные NER модели могут быть эффективными в задачах с качественно размеченными обучающими выборками, оставаясь более лёгкими для развёртывания. Дальнейшие исследования будут направлены на построение и сравнение RAG (retrieval augmented generation) систем, объединяющих векторные базы данных и модели детекции.

Список литературы

[1] Agarwal, V., Jin, Y., Chandra, M., Choudhury, M.D., Kumar, S., Sastry, N.: Medhalu: Hallucinations in responses to healthcare queries by large language models. arXiv preprint arXiv:2409.19492 (2024), https://arxiv.org/abs/2409.19492

[2] Cao, M., Dong, Y., Cheung, J.C.K.: Hallucinated but factual! inspecting the factuality of hallucinations in abstractive summarization. arXiv preprint arXiv:2109.09784 (2021), https://arxiv.org/abs/2109.0f9784

[3] Dziri, N., Madotto, A., Zaiane, O., Bose, A.J.: Neural path hunter: Reducing hallucination in dialogue systems via path grounding. arXiv preprint arXiv:2104.08455 (2021), https://arxiv.org/abs/2104.08455

[4] Hochreiter, S., Schmidhuber, J.: Long short-term memory. Neural computation 9, 1735–80 (12 1997). doi:10.1162/neco.1997.9.8.1735

[5] Hochreiter, S., Schmidhuber, J.: Long short-term memory. Neural Computation 9(8), 1735–1780 (1997). doi:10.1162/neco.1997.9.8.1735

[6] Ji, Z., Lee, N., Frieske, R., Yu, T., Su, D., Xu, Y., Ishii, E., Bang, Y., Chen, D., Dai, W., Chan, H.S., Madotto, A., Fung, P.: Survey of hallucination in natural language generation. arXiv preprint arXiv:2202.03629 (2024), https://arxiv.org/abs/2202.03629

[7] Li, J., Cheng, X., Zhao, W.X., Nie, J.Y., Wen, J.R.: Halueval: A large-scale hallucination evaluation benchmark for large language models. arXiv preprint arXiv:2305.11747 (2023), https://arxiv.org/abs/2305.11747

[8] Li, M., Peng, B., Galley, M., Gao, J., Zhang, Z.: Self-checker: Plug-and-play modules for fact-checking with large language models. arXiv preprint arXiv:2305.14623 (2024), https://arxiv.org/abs/2305.14623

[9] Liu, P., Yuan, W., Fu, J., Jiang, Z., Hayashi, H., Neubig, G.: Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing (2021), https://arxiv.org/abs/2107.13586

[10] Liu, T., Zhang, Y., Brockett, C., Mao, Y., Sui, Z., Chen, W., Dolan, B.: A token-level reference-free hallucination detection benchmark for free-form text generation. arXiv preprint arXiv:2104.08704 (2021), https://arxiv.org/abs/2104.08704

[11] Manakul, P., Liusie, A., Gales, M.J.F.: Selfcheckgpt: Zero-resource black-box hallucination detection for generative large language models. arXiv preprint arXiv:2303.08896 (2023), https://arxiv.org/abs/2303.08896

[12] OpenAI, Achiam, J., et al., S.A.: Gpt-4 technical report. arXiv preprint arXiv:2303.08774 (2024), https://arxiv.org/abs/2303.08774

[13] Rebuffel, C., Roberti, M., Soulier, L., Scoutheeten, G., Cancelliere, R., Gallinari, P.: Controlling hallucinations at word level in data-to-text generation. arXiv preprint arXiv:2102.02810 (2021), https://arxiv.org/abs/2102.02810

[14] Reynolds, L., McDonell, K.: Prompt programming for large language models: Beyond the few-shot paradigm (2021), https://arxiv.org/abs/2102.07350

[15] Sansford, H., Richardson, N., Maretic, H.P., Saada, J.N.: Grapheval: A knowledge-graph based llm hallucination evaluation framework. arXiv preprint arXiv:2407.10793 (2024), https://arxiv.org/abs/2407.10793

[16] Wang, S., Wang, X., Mei, J., Xie, Y., Muarray, S., Li, Z., Wu, L., Chen, S.Q., Xiong, W.: Developing a reliable, general-purpose hallucination detection and mitigation service: Insights and lessons learned. arXiv preprint arXiv:2407.15441 (2024), https://arxiv.org/abs/2407.15441

[17] Wei, J., Tay, Y., Bommasani, R., Raffel, C., Zoph, B.: Emergent abilities of large language models. arXiv preprint arXiv:2206.07682 (2022), https://arxiv.org/abs/2206.07682

[18] Wei, J., Tay, Y., Bommasani, R., Raffel, C., Zoph, B., Borgeaud, S., Yogatama, D., Bosma, M., Zhou, D., Metzler, D., Chi, E.H., Hashimoto, T., Vinyals, O., Liang, P., Dean, J., Fedus, W.: Emergent abilities of large language models (2022), https://arxiv.org/abs/2206.07682

[19] Zha, Y., Yang, Y., Li, R., Hu, Z.: Alignscore: Evaluating factual consistency with a unified alignment function. arXiv preprint arXiv:2305.16739 (2023), https://arxiv.org/abs/2305.16739

[20] Zhao, T., Wei, M., Preston, J.S., Poon, H.: Automatic calibration and error correction for large language models via pareto optimal self-supervision. arXiv preprint arXiv:2306.16564 (2024), https://arxiv.org/abs/2306.16564

6 Приложение