# 1 Принцип работы LongLLMLingua

LongLLMLingua представляет собой метод сжатия длинных промптов для языковых моделей, который работает в несколько этапов:

1. **Постановка задачи**

Задача формулируется как оптимизационная проблема:

$$\min_{\tilde{x}} D_\phi(y, \tilde{y}) + \lambda \|\tilde{x}\|_0$$

где необходимо найти сжатый промпт $\tilde{x}$, минимизирующий расстояние между ответами языковой модели на исходный ($y$) и сжатый ($\tilde{y}$) промпты при сохранении заданной степени сжатия.

2. **Крупнозернистое сжатие**

На первом этапе оценивается важность каждого документа относительно заданного вопроса:

$$r_k = -\frac{1}{N_c} \sum_i \log p(x_i^{que,restrict} | x_i^{doc})$$

где $x_i^{que,restrict}$ — вопрос с ограничивающим контекстом, а $x_i^{doc}$ — документ. Документы с высоким значением $r_k$ считаются более релевантными для ответа.

3. **Мелкозернистое сжатие**

Далее для каждого токена вычисляется контрастная перплексия:

$$s_i = \text{perplexity}(x_i | x_{<i}) - \text{perplexity}(x_i | x^{que}, x_{<i})$$

Это позволяет определить важность отдельных токенов в контексте заданного вопроса.

4. **Динамическое распределение бюджета**

Бюджет сжатия распределяется между документами согласно их важности:

$$\tau_i = \tau_k^{doc}, \forall x_i \in x_k^{doc}$$

$$\tau_k^{doc} = \max(\min((1 - \frac{2I(r_k)}{K'})\delta\tau + \tau_{doc}, 1), 0)$$

где более важные документы получают больший бюджет для сохранения информации.

5. **Переупорядочивание и восстановление**

Финальными этапами являются:

- Переупорядочивание документов для минимизации эффекта "потери в середине"
- Восстановление подпоследовательностей для сохранения целостности информации

Такой многоэтапный подход позволяет:

- Значительно уменьшить размер входного контекста
- Сохранить наиболее релевантную для вопроса информацию
- Оптимально расположить важную информацию в сжатом промпте

# 2 Количественные результаты

Таблица 1: Результаты на датасете NaturalQuestions при различных степенях сжатия

| Метод | Точность | Токены | Сжатие | Латентность | Ускорение |
|---|---|---|---|---|---|
| Сжатие 2x | | | | | |
| BM25 | 49.3% | 1,545 | 1.9x | 2.1 | 1.9x |
| SBERT | 67.9% | 1,549 | 1.9x | 2.2 | 1.9x |
| LongLLMLingua | **77.2%** | 1,429 | 2.1x | 2.9 | 1.4x |
| Сжатие 4x | | | | | |
| BM25 | 38.6% | 798 | 3.7x | 1.5 | 2.7x |
| SBERT | 61.1% | 808 | 3.6x | 1.6 | 2.5x |
| LongLLMLingua | **75.0%** | 748 | 3.9x | 2.1 | 2.0x |
| Исходный промпт | 75.7% | 2,946 | - | 4.1 | - |

Таблица 2: Результаты на различных задачах бенчмарка LongBench (ограничение 2000 токенов)

| Метод | Single | Multi | Summ | Few | Synth | Code | AVG |
|---|---|---|---|---|---|---|---|
| BM25 | 30.1 | 29.4 | 21.2 | 19.5 | 12.4 | 29.1 | 23.6 |
| SBERT | 33.8 | 35.9 | 25.9 | 23.5 | 18.0 | 17.8 | 25.8 |
| OpenAI | 34.3 | 36.3 | 24.7 | 32.4 | 26.3 | 24.8 | 29.8 |
| LongLLMLingua | **39.9** | **43.2** | **27.4** | **69.8** | **53.0** | **56.7** | **48.3** |
| Исходный промпт | 39.7 | 38.7 | 26.5 | 67.0 | 37.8 | 54.2 | 44.0 |
| Zero-shot | 15.6 | 31.3 | 15.6 | 40.7 | 1.6 | 36.2 | 23.5 |

## 2.1 Интерпретация результатов

В ходе экспериментов метод LongLLMLingua был протестирован на нескольких ключевых наборах данных:

- **NaturalQuestions**: При сжатии 4x достигнута точность 71-75% независимо от позиции релевантной информации, что превосходит исходные результаты
- **LongBench**: При ограничении в 2000 токенов средняя производительность 48.3% (исходная 44.0%) с шестикратным сжатием входных данных
- **ZeroSCROLLS**: Стабильные результаты даже при сильном сжатии, с улучшением некоторых метрик (например, BkSS: 47.2% против исходных 44.1%)
- **Экономические показатели**: Сокращение затрат от 52.6% до 94.0% в зависимости от датасета при сохранении или улучшении качества

# 3 Преимущества

a. **Технологическая эффективность:**

- Сокращение расходов на API-запросы до 94.0% в тестах LooGLE
- Ускорение общей латентности в 1.4x-2.6x при сжатии промптов около 10k токенов
- Стабильное 4-6 кратное уменьшение количества токенов при сохранении или улучшении качества работы
- Универсальность применения для различных языковых моделей (GPT-3.5, GPT-4, Claude и др.)

b. **Качественные улучшения:**

- Значительное улучшение точности в тестах NaturalQuestions (до 21.4%)
- Существенный прирост в синтетических задачах LongBench (15.5 пунктов)
- Стабильное улучшение на 3-5 пунктов в различных задачах LongBench
- Эффективное решение проблемы "потери информации в середине"(lost in the middle)

c. **Практическая применимость:**

- Простая интеграция через популярные фреймворки (LangChain, LlamaIndex)
- Полная совместимость с основными облачными API (OpenAI, Anthropic, Azure)
- Подробная документация с примерами для различных сценариев использования
- Открытый исходный код модели сжатия с инструкциями по обучению

d. **Универсальность применения:**

- Эффективность в multi-document QA (увеличение F1-score на 5.4 пункта в MuSiQue)
- Улучшение производительности в long-dependency QA (на 9.5 пунктов в LooGLE)
- Сохранение качества при суммаризации с существенным сокращением токенов
- Поддержка различных форматов входных данных и типов задач

e. **Экономическая выгода:**

- Существенное снижение затрат на API-запросы во всех тестах:
  - LooGLE: -94.0% ($93.6 → $5.6 на 1000 запросов)
  - LongBench: -90.5% ($31.5 → $3.0)
  - ZeroSCROLLS: -89.5% ($30.6 → $3.2)
  - Multi-document QA: -71.7% ($4.6 → $1.3)
  - MuSiQue: -52.6% ($3.8 → $1.8)
- Сокращение времени обработки запросов на 40-160%
- Снижение нагрузки на вычислительные ресурсы

f. **Техническая поддержка и развитие:**

- Понятный структурированный GitHub репозиторий с Quick Start и ссылками на статьи и Huggingface
- Подробные примеры использования в Jupyter notebooks
- Готовые интеграции с популярными инструментами и фреймворками
- Открытый код обучения модели сжатия с документацией

# 4 Недостатки

a. **Повторная компрессия для каждого запроса:** LongLLMLingua требует новой компрессии для каждого вопроса, даже если контекст тот же самый. Это увеличивает вычислительные затраты в 2 раза по сравнению с базовым LLMLingua, что может быть критично в промышленных системах.

b. **Проблемы со сложными зависимостями:** Несмотря на хорошие результаты на датасете MuSiQue, где модель улучшает базовый результат на 5.4 пункта, эффективность может снижаться в задачах с более тонкими семантическими связями из-за использования грубой вопросно-ориентированной компрессии.

c. **Ограничения на типы задач:** Метод особенно эффективен для задач вопросно-ответного типа, где скорость ответа критична. Например, на NaturalQuestions достигается ускорение в 2.6 раза при сжатии 6x. Однако для задач, требующих сохранения полного контекста (например, творческой генерации), такая агрессивная компрессия может быть неприменима.

d. **Качество сжатия зависит от малой модели:** При замене LLaMA-2-7B на GPT2-small производительность падает на 2.8-5.3 пункта в различных задачах, хотя всё ещё превосходит базовые методы. Это показывает важность выбора качественной малой модели для компрессии.

# 5 Влияние

a. **Существенное снижение затрат:** Метод показывает впечатляющее сокращение расходов на использование LLM: экономия составляет 71.7% для Multi-document QA, 90.5% для LongBench и до 94% для LooGLE. Это делает использование больших языковых моделей значительно доступнее, особенно для малого и среднего бизнеса.

b. **Ускорение обработки запросов:** При сжатии длинных промптов (около 10 тысяч токенов) достигается ускорение в 1.4-2.6 раза. В некоторых задачах, например на датасете NaturalQuestions, удаётся сократить количество токенов в 4 раза при одновременном улучшении точности на 21.4%.

c. **Повышение качества ответов:** На ключевых бенчмарках метод не только сохраняет, но и улучшает точность ответов. Например, в задачах Multi-document QA точность повышается на 3-5 пунктов, а в синтетических задачах LongBench - на 15.5 пунктов. Это открывает возможности для более надёжного использования LLM в критически важных областях, таких как анализ документов и научные исследования.

d. **Технологический прорыв в работе с контекстом:** LongLLMLingua предлагает новый подход к решению проблемы "lost in the middle потери информации в середине длинного контекста. Эффективность метода подтверждается на разных моделях, включая GPT-3.5-Turbo и LongChat-13B, что создаёт основу для дальнейшего развития технологий обработки длинных текстов.

# 6 Бизнес-кейсы

a. **Поисковые системы и QA-сервисы:** На примере NaturalQuestions показано повышение точности на 21.4% при использовании в 4 раза меньшего количества токенов. Это значительно снижает затраты на API-запросы и ускоряет работу системы в 1.4-2.6 раза при сжатии контекста в 2-6 раз.

b. **Обработка длинных документов:** При работе с документами объёмом около 10 тысяч токенов технология позволяет сократить затраты на 90.5% (данные бенчмарка LongBench) при сохранении или даже улучшении качества результатов. В некоторых случаях, как показано на бенчмарке LooGLE, экономия достигает 94%.

c. **Многоступенчатый анализ:** В задачах, требующих анализа нескольких документов (multi-hop QA), технология демонстрирует улучшение производительности на 5.4

пункта по метрике F1 при двукратном сжатии входных данных. Это особенно важно для систем, работающих с множеством взаимосвязанных документов.

d. **Работа с кодом:** В задачах автодополнения кода технология сохраняет высокую точность даже при значительном сжатии контекста, что подтверждается результатами тестирования на Code Completion части бенчмарка LongBench.

e. **Генерация текста:** В задачах суммаризации и создания отчётов система демонстрирует стабильные результаты даже при сжатии входных данных в 3-6 раз, что подтверждается тестами на бенчмарках ZeroSCROLLS и Gov Report.

# 7 Будущие разработки

a. **Оптимизация вычислительной эффективности:** Текущая реализация требует двойных вычислений по сравнению с базовой LLMLingua из-за question-aware подхода. Приоритетная задача - разработка task-aware версии, позволяющей кэшировать и переиспользовать сжатый контекст для схожих задач, что значительно снизит накладные расходы.

b. **Улучшение работы со сложными зависимостями:** Тесты на датасете MuSiQue показали, что при многоступенчатых запросах система может терять часть важных связей между документами. Необходимо усовершенствовать механизмы определения и сохранения сложных смысловых зависимостей, особенно в задачах, где F1-мера сейчас достигает 51.2%.

c. **Масштабирование на сверхдлинные контексты:** При работе с контекстами более 24K токенов (как в бенчмарке LooGLE) текущая реализация демонстрирует снижение эффективности. Требуется адаптация алгоритмов для работы со сверхдлинными последовательностями при сохранении высокой степени сжатия (сейчас до 94% на LooGLE).

d. **Динамическая настройка сжатия:** Развитие механизма динамических коэффициентов сжатия, который сейчас использует линейный планировщик ... с гиперпараметром $\delta\tau = 0.3$. Необходимо исследовать более сложные стратегии распределения бюджета токенов для различных частей документа.

e. **Улучшение восстановления последовательностей:** Существующий алгоритм восстановления подпоследовательностей может быть улучшен для более точного воспроизведения критически важных частей текста, особенно в задачах, где важна дословная точность цитирования (сейчас используется prefix trees и sequence automata).