

# Ограничения памяти в LLM-системах: углублённый анализ архитектуры, вычислительной сложности и инженерных компромиссов

19 января 2026 г.

## Аннотация

Большие языковые модели (Large Language Models, LLM) стали основой современных систем обработки естественного языка. Несмотря на впечатляющие результаты в генерации текста, рассуждениях и анализе данных, практическое применение LLM ограничено фундаментальными и инженерными аспектами памяти. В данной работе представлен углублённый анализ ограничений длины контекста, вычислительной сложности механизма self-attention, требований к оперативной памяти при инференсе и обучении, а также отсутствия истинной долгосрочной памяти у LLM. Рассматриваются современные подходы к смягчению этих ограничений, включая Retrieval-Augmented Generation (RAG), длинноконтекстные архитектуры и альтернативные механизмы внимания.

## Содержание

<b>1 Введение</b>	<b>3</b>
<b>2 Архитектура трансформеров и память</b>	<b>3</b>
2.1 Общая структура трансформера . . . . .	3
2.2 Self-attention и память . . . . .	3
<b>3 Типы памяти в LLM-системах</b>	<b>4</b>
3.1 Параметрическая память . . . . .	4
3.2 Контекстная память . . . . .	4
3.3 Внешняя память . . . . .	4
<b>4 Ограничение длины контекста</b>	<b>4</b>
4.1 Теоретические пределы . . . . .	4
4.2 Практические значения . . . . .	4
<b>5 Память при инференсе и обучении</b>	<b>5</b>
5.1 Инференс . . . . .	5
5.2 Обучение . . . . .	5
<b>6 Отсутствие долгосрочной памяти</b>	<b>5</b>

<b>7 Инженерные подходы к смягчению ограничений</b>	<b>5</b>
7.1 Сжатие контекста . . . . .	5
7.2 Окноное и разреженное внимание . . . . .	5
7.3 Retrieval-Augmented Generation . . . . .	6
7.4 Длинноконтекстные архитектуры . . . . .	6
<b>8 Последствия для практических систем</b>	<b>6</b>
<b>9 Будущие направления</b>	<b>6</b>
<b>10 Заключение</b>	<b>6</b>

# 1 Введение

Большие языковые модели, такие как GPT, LLaMA, Claude и их производные, основаны на архитектуре трансформеров, предложенной в работе Vaswani и др. [1]. Ключевым компонентом этой архитектуры является механизм самовнимания (self-attention), позволяющий моделировать зависимости между всеми токенами входной последовательности.

Несмотря на высокую выразительную способность трансформеров, их практическое использование сопровождается рядом жёстких ограничений, связанных с памятью:

- ограниченная длина входного контекста;
- квадратичная вычислительная и памятная сложность self-attention;
- высокие требования к VRAM/RAM при инференсе и обучении;
- отсутствие механизма долгосрочной памяти.

Эти ограничения существенно влияют на применимость LLM для работы с длинными документами, кодовыми базами и диалоговыми агентами [2].

## 2 Архитектура трансформеров и память

### 2.1 Общая структура трансформера

Трансформер состоит из стека идентичных слоёв, каждый из которых включает:

- механизм multi-head self-attention;
- позиционно-независимые полносвязные сети;
- остаточные соединения и нормализацию.

Пусть входная последовательность имеет длину  $L$ , а размерность скрытого пространства —  $d$ . Тогда для каждого слоя формируются матрицы запросов  $Q$ , ключей  $K$  и значений  $V$  размером  $L \times d$ .

### 2.2 Self-attention и память

Механизм self-attention вычисляется как:

$$\text{Attention}(Q, K, V) = \text{softmax} \left( \frac{QK^T}{\sqrt{d}} \right) V \quad (1)$$

Матрица  $QK^T$  имеет размер  $L \times L$ , что приводит к квадратичному росту памяти и времени вычислений:

$$O(L^2) \quad (2)$$

Это фундаментальное свойство архитектуры трансформеров и основная причина ограничения длины контекста [1, 3].

## 3 Типы памяти в LLM-системах

### 3.1 Параметрическая память

Параметрическая память — это знания, закодированные в весах модели в процессе обучения. Современные LLM содержат от миллиардов до сотен миллиардов параметров.

Преимущества:

- быстрый доступ к знаниям;
- отсутствие необходимости во внешних источниках.

Недостатки:

- устаревание знаний;
- невозможность точечного обновления фактов;
- высокая стоимость переобучения.

### 3.2 Контекстная память

Контекстная память представлена входной последовательностью токенов. Максимальная длина контекста  $L$  жёстко ограничена архитектурой и доступной памятью.

Если длина входа превышает  $L$ , часть информации должна быть отброшена или сжата, что может приводить к потере важных зависимостей [2].

### 3.3 Внешняя память

Внешняя память реализуется через базы знаний, поисковые индексы и векторные базы данных. Она активно используется в Retrieval-Augmented Generation (RAG) [4].

## 4 Ограничение длины контекста

### 4.1 Теоретические пределы

Квадратичная сложность self-attention приводит к следующим ограничениям:

- рост потребления памяти как  $O(L^2)$ ;
- рост времени инференса как  $O(L^2)$ ;
- ограничение масштабируемости на длинных последовательностях.

### 4.2 Практические значения

Типичные значения максимального контекста:

- 2k–8k токенов — ранние GPT-подобные модели;
- 16k–32k токенов — современные коммерческие модели;
- 100k+ токенов — специализированные длинноконтекстные модели [5].

## 5 Память при инференсе и обучении

### 5.1 Инференс

Память при инференсе расходуется на:

- хранение весов модели;
- активации слоёв;
- ключи и значения (KV-cache).

KV-cache растёт линейно с длиной контекста и числом слоёв [6].

### 5.2 Обучение

Во время обучения дополнительно хранятся:

- градиенты;
- состояния оптимизатора (например, Adam).

Это увеличивает потребление памяти в 2–4 раза по сравнению с инференсом [7].

## 6 Отсутствие долгосрочной памяти

LLM не имеют истинной долгосрочной памяти:

- контекст теряется после завершения запроса;
- веса модели не обновляются в ходе диалога;
- «память» реализуется на уровне приложения.

Это требует хранения истории диалога и повторной передачи её в prompt [2].

## 7 Инженерные подходы к смягчению ограничений

### 7.1 Сжатие контекста

Используются:

- суммаризация;
- извлечение ключевых фактов;
- удаление нерелевантных фрагментов.

### 7.2 Оконное и разреженное внимание

Модели с локальным вниманием ограничивают область self-attention, снижая сложность до  $O(L \cdot w)$ , где  $w$  — размер окна [3].

### **7.3 Retrieval-Augmented Generation**

RAG объединяет LLM с внешней памятью:

1. запрос преобразуется в embedding;
2. релевантные документы извлекаются из векторной БД;
3. они добавляются в prompt модели.

Подход описан в работе Lewis и др. [4].

### **7.4 Длинноконтекстные архитектуры**

Используются:

- sparse attention;
- low-rank аппроксимации;
- рекуррентные механизмы [5].

## **8 Последствия для практических систем**

- невозможность прямой загрузки больших документов;
- необходимость чанкования и индексации;
- рост латентности;
- увеличение стоимости инференса.

## **9 Будущие направления**

- объединение внешней и внутренней памяти;
- модели с постоянной памятью;
- аппаратные ускорители для attention;
- альтернативные архитектуры (Mamba, RWKV) [8].

## **10 Заключение**

Ограничения памяти в LLM-системах являются следствием фундаментальных свойств архитектуры трансформеров и текущих инженерных компромиссов. Несмотря на развитие длинноконтекстных моделей и внешней памяти, данные ограничения остаются ключевым фактором при проектировании масштабируемых LLM-приложений.

## Список литературы

- [1] Vaswani A. et al. Attention Is All You Need. NeurIPS, 2017.
- [2] Liu J. et al. Lost in the Middle: How Language Models Use Long Contexts. ACL, 2023.
- [3] Beltagy I. et al. Longformer: The Long-Document Transformer. arXiv:2004.05150, 2020.
- [4] Lewis P. et al. Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks. NeurIPS, 2020.
- [5] Dao T. et al. FlashAttention: Fast and Memory-Efficient Exact Attention. NeurIPS, 2022.
- [6] Kwon W. et al. Efficient Memory Management for Large Language Model Serving. MLSys, 2023.
- [7] Rajbhandari S. et al. ZeRO: Memory Optimization Toward Training Trillion Parameter Models. SC, 2020.
- [8] Gu A., Dao T. Mamba: Linear-Time Sequence Modeling with Selective State Spaces. arXiv:2312.00752, 2023.