

# Детекция галлюцинаций больших языковых моделей

Левыкин Александр Михайлович

Научный руководитель: Воронцов Константин Вячеславович

МГУ им. М.В.Ломоносова

14 декабря 2024 г.

# Понятие галлюцинации и их классификация

- **Цель работы:** исследование методов детекции галлюцинаций в token classification постановке. Фокус на детекции фактологических галлюцинаций для задачи в reference-free постановке. Анализируются подходы instruction-based, дообучение NER моделей, а также анализ временных рядов.
- **Галлюцинация LLM** - ответ (или его часть) модели, не соответствующий входным данным или сгенерированному ранее контексту, либо противоречащий общеизвестным фактам о мире.
- Галлюцинации делятся на две части:
  - **Intrinsic** (внутренние) галлюцинации - генерируемый текст не соответствует входным данным.
  - **Extrinsic** (внешние) галлюцинации -
    - ответ не соответствует фактам реального мира (в задачах без референсного документа)
    - ответ не может быть подтвержден информацией, содержащейся во входных данных (в задачах с референсным документом)

# Классификация задач детекции галлюцинаций

Подходы детекции галлюцинаций бывают:

- **reference-based** - когда есть документ-источник (суммаризация, переводчики, image caption и т.д.)
- **reference-free** (question answering - наша задача)

Подходы детекции галлюцинаций бывают:

- **online** - при классификации  $i$ -го токена можно смотреть только на  $1, \dots, i$ -е токены.
- **offline** - можно смотреть на весь текст.

Подходы детекции галлюцинаций бывают:

- **document-level** - классифицируем весь текст одной меткой.
- **token-level** - классифицируем каждый токен по-отдельности.

В данной работе - фокус на **token-level reference-free** задаче.

# Постановка задачи

- Пусть заданы:
  - Запрос пользователя - последовательность токенов  $Q = \{q_1, \dots, q_m\}$ .
  - Ответ модели - последовательность токенов  $X = \{x_1, x_2, \dots, x_n\}$ .
- Задача: поиск всех пар вида  $(i_k, j_k)$ , где  $i_k$  и  $j_k$ , такие что  $i_k \leq j_k$  - индексы начала и конца текстового фрагмента  $\overline{x_{i_k} x_{i_k+1} \dots x_{j_k}}$ , являющегося галлюцинацией.

Предполагается отсутствие вложенных сущностей, поэтому постановка выше эквивалентна следующей:

- Задача: классификация каждого токена ответа  $X = \{x_1, x_2, \dots, x_n\}$  на один из 3-х классов  $y_i \in \{B, I, O\}$ , где  $B$  - начало галлюцинации,  $I$  — продолжение, а  $O$  — токен, не являющийся частью галлюцинации.

# Предложенный метод 1. Instruction-based подход

Метод опирается на “prompt engineering”, и “emergent behavior”.

Данные в задаче - наборы пар: запрос пользователя  $Q = \{q_1, q_2, \dots, q_m\}$  и ответ модели  $X = \{x_1, x_2, \dots, x_n\}$ .

**Подход:** создать промпт  $P$  для генеративной LLM, на котором достигается максимум по метрикам качества на валидационной выборке.

**Преимущества:**

- **Адаптивность:** подбор промптов позволяет решать широкий класс задач.
- **Экономичность:** не нужно дообучать модель, что снижает затраты на дополнительные данные и вычислительные ресурсы.
- **Интерпретируемость:** промптом можно потребовать пояснение галлюцинации.

**Недостатки:**

- **Чувствительность к промпту**
- **Отсутствие вероятностей**  $\Rightarrow$  сложно управлять степенью уверенности.
- **Зависимость от масштабов модели:** требуются очень крупные модели, что увеличивает временные затраты.

# Instruction-based подход. Эксперимент

**Данные:** SemEval-2025 Validation Set на английском языке (50 троек: (вопрос, ответ, фрагменты\_галлюцинации)).

**Модель:** Meta-Llama-3.1-8B-Instruct (release - 2024).

**Лучший результат:** 39.7% по IoU.

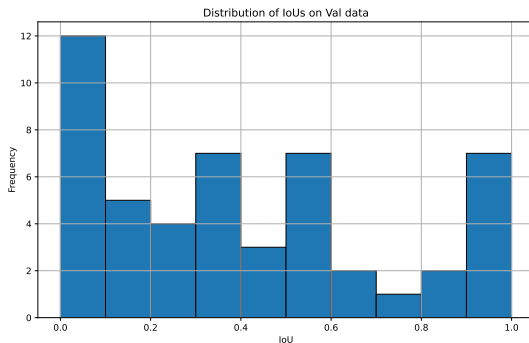


Рис.: Распределение IoU на объектах Val выборки

# Предложенный метод 2. Обучение LSTM

В датасете SemEval-2025 известны логиты каждого токена  $\Rightarrow$  можно обучить модель, анализирующую лишь временной ряд логитов.

**Архитектура:** LSTM  $\rightarrow$  MLP.

**Обучение:** минимизация функции потерь BCE (Binary Cross-Entropy Loss):

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

где:

- $N$  — количество токенов в обучающей выборке,
- $y_i$  — истинная метка для токена  $i$ ,
- $\hat{y}_i$  — предсказанная моделью вероятность  $p(y_i = 1)$ ,
- $\Theta$  — параметры модели.

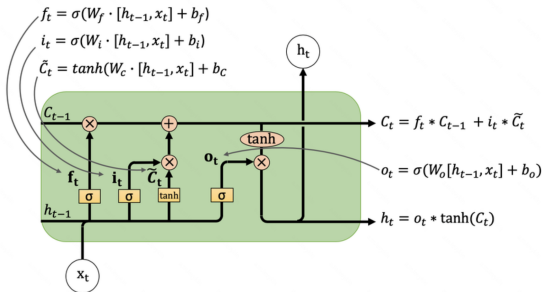
# Предложенный метод 2. Обучение LSTM

## Плюсы подхода:

- **Свобода от языка:** модель анализирует временные зависимости без учёта лексики  $\Rightarrow$  ниже зависимость от языка, концентрация на статистике.

## Минусы подхода:

- **Потеря информации:** снижение точности, т.к. модель не учитывает семантику слов.
- **Слабость модели:** модель была представлена в 1997 году и уже давно не является SOTA.





# Обучение LSTM. Эксперимент

**Данные:** SemEval-2025 Validation Set на 10 языках (500 троек: (вопрос, ответ, фрагменты\_галлюцинации)).

**Модель:** `torch.nn.LSTM`  $\rightarrow$  `torch.nn.Linear`.

**Setup:** Train/Test = 9:1, `hidden_size` = 2048. `n_epochs` = 50.

**Результат:** 15.5% по IoU на Test.

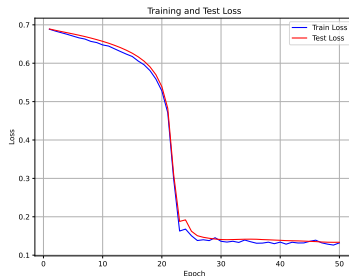
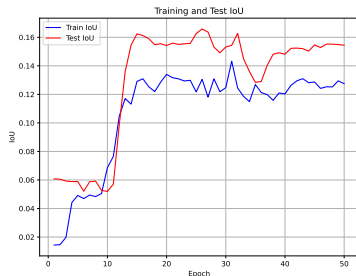


Рис.: Зависимость IoU и BCE Loss от эпохи обучения на Train и Test.

# Предложенный метод 3. Дообучение NER модели

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  $t \in \{B, I, O\}$ .

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

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

# Дообучение NER модели. Эксперимент

**Данные:** HaluEval с переразметкой в token classification формат с помощью GPT-4 (10000 объектов: (вопрос, ответ, фрагменты\_галлюцинации)).

**Модель:** distilbert-base-uncased.

## Setup:

- Train/Test = 9:1,
- Обучение - 3 эпохи.
- warmup\_steps=500
- weight\_decay=0.01,
- per\_device\_train\_batch\_size=16,

## Качество обученной модели на Test Set:

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

# Дообучение NER модели. Эксперимент

Процесс обучения представлены на графиках



Рис.: BCE Loss на обучающей выборке.

# Дообучение NER модели. Эксперимент

Процесс обучения представлены на графиках

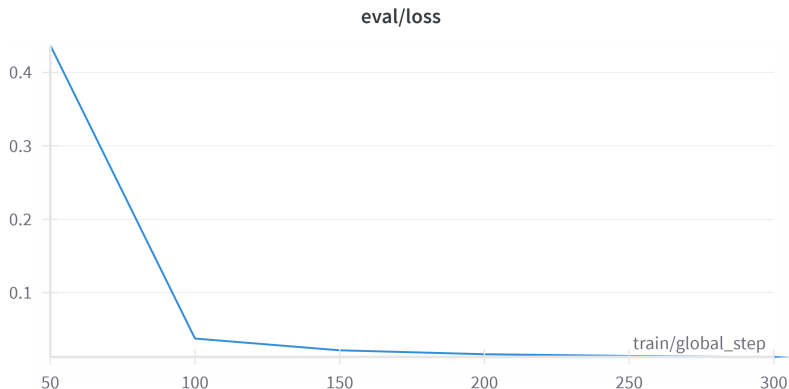


Рис.: BCE Loss на валидационной выборке.

# Дообучение NER модели. Результаты эксперимента

**Модель:** distilbert-base-uncased.

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

Таблица: Classification Report

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

# Сравнение качества работы моделей на всех датасетах

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





Таблица: F1 Macro & IoU для всех моделей по всем датасетам на их валидационных частях

## Выводы:

- Наиболее устойчивой к формату данных является Llama-3.1-8B
- На выборке, схожей с обучающей, DistilBERT превосходит по метрикам большую генеративную Llama-3.1-8B, но хуже обобщается под другие датасеты.
- LSTM показывает качество несопоставимо хуже остальных подходов.

- Наиболее удобным, устойчивым к формату данных, быстрым в реализации является prompt-engineering LLM
- При всех преимуществах prompt-engineering работает крайне медленно на inference.
- При качественной обучающей выборке NER метод приближается по метрикам к большим LLM.
- LSTM показывает наибольшую скорость, но качество несопоставимо хуже остальных подходов, из-за чего подход неактуален.
- Дальнейшие исследования будут связаны с построением RAG систем.



-  Hochreiter, Sepp и Jürgen Schmidhuber (дек. 1997). “Long Short-term Memory”. В: *Neural computation* 9, с. 1735—80. DOI: 10.1162/neco.1997.9.8.1735.
-  OpenAI и др. (2024). *GPT-4 Technical Report*. arXiv: 2303.08774 [cs.CL]. URL: <https://arxiv.org/abs/2303.08774>.
-  Reynolds, Laria и Kyle McDonell (2021). *Prompt Programming for Large Language Models: Beyond the Few-Shot Paradigm*. arXiv: 2102.07350 [cs.CL]. URL: <https://arxiv.org/abs/2102.07350>.
-  Wei, Jason и др. (2022). *Emergent Abilities of Large Language Models*. arXiv: 2206.07682 [cs.CL]. URL: <https://arxiv.org/abs/2206.07682>.