ИНФОРМАЦИОННОЕ ПОЛЕ ОБЪЕКТОВ В БОЛЬШИХ ЯЗЫКОВЫХ МОДЕЛЯХ: ОТ АРХИТЕКТУРЫ К ФОРМИРОВАНИЮ ПРЕДСТАВЛЕНИЙ

Н. А. Бондарева[а], **А. Е. Бондарев**[б]
Институт прикладной математики им. М.В. Келдыша РАН, г. Москва, Российская Федерация
[а] *ORCID: http://orcid.org/0000-0002-7586-903X,* ✍ *nicibond9991@gmail.com*
[б] *ORCID: http://orcid.org/0000-0003-3681-5212, bond@keldysh.ru*

Аннотация: настоящая работа представляет обзор архитектурных принципов и алгоритмов современных генеративных нейронных сетей и больших языковых моделей. Рассматриваются фундаментальные механизмы обработки текста: векторные представления, токенизация, многослойная контекстуализация, авторегрессивная генерация и управление контекстной памятью. Детально анализируется архитектура Transformer с акцентом на механизм самовнимания как ключевую инновацию, обеспечивающую параллельную обработку последовательностей и эффективное улавливание дальних зависимостей. Описываются варианты архитектуры (encoder-only, decoder-only, encoder-decoder), позиционное кодирование и вычислительная сложность компонентов.

Рассматриваются процессы предобучения на больших текстовых корпусах, адаптации через fine-tuning и instruction tuning, выравнивания с человеческими предпочтениями посредством RLHF, а также методы эффективной адаптации (LoRA, квантизация). Вводится концепция информационного поля объектов как интегральной характеристики качества и плотности информации в цифровом пространстве, а также совокупности всех информационных единиц, содержащих упоминания исследуемого объекта в цифровом пространстве. Демонстрируется связь между структурой информационного поля в обучающих данных и формированием представлений в пространстве эмбеддингов модели. Обсуждаются проблемы качества информационного окружения, включая деградацию точности при включении низкокачественных источников, и подходы к их решению.

Ключевые слова: большие языковые модели, Transformer, механизм самовнимания, информационное поле, обучение с подкреплением от человека, векторные представления, генерация текста.

Для цитирования: Бондарева Н. А., Бондарев А. Е. Информационное поле объектов в больших языковых моделях: от архитектуры к формированию представлений. *Успехи кибернетики*. 2026;7(1):12–23.

Поступила в редакцию: 15.12.2025. *В окончательном варианте: 22.12.2025.*

REPRESENTATION OF OBJECTS IN LARGE LANGUAGE MODELS: FROM ARCHITECTURE TO EMERGENCE

N. A. Bondareva[a], **A. E. Bondarev**[b]
Keldysh Institute of Applied Mathematics, Russian Academy of Sciences, Moscow, Russian Federation
[a] *ORCID: http://orcid.org/0000-0002-7586-903X,* ✍ *nicibond9991@gmail.com*
[b] *ORCID: http://orcid.org/0000-0003-3681-5212, bond@keldysh.ru*

Abstract: we reviewed the architectural principles and algorithms of modern generative neural networks and large language models. We examined the core mechanisms of text processing, including tokenization, embeddings, deep contextualization, autoregressive generation, and context management. We analyzed the transformer architecture in detail and emphasized self-attention as the key innovation that enables parallel sequence processing and captures long-range dependencies efficiently. We described common architecture variants (encoder-only, decoder-only, and encoder–decoder), positional encoding methods, and the computational complexity of major components.

We studied pretraining on large text corpora and subsequent adaptation through fine-tuning and instruction tuning. We examined alignment with human preferences using reinforcement learning from human feedback (RLHF) and reviewed efficient adaptation methods, including low-rank adaptation (LoRA) and quantization. We introduced the concept of an object information context, which we define as the overall amount and distribution of information about a given object in digital data, including all references to that object in the training corpus. We showed that the structure of this context in the training data influences the formation of object representations in the model's embedding space. We also examined the quality

of the data environment and showed that low-quality sources degrade model accuracy, and we discussed approaches to mitigate these effects.

Keywords: large language models, transformer, self-attention, representation of objects, reinforcement learning with human feedback, vector representations, text generation.

Cite this article: Bondareva N. A., Bondarev A. E. Representation of Objects in Large Language Models: From Architecture to Emergence. *Russian Journal of Cybernetics*. 2026;7(1):12–23.

Original article submitted: 15.12.2025. *Revision submitted: 22.12.2025*.

Введение

2022–2023 годы ознаменовали качественный скачок в развитии искусственного интеллекта, когда генеративные модели, способные создавать различные виды медиа и текстовой информации, включая программный код, прошли этап исследований и поступили в массовое использование. Запущенный в ноябре 2022 года ChatGPT, который в то время основывался на моделях GPT-3.5 и GPT-4, продемонстрировал миру возможности больших языковых моделей LLM (Large Language Models,), которые способны поддерживать связный диалог, отвечать на вопросы, анализировать тексты и выполнять множество других интеллектуальных задач. Параллельно с этим выпущенные системы генерации изображений DALL-E, Midjourney и Stable Diffusion показали миру впечатляющие результаты в создании визуального контента по текстовым описаниям.

Однако за внешней «магией» работы этих систем стоят конкретные архитектурные решения и алгоритмы, которые развивались на протяжении последних лет. В основе практически всех современных LLM лежит архитектура Transformer, предложенная в 2017 году в работе “Attention is All You Need” [1]. Эта архитектура совершила прорыв в обработке естественного языка, заменив рекуррентные подходы механизмом самовнимания (self-attention) и открыв путь к эффективному масштабированию моделей до сотен миллиардов параметров. Траектория развития от GPT-1 с 117 миллионами параметров [2, 3] до GPT-4 с, предположительно, более чем триллионом параметров демонстрирует не только количественный рост, но и качественные изменения: так называемые эмерджентные способности, которые возникают только при достижении определенного масштаба модели [4, 5]. Исследования показывают, что производительность моделей предсказуемо улучшается с увеличением размера модели, объема обучающих данных и вычислительных ресурсов, следуя scaling laws [11, 12].

Несмотря на широкое распространение и использование LLM, принципы их работы остаются малопонятными для большинства пользователей и даже многих специалистов смежных областей. Цель данной обзорной статьи — систематически описать архитектурные принципы и алгоритмы, лежащие в основе современных генеративных нейронных сетей, с акцентом на доступность изложения для широкого круга читателей.

Большая языковая модель (LLM) — это тип нейронной сети, обученной на огромных объемах текстовых данных с целью понимания и генерации человеческого языка. Термин «большая» отражает не только размер модели (от сотен миллионов до сотен миллиардов параметров), но и масштаб обучающих данных (сотни миллиардов, триллионы токенов текста) и вычислительных ресурсов, затраченных на обучение.

В основе работы современных LLM, таких как GPT [3–6], LLaMA [9], Mistral [10] и других, лежит авторегрессивный принцип: модель обучается предсказывать следующий токен (подслово или слово) в последовательности на основе всех предыдущих токенов. Эта, на первый взгляд, простая задача в сочетании с масштабом данных и модели приводит к возникновению сложных взаимозависимостей: модель «учится» грамматике, фактам о мире, элементам логического вывода и даже некоторым аспектам здравого смысла, извлекая статистические паттерны из обучающего свода материалов.

Ключевое свойство современных LLM — генеративность: в отличие от более ранних моделей обработки языка, которые специализировались на задачах классификации или извлечения информации (например, BERT [7]), генеративные модели способны создавать новый контент: писать тексты, отвечать на вопросы, генерировать код, переводить и выполнять множество других задач без специального обучения на каждую из них. Это свойство называется zero-shot и few-shot learning [5, 14]: модель может выполнять новые задачи, основываясь только на инструкции в текстовом запросе, или так называемом промпте (prompt), возможно, с несколькими примерами.

Важно понимать, что LLM работают на уровне статистических паттернов, а не истинного «понимания» в человеческом смысле [23]. Модель не обладает сознанием, убеждениями или намерениями;

она не имеет связи с реальным физическим миром. Тем не менее мощность статистической аппроксимации оказывается достаточной для решения широкого спектра практических задач, что и объясняет быстрое внедрение этих технологий в индустрию, образование, науку и повседневную жизнь.

Архитектурная основа практически всех современных больших языковых моделей — Transformer, а точнее его decoder-only вариант. Эта архитектура, в отличие от предшествовавших рекуррентных нейронных сетей (RNN, LSTM), позволяет эффективно параллелизовать обработку последовательностей и масштабироваться до очень больших размеров. Парадигма предобучения и дообучения (pre-training and fine-tuning) [3, 7, 8] стала стандартом: сначала модель обучается на общем текстовом объеме данных в режиме самообучения (unsupervised learning), а затем адаптируется для конкретных задач или выравнивается с человеческими предпочтениями через методы вроде RLHF [13] или Constitutional AI [16].

Как нейросети «понимают» и генерируют текст

Прежде чем погружаться в архитектурные детали Transformer и специфику больших языковых моделей, необходимо разобраться с фундаментальными принципами работы нейронных сетей применительно к текстовым данным. В этом разделе последовательно рассматривается, как слова преобразуются в числовые представления, понятные нейросети; как многослойная обработка извлекает все более абстрактные паттерны; как модель генерирует текст токен за токеном и как работает механизм контекстной памяти.

Нейронные сети, как и любые компьютерные системы, оперируют числами — векторами и матрицами чисел с плавающей точкой. Человеческий же язык состоит из дискретных символических единиц: букв, слов, предложений. Первая фундаментальная задача при обработке текста нейросетями — преобразовать дискретные символы в непрерывные числовые представления, с которыми может работать математический аппарат нейросетей.

Ранние подходы использовали так называемое one-hot кодирование: каждое слово представлялось вектором длиной, равной размеру словаря (например, 50 000 слов), где все элементы равны нулю, кроме одного, соответствующего данному слову. Однако такое представление имеет такие недостатки, как векторы огромной размерности, отсутствие информации о семантическом сходстве слов (слова «кот» и «кошка» так же далеки друг от друга, как «кот» и «компьютер»), невозможность обработки слов, не встречавшихся при обучении.

Решением этой проблемы стали эмбеддинги (embeddings): плотные векторные представления небольшой фиксированной размерности (обычно от 256 до 12 288 измерений в современных LLM). Каждое слово или токен преобразуется в вектор действительных чисел, где близкие по смыслу слова располагаются близко в векторном пространстве. Интуитивная аналогия: представьте многомерную карту смыслов, где каждое слово — точка в пространстве. Слова со схожими значениями или употребляемые в схожих контекстах оказываются рядом. «Король», «королева», «монарх» находятся в одной области пространства; «программирование», «код», «алгоритм» — в другой. Более того, в этом пространстве возникают семантические направления: вектор от «король» к «королева» приблизительно параллелен вектору от «мужчина» к «женщина», что отражает гендерное различие. При этом эмбеддинги обучаемы. В процессе обучения нейросети веса, преобразующие токены в векторы, настраиваются так, чтобы захватить семантические и синтаксические отношения, полезные для решаемой задачи. В современных LLM эмбеддинги учатся совместно со всей архитектурой в процессе предобучения на огромных текстовых выборках данных [3, 5].

Прежде чем преобразовать текст в эмбеддинги, его необходимо разбить на токены — базовые единицы обработки. Подход простого разбиения по словам на практике оказывается достаточно наивным, так как здесь встает ряд проблем:

- огромный словарь: языки содержат сотни тысяч уникальных слов, включая словоформы, имена собственные, термины;
- многие слова встречаются крайне редко, что затрудняет обучение их представлений;
- слова вне словаря (OOV), когда новые слова, опечатки, иностранные заимствования не могут быть обработаны;
- разные языки требуют разных словарей, что исключает многоязычность.

Современные LLM используют разбиение на подсловные единицы (subword tokenization). Наиболее популярные алгоритмы: BPE (Byte Pair Encoding) итеративно объединяет наиболее частые пары

символов или токенов; WordPiece (похож на BPE, используется в BERT); SentencePiece (работает напрямую с последовательностями символов, удобен для многоязычных моделей).

```
Пример токенизации (приблизительный):
- "Нейросети" → ["Ней", "ро", "сети"] (редкое слово разбивается на части);
- "ChatGPT" → ["Chat", "GPT"];
- "transformer" → ["transform", "er"];
- "кот" → ["кот"] (частое слово --- отдельный токен)
```

После преобразования текста в эмбеддинги последовательность векторов проходит через множество слоев нейронной сети. В современных LLM количество слоев варьируется от десятков до более сотни.

Ключевая идея глубокого обучения — иерархическое извлечение признаков [24]. Каждый слой создает все более абстрактные описания входных данных, подобно тому, как человек воспринимает текст: от линий букв к словам, затем к грамматической структуре и смыслу высокого уровня.

Исследования показывают функциональную специализацию слоев в LLM:

- нижние слои (1–20%): низкоуровневые паттерны — части речи, синтаксические отношения, частотные комбинации токенов;
- средние слои (20–70%): семантические отношения и репрезентации понятий. Здесь хранятся фактические знания модели (например, что «Париж» — столица Франции);
- верхние слои (70–100%): абстрактные концепции, рассуждения, адаптация к конкретному контексту запроса.

Каждый слой выполняет несколько операций: механизм внимания (контекстуализация токенов), feed-forward преобразование (обогащение представлений) и остаточные связи с нормализацией (стабильность обучения).

Контекстуализация — ключевое свойство трансформеров. В отличие от статических эмбеддингов, представление каждого токена зависит от контекста. Слово «замок» получит разные представления в контекстах «открыть замок» и «средневековый замок». После всех слоев финальное преобразование в генеративных моделях (GPT) превращает представление в вероятностное распределение над словарем токенов.

Последний слой модели — это линейная проекция представления токена на размерность словаря (например, 50,000 чисел) с последующим применением функции softmax, преобразующей логиты в вероятности. Каждое число — это оценка вероятности соответствующего токена. Модель обучается максимизировать вероятность правильного следующего токена. После обучения на триллионах токенов она «выучила» статистические паттерны языка.

Наивный подход (greedy decoding) заключается в том, чтобы всегда выбирать токен с максимальной вероятностью. Его применение приводит к детерминированным, повторяющимся и «скучным» текстам.

Современные системы используют стохастические стратегии с контролируемой случайностью:

1. Temperature sampling.

Параметр температуры τ модулирует распределение вероятностей:

- низкая температура (τ = 0.1–0.3): модель выбирает наиболее вероятные токены. Результат — консервативный, предсказуемый, фактологически надежный текст;
- температура = 1.0: исходное распределение модели;
- высокая температура (τ = 0.8–1.5): увеличивает шансы менее вероятных токенов. Результат — разнообразный, творческий текст, но выше риск бессмыслицы.

2. Top-k sampling.

Рассматриваем только k наиболее вероятных токенов (например, k=40), остальные отбрасываем.

3. Top-p (nucleus) sampling (используется в современных моделях).

Адаптивный подход: включаем минимальное число токенов, чья суммарная вероятность достигает порога p (обычно 0.9-0.95).

Пример: если топ-3 токена имеют вероятности 0.5, 0.3, 0.15 и p=0.9, выбираем из этих трех (сумма $0.95 \geq 0.9$). При равномерном распределении может включаться 10–20 токенов.

Для фактологических задач используют низкую температуру и nucleus sampling с небольшим p. Для творческих задач — высокую температуру и больший p.

Важно отметить, что модель не «понимает» в человеческом смысле. Она выполняет статистическое сопоставление паттернов [23]. Однако масштаб делает качественную разницу. Обучение на сотнях миллиардов токенов позволяет запомнить огромное количество фактов, выучить грамматику множества языков, улавливать паттерны логических рассуждений, математики, программирования, имитировать различные стили письма. Эта «компрессия» интернета в параметры сети оказывается мощной для решения широкого спектра задач без явного программирования логики.

В отличие от человеческой памяти, LLM имеют строго ограниченную «рабочую память» — контекстное окно (context window). Контекстное окно определяет максимальное количество токенов, которые модель может обрабатывать одновременно. Это включает и входной промпт, и сгенерированный ответ. Эволюция размеров контекста составляла примерно: GPT-2 (2019) содержал приблизительно 1 024 токена (∼750 слов), а GPT-4 Turbo — уже 128 000 токенов.

2K токенов	≈ 1-2 страницы текста
8K токенов	≈ небольшая статья или глава
32K токенов	≈ короткая книга
100K токенов	≈ средний роман

Модель «видит» только содержимое текущего окна. Информация за пределами окна для нее не существует в рамках текущего запроса.

Архитектура Transformer: сердце современных LLM

Ключевая инновация архитектуры Transformer заключается в полном отказе от рекуррентной обработки в пользу механизма самовнимания (self-attention), что позволило обрабатывать все элементы последовательности параллельно и эффективно улавливать зависимости.

До появления Transformer доминирующими архитектурами для работы с последовательностями были рекуррентные нейронные сети (RNN) и их модификации LSTM. Эти модели обрабатывали текст последовательно, что создавало два ограничения: невозможность эффективной параллелизации и сложность улавливания дальних зависимостей. Механизм внимания, изначально предложенный в работе [2] как дополнение к RNN, в архитектуре Transformer стал основным механизмом взаимодействия между элементами последовательности.

Механизм самовнимания (Self-Attention)

Задача контекстуализации. Рассмотрим предложение: «Банк реки был крутым, но банк отказал в кредите». Слово «банк» встречается дважды в различных значениях. Для корректной интерпретации каждого вхождения необходимо проанализировать окружающий контекст: первое «банк» следует понимать через связь со словами «реки» и «крутым», второе — через «кредите». Механизм self-attention решает эту задачу, позволяя каждому токену взаимодействовать со всеми остальными токенами последовательности для определения их релевантности.

Концептуально механизм можно представить как систему выборочного внимания: каждый элемент последовательности одновременно анализирует все остальные элементы, определяет степень их релевантности и формирует свое обновленное представление на основе взвешенной комбинации их информации. Этот процесс аналогичен поиску в базе данных, где для каждого запроса (query) определяется степень соответствия всех доступных ключей (keys) и извлекаются соответствующие значения (values).

Для последовательности векторов-эмбеддингов $[x_1, x_2, \ldots x_n]$, где каждый x_i имеет размерность d, механизм self-attention выполняет следующие операции.

Проекции в пространства Q, K, V. Для каждого токена создаются три представления путем умножения на обучаемые матрицы весов:

$$\mathrm{Q(Query)} = x_i \times W^{\phi},$$
$$\mathrm{K(Key)} = x_i \times W_K,$$
$$\mathrm{V(Value)} = x_i \times W_v.$$

Три различных проекции необходимы, поскольку токен выполняет разные функциональные роли: как query токен формулирует запрос на контекстуальную информацию, как key предлагает свою информацию для других токенов, как value предоставляет содержательное представление для агрегации.

Вычисление весов внимания. Для токена i степень его связи с токеном j определяется через скалярное произведение их Query и Key представлений:

$$score\,(i,j) = Q_i \cdot K_j/\sqrt{d_k},$$

где $\sqrt{d_k}$ — масштабирующий коэффициент, предотвращающий чрезмерный рост значений при больших размерностях. Скалярное произведение измеряет направленную похожесть векторов: высокие значения указывают на релевантность токена j для понимания токена i.

Полученные scores нормализуются функцией softmax для получения распределения вероятностей:

$$\alpha_{ij} = softmax(score(i,j)),$$

где α_{ij} представляет вес внимания, который токен i должен уделить токену j.

Взвешенная агрегация. Финальное представление формируется как взвешенная сумма Value векторов:

$$output_i = \sum_j (\alpha_i \times V_j).$$

Таким образом, каждый токен получает контекстуализированное представление, объединяющее информацию от всех токенов в последовательности пропорционально их релевантности. Пример работы механизма: рассмотрим предложение «The animal didn't cross the street because it was too tired». При обработке местоимения «it» механизм attention вычисляет высокие веса для токенов «animal» (потенциальный референт) и «tired» (характеризующее прилагательное), в то время как служебные слова получают низкие веса. В результате представление «it» обогащается семантикой этих ключевых слов, что позволяет модели корректно интерпретировать смысл.

Визуализация весов внимания в виде матрицы [длина × длина] демонстрирует интерпретируемые лингвистические паттерны: существительные фокусируются на своих модификаторах, глаголы — на актантах, местоимения — на референтах.

Multi-Head Attention

Один механизм self-attention способен уловить определенный аспект отношений между токенами. Однако естественный язык характеризуется множественностью одновременно существующих зависимостей: синтаксических, семантических, референциальных. Механизм multi-head attention реализует параллельное применение нескольких независимых механизмов attention, каждый со своими матрицами W^{ϕ}, W_K, W_v.

При общей размерности модели d и количестве голов h каждая голова работает с размерностью $d_k = d/h$. Например, при d = 768 и h = 12 каждая из 12 голов оперирует векторами размерности 64. После параллельного вычисления attention всеми головами их выходы конкатенируются и проходят через линейное преобразование:

$$MultiHead\,(Q,K,V) = Concat\,(head_1, \ldots head_h) \times W_o,$$

где $head_i = Attention(Q \times W^{\phi(i)}, K \times W_{K^{(i)}}, V \times W_{v^{(i)}})$.

Количество голов варьируется в зависимости от размера модели: от 8–12 в компактных моделях до 96–128 в крупных системах, таких как GPT-3. Эмпирические исследования показывают, что различные головы специализируются на разных аспектах языковой структуры: локальных зависимостях, синтаксических связях, позиционных паттернах, хотя специализация не всегда имеет четкую интерпретацию.

Механизм multi-head attention составляет один из компонентов полного Transformer блока. Стандартная архитектура блока включает следующие элементы:

- Multi-Head Self-Attention обеспечивает взаимодействие между всеми токенами последовательности через описанный выше механизм;

• Residual Connection и Layer Normalization. После каждого подслоя применяется остаточная связь и нормализация:

$$output = LayerNorm(input + Sublayer(input)).$$

Остаточные связи особенно важны для обучения глубоких сетей. Концептуально каждый слой не заменяет представление, а добавляет к нему инкрементальное уточнение. Нормализация уровня стабилизирует распределение активаций, что ускоряет и стабилизирует процесс обучения.

Современные LLM представляют собой стек из множества Transformer блоков. Количество слоев значительно варьируется: GPT-2 содержит от 12 до 36 блоков, BERT — 12-24 блока [7], GPT-3 — 96 блоков [5], GPT-4, предположительно, более 120 блоков. Каждый последующий слой строит более абстрактные представления, формируя иерархию признаков: нижние слои обрабатывают синтаксические паттерны, средние — семантические отношения и фактические знания, верхние — абстрактные концепции и задачно-специфичное рассуждение.

Механизм self-attention инвариантен к перестановкам: изменение порядка токенов в последовательности не влияет на структуру вычислений, хотя результаты будут применены к переставленным элементам. Поскольку порядок слов является весьма важным для понимания естественного языка («Кот ест мышь» семантически отличается от «Мышь ест кот»), необходим механизм внесения позиционной информации.

Позиционное кодирование решает эту задачу путем добавления к эмбеддингам токенов векторов, кодирующих их позиции:

$$input = token\_embedding + positional\_encoding.$$

Существует несколько подходов к реализации позиционных кодирований.

Синусоидальные кодирования, предложенные в оригинальной работе [1], используют функции sin и cos различных частот:

$$PE(pos, 2i) = \sin(pos/10000 \wedge (2i/d)),$$

$$PE(pos, 2i + 1) = \cos(pos/10000 \wedge (2i/d)).$$

Эти детерминированные (необучаемые) кодирования обладают свойством уникальности для каждой позиции и теоретической способностью к экстраполяции на последовательности, длиннее обучающих.

Обучаемые позиционные эмбеддинги, используемые в GPT и BERT, представляют собой таблицу векторов — по одному на каждую возможную позицию, оптимизируемую в процессе обучения. Этот подход обеспечивает большую гибкость и эмпирически часто демонстрирует лучшие результаты, однако ограничен фиксированной максимальной длиной последовательности. Относительные позиционные кодирования фокусируются на относительных расстояниях между токенами вместо абсолютных позиций, что обеспечивает инвариантность к сдвигам.

Базовая архитектура Transformer включала компоненты encoder и decoder. Современные модели используют различные конфигурации этих компонентов в зависимости от целевых задач.

Encoder-only архитектура (BERT) использует стек Transformer блоков с двусторонним (bidirectional) self-attention, где каждый токен имеет доступ ко всем токенам последовательности. Обучение осуществляется через Masked Language Modeling: случайно маскируется некоторая доля токенов (обычно 15%), которые модель должна предсказать на основе окружающего контекста. Такая архитектура оптимальна для задач понимания текста: классификации, извлечения именованных сущностей, вопросно-ответных систем. Двусторонний контекст не позволяет использовать эти модели для генерации, поскольку при автогрессивной генерации будущие токены еще не существуют.

Decoder-only архитектура (GPT, LLaMA, Mistral) представляет доминирующий подход в современных LLM. Используется каузальный (causal) self-attention с маскированием: при обработке токена на позиции i модель имеет доступ только к токенам на позициях 1...i, но не к последующим. Обучение проводится через предсказание следующего токена (next token prediction): для каждой позиции модель предсказывает следующий токен, что обеспечивает плотный обучающий сигнал из каждого примера. Decoder-only архитектура стала стандартом благодаря нескольким преимуществам: универсальности

(единая архитектура для различных задач через промпты), эффективности обучения (каждая позиция дает обучающий сигнал), простоте масштабирования и естественной поддержке генерации.

Encoder-Decoder архитектура [8] комбинирует encoder с bidirectional attention для обработки входа и decoder с causal attention для генерации выхода. Эта конфигурация оптимальна для задач преобразования: машинного перевода, суммаризации, где явное разделение на вход и выход является естественным.

Обучение и адаптация больших языковых моделей

Современные LLM проходят через несколько этапов обучения: предобучение на больших текстовых выборках для формирования базовых языковых способностей, последующую адаптацию для конкретных задач и выравнивание с человеческими предпочтениями.

Предобучение формирует фундамент возможностей LLM. Модели обучаются на огромных текстовых данных (корпусах), включающих веб-страницы, книги, научные статьи, код и другие источники. Объем обучающих данных достигает сотен миллиардов токенов для GPT-3 и, предположительно, триллионов для GPT-4.

Задача обучения для decoder-only моделей — предсказание следующего токена (next token prediction). Для каждой позиции в обучающей последовательности модель предсказывает вероятностное распределение следующего токена, и параметры обновляются для максимизации вероятности корректного токена.

Вычислительные требования предобучения огромны: GPT-3 потребовал $\sim 3.14 \times 10^{23}$ FLOP, что соответствует тысячам GPU-месяцев и стоимости в миллионы долларов. Исследования законов масштабирования [11, 12] показывают предсказуемую зависимость производительности от размера модели, объема данных и вычислительных ресурсов. Работа [12] установила оптимальный баланс между размером модели и количеством обучающих токенов для заданного бюджета вычислений. При достижении определенного масштаба возникают качественно новые способности (арифметика, рассуждения, программирование), которые не проявляются в меньших моделях.

Предобученные модели адаптируются для конкретных задач через дообучение на специализированных датасетах с парами вход-выход (supervised fine-tuning). Этот процесс настраивает параметры модели для оптимальной производительности на целевой задаче.

Instruction tuning [14, 15] представляет обучение на разнообразных датасетах, сформатированных как инструкции с примерами. Модель обучается интерпретировать и выполнять явные указания для широкого спектра задач. Модели, обученные исключительно на предсказании текста из интернета, могут генерировать нежелательный контент, поскольку обучающие данные содержат токсичность, предвзятости и неполезные паттерны. Reinforcement Learning from Human Feedback (RLHF) [13] решает эту проблему через трехэтапный процесс:

1) сбор предпочтений, когда формируется датасет сравнительных предпочтений;
2) обучение отдельной нейросети предсказывать человеческие оценки качества для пары (запрос, ответ), она выступает в роли автоматизированного оценщика;
3) оптимизация через RL.

Результатом является модель, генерирующая более полезные, правдивые и безопасные ответы. InstructGPT и ChatGPT строятся на этой методологии. Альтернативные подходы включают Constitutional AI [16], где модель самостоятельно критикует и улучшает свои ответы.

Для эффективной адаптации крупных языковых моделей, требующей значительно меньше ресурсов, чем полный fine-tuning, применяются следующие методы: LoRA (Low-Rank Adaptation) [17], добавляющий обучаемые низкоранговые матрицы к замороженным весам (обучается менее 1 % параметров с сохранением $\sim$99 % качества); квантизация [18], снижающая разрядность весов и уменьшающая требования к памяти в 2–8 раз при минимальной потере качества; а также Mixture of Experts (MoE), активирующий подмножество параметров для каждого токена через механизм маршрутизации [19], что позволяет наращивать емкость модели без пропорционального роста вычислительных затрат.

Несмотря на впечатляющие возможности, современные LLM характеризуются рядом значительных ограничений. Галлюцинации, генерация правдоподобных, но фактически неверных утверждений указывают на существующий разрыв между статистическим сопоставлением паттернов и символьным рассуждением. Ряд алгоритмов [20, 21] частично смягчает эти ограничения, но не решает их полностью.

Информационное поле объектов и формирование представлений в LLM

Представления объектов в больших языковых моделях формируются на основе обучающих данных, однако механизмы этого формирования выходят за рамки простого количественного накопления упоминаний. Информационное поле объекта O можно определить как множество всех информационных единиц, доступных в цифровом пространстве и содержащих прямые или косвенные упоминания, описания или ассоциации с данным объектом [22]. Формально:

$$IF(O) = \{i \in I \mid R(i,O) > \theta\},$$

где $R(i,O)$ представляет функцию релевантности информационной единицы i относительно объекта O, а θ — пороговое значение минимальной релевантности. Границы информационного поля носят размытый характер, поскольку релевантность варьируется от прямых упоминаний до сложных контекстуальных ассоциаций.

Информационное поле характеризуется пространственной неоднородностью: различные области цифрового пространства содержат неравномерное распределение информации об объекте. Некоторые источники аккумулируют значительные объемы релевантной информации, в то время как другие сегменты практически не содержат упоминаний. Центральной характеристикой поля является его плотность — интегральная мера концентрации качественной информации в единице информационного пространства, учитывающая не только объем данных, но и их уникальность, релевантность, авторитетность источников и временную актуальность.

Описанная концепция имеет соответствие в архитектуре LLM. Пространство эмбеддингов, формируемое в процессе обучения, представляет собой непрерывное многомерное пространство, в котором векторные представления токенов, предложений и документов располагаются в соответствии с их семантическими отношениями. Это пространство можно интерпретировать как материализацию информационного поля обучающего материала.

Объекты с высокой плотностью качественной информации в обучающем корпусе формируют четко определенные регионы в пространстве эмбеддингов с сильными активациями соответствующих нейронов. Напротив, объекты с разреженным или противоречивым информационным полем порождают размытые, слабо дифференцированные представления, что проявляется в неуверенности модели и склонности к галлюцинациям при генерации.

Механизм самовнимания, рассмотренный в разделе «Архитектура Transformer: сердце современных LLM», может быть интерпретирован как динамическое вычисление локального информационного поля для каждого токена. В этом случае функция релевантности $R(i,O)$ из определения информационного поля реализуется через скалярное произведение Query и Key векторов:

$$R\left(token_i, token_j\right) = score(i,j) = Q_i \cdot K_j / \sqrt{d_k}.$$

Веса внимания после нормализации $\alpha_{ij} = softmax(score(i,j))$ представляют распределение релевантности в локальном информационном поле токена. Взвешенная агрегация Values:

$$output_i = \sum_j \alpha_{ij} \cdot V_j$$

формирует контекстуализированное представление, интегрирующее информацию из релевантной окрестности. Таким образом, каждый слой Transformer выполняет операцию по уточнению границ информационного поля и извлечению наиболее релевантной информации для текущего контекста. Многослойная архитектура позволяет последовательно расширять радиус информационного поля и строить иерархию абстракций от локальных паттернов к глобальным семантическим структурам.

Ряд современных исследований, проводимых в том числе компанией Newsguard, начинают демонстрировать, что расширение информационного охвата и интеграция веб-поиска в языковые модели привели к заметному ухудшению точности результатов. В контексте событий реального времени модели чаще усиливали ложные нарративы, не дифференцируя авторитетные источники от дезинформационных. Это явление можно объяснить через концепцию информационного поля: включение большого объема низкокачественной информации создает шум в пространстве эмбеддингов, размывая границы между достоверными и ложными представлениями.

Формально, если $Q(e)$ характеризует качество информации, то средняя плотность качественной информации:

$$\rho_{quality}(O) = \left(\int Q(e) \cdot \delta(e \in IF\,(O)\, de \right) / |IF(O)|$$

может снижаться при росте $|IF\,(O)\,|$ (размера информационного поля), если новые информационные единицы имеют низкое качество $Q\,(e) \ll 1$. Модель, обученная на таком «загрязненном» поле, формирует представления, смешивающие достоверные и ложные паттерны, что дополнительно проявляется в генерации противоречивой или недостоверной информации.

Оценка плотности и качества информационного поля требует систематического анализа корпусов данных. Методы [25, 26] позволяют количественно характеризовать распределение упоминаний объектов и выявлять тональную окраску информации, что может оказаться эффективным для объектов, окруженных противоположными мнениями.

Концепция информационного поля предлагает объяснительный фреймворк для ряда практических проблем LLM и указывает направления их решения. Retrieval-Augmented Generation (RAG) [27] может быть интерпретирован как механизм фильтрации информационного поля, когда вместо опоры на все параметрическое знание (сжатое информационное поле обучающего корпуса) модель получает доступ к фильтрованному подмножеству с высокой плотностью качественной информации и реализует функцию релевантности $R\,(i,O)$, извлекая документы из верифицированной базы знаний, что повышает $\rho_{quality}(O)$ для конкретного запроса.

Для объектов, обладающих неоднородным информационным полем с полярными точками зрения, создание нейтральной модели, удовлетворяющей все стороны, практически невозможно, поскольку различные сегменты информационного поля содержат противоречивые паттерны. Перспективным направлением является разработка моделей, способных явно представлять множественные перспективы и их информационное основание, вместо усреднения в единое размытое представление.

Концепция информационного поля представляет собой попытку создать концептуальный мост между архитектурными механизмами Transformer и эпистемологическими вопросами формирования представлений в LLM. Понимание того, как плотность и качество информационного окружения объектов в обучающих данных трансформируется в структуру пространства эмбеддингов и влияет на генеративное поведение модели, может стать эффективным дополнением для разработки более надежных и интерпретируемых систем искусственного интеллекта.

Заключение

Современные большие языковые модели представляют собой результат конвергенции нескольких фундаментальных инноваций. Архитектура Transformer через механизм самовнимания обеспечила эффективное решение задачи контекстуализации и параллельной обработки последовательностей, что позволило масштабировать модели до сотен миллиардов параметров. Парадигма предобучения на общих текстовых корпусах с последующей адаптацией трансформировала методологию обработки естественного языка, создав универсальные базовые модели. Законы масштабирования установили предсказуемые зависимости между размером модели, данными и производительностью.

Концепция информационного поля, представленная в разделе «Информационное поле объектов и формирование представлений в LLM», обеспечивает объяснительный фреймворк для понимания связи между характеристиками обучающих данных и формированием представлений в пространстве эмбеддингов модели. Плотность качественной информации об объекте в цифровом пространстве определяет устойчивость и надежность его представления в параметрической памяти. Эмпирические свидетельства деградации точности при включении низкокачественных источников демонстрируют необходимость переосмысления традиционного подхода максимизации объема данных в пользу кураторских методов и оптимизации качества информационного окружения.

Несмотря на впечатляющие возможности, современные LLM характеризуются фундаментальными ограничениями: галлюцинации отражают обучение на правдоподобности вместо истинности, сложности многошагового рассуждения указывают на разрыв между статистическим сопоставлением паттернов и символьным. Перспективные направления включают мультимодальность, интеграцию с символьными системами, улучшение эффективности через архитектурные инновации для адаптации к динамически изменяющемуся информационному полю.

Трансформационное влияние LLM на общество требует внимательного рассмотрения этических аспектов: предвзятости в данных, потенциал злоупотреблений, влияние на рынок труда и др. Методы выравнивания представляют технические подходы к созданию более безопасных систем, однако требуют дополнения социальными и регуляторными механизмами. Ответственное развитие технологии требует междисциплинарного подхода, комбинирующего достижения машинного обучения, лингвистики, когнитивистики и этики для создания систем, которые не только мощны, но и надежны, интерпретируемы и выровнены с человеческими ценностями.

ЛИТЕРАТУРА

1. Vaswani A., Shazeer N., Parmar N. et al. Attention is All You Need. 2017. arXiv:1706.03762. DOI: 10.48550/arXiv.1706.03762.
2. Bahdanau D., Cho K., Bengio Y. Neural Machine Translation by Jointly Learning to Align and Translate. 2016. arXiv:1409.0473.
3. Radford A., Narasimhan K., Salimans T., Sutskever I. *Improving Language Understanding by Generative Pre-Training*. 2018. Режим доступа: https://cdn.openai.com/research-covers/language-unsupervised/language_understanding_paper.pdf.
4. Radford A., Wu J., Child R. et al. *Language Models are Unsupervised Multitask Learners*. 2019. Режим доступа: https://cdn.openai.com/better-language-models/language_models_are_unsupervised_multitask_learners.pdf.
5. Brown T. B., Mann B., Ryder N. et al. Language Models are Few-Shot Learners. 2020. arXiv:2005.14165.
6. GPT-4 Technical Report. arXiv:2303.08774. DOI: 10.48550/arXiv.2303.08774.
7. Devlin J., Chang M.-W., Lee K., Toutanova K. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. 2018. arXiv:1810.04805. DOI: 10.18653/v1/N19-1423.
8. Raffel C., Shazeer N., Roberts A. et al. Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer. *Journal of Machine Learning Research.* 2020;21:1–67. arXiv:1910.10683. DOI: 10.48550/arXiv.1910.10683.
9. Touvron H., Lavril T., Izacard G. et al. LLaMA: Open and Efficient Foundation Language Models. 2023. arXiv:2302.13971. DOI: 10.48550/arXiv.2302.13971.
10. Jiang A. Q., Sablayrolles A., Mensch A. et al. Mistral 7B. 2023. arXiv:2310.06825. DOI: 10.48550/arXiv.2310.06825.
11. Kaplan J., McCandlish S., Henighan T. et al. Scaling Laws for Neural Language Models. 2020. arXiv:2001.08361. DOI: 10.48550/arXiv.2001.08361.
12. Hoffmann J., Borgeaud S., Mensch A. et al. Training Compute-Optimal Large Language Models. *Advances in Neural Information Processing Systems (NeurIPS)*. 2022;35:30016-30030. arXiv:2203.15556.
13. Ouyang L., Wu J., Jiang X. et al. Training Language Models to Follow Instructions with Human Feedback. *Advances in Neural Information Processing Systems (NeurIPS)*. 2022;35:27730–27744. arXiv:2203.02155.
14. Wei J., Bosma M., Zhao V. Y. et al. Finetuned Language Models Are Zero-Shot Learners. *International Conference on Learning Representations (ICLR)*. 2022. arXiv:2109.01652.
15. Chung H. W., Hou L., Longpre S. et al. Scaling Instruction-Finetuned Language Models. 2022. arXiv:2210.11416. DOI: 10.48550/arXiv.2210.11416.
16. Bai Y. et al. Constitutional AI: Harmlessness from AI Feedback. 2022. arXiv:2212.08073. DOI: 10.48550/arXiv.2212.08073.
17. Hu E. J., Shen Y., Wallis P. et al. LoRA: Low-Rank Adaptation of Large Language Models. *International Conference on Learning Representations (ICLR)*. 2023. arXiv:2106.09685. DOI: 10.48550/arXiv.2106.09685.
18. Dettmers T., Lewis M., Belkada Y., Zettlemoyer L. LLM.int8(): 8-bit Matrix Multiplication for Transformers at Scale. *Advances in Neural Information Processing Systems (NeurIPS)*. 2022;35:30318–30332. arXiv:2208.07339.
19. Shazeer N., Mirhoseini A., Maziarz K. et al. Outrageously Large Neural Networks: The Sparsely-Gated Mixture-of-Experts Layer. *International Conference on Learning Representations (ICLR)*. 2017. arXiv:1701.06538.

20. Wei J., Wang X., Schuurmans D. et al. Chain-of-Thought Prompting Elicits Reasoning in Large Language Models. *Advances in Neural Information Processing Systems (NeurIPS)*. 2022;35:24824–24837. arXiv:2201.11903.
21. Yao S., Yu D., Zhao J. et al. Tree of Thoughts: Deliberate Problem Solving with Large Language Models. 2023. arXiv:2305.10601. DOI: 10.48550/arXiv.2305.10601.
22. Бондарева Н. А., Бондарев А. Е., Андреев С. В., Рыжова И. Г. Информационная плотность объектов в цифровой среде: теоретические основы. *Научная визуализация*. 2025;17(4):87–98. DOI: 10.26583/sv.17.4.09.
23. Bommasani R., Hudson D. A., Adeli E. et al. On the Opportunities and Risks of Foundation Models. 2021. arXiv:2108.07258. DOI: 10.48550/arXiv.2108.07258.
24. Goodfellow I., Bengio Y., Courville A. *Deep Learning*. MIT Press; 2016. ISBN: 9780262035613.
25. McEnery T., Hardie A. *Corpus Linguistics: Method, Theory and Practice*. Cambridge University Press; 2011. ISBN: 9780521761942.
26. Liu B. *Sentiment Analysis: Mining Opinions, Sentiments, and Emotions*. Cambridge University Press; 2015. ISBN: 9781107017894.
27. Lewis P., Perez E., Piktus A. et al. Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks. *Advances in Neural Information Processing Systems (NeurIPS)*. 2020;33:9459-9474. arXiv:2005.11401.