СТРЮКОВ П. В., ГЕРБЕРТ Д. В., КОЖЕВНИКОВА П. В. КВАНТОВАНИЕ БОЛЬШИХ ЯЗЫКОВЫХ МОДЕЛЕЙ: СОВРЕМЕННЫЕ МЕТОДЫ, ВЫЗОВЫ И ПЕРСПЕКТИВЫ ОПТИМИЗАЦИИ

УДК 025.4.03; 002.6:004.65, ГРНТИ 20.23.17

Квантование больших языковых моделей: современные методы, вызовы и перспективы оптимизации

Quantization of large language models: modern methods, challenges and prospects for optimization

П. В. Стрюков, Д. В. Герберт, П. В. Кожевникова

Ухтинский государственный технический университет, г. Ухта

P. V. Stryukov, D. V. Herbert, P. V. Kozhevnikova

Ukhta State Techincal University, Ukhta

В статье рассматриваются современные методы квантования больших языковых моделей (LLM), такие как посттренировочное квантование (PTQ) и квантование с дообучением (QAT), а также их влияние на производительность и качество моделей. Освещены ключевые вызовы и перспективные направления развития технологии, включая гибридные методы и аппаратно-ориентированные решения.

The article discusses modern methods of quantization of large language models (LLM), such as post-training quantization (PTQ) and quantization with additional training (QAT), as well as their impact on the performance and quality of models. Key challenges and promising areas of technology development, including hybrid methods and hardware-oriented solutions, are highlighted.

Ключевые слова: *квантование, большие языковые модели, оптимизация инференса*

Keywords*: *quantization, large language models, inference optimization

Введение

Современные большие языковые модели (LLM) демонстрируют значительный прогресс в обработке естественного языка, генерации кода и решении других сложных интеллектуальных задач. Однако их широкое практическое применение ограничено существенными вычислительными ресурсами, необходимыми для их обучения и эксплуатации. Модели, насчитывающие сотни миллиардов параметров, такие как DeepSeek-R1 (671 млрд параметров) или Llama 4 Maverick (400 млрд параметров, требуют кластеров высокопроизводительных графических процессоров (GPU) и значительных энергетических затрат. В данном контексте, квантование LLM выступает как ключевая технология, позволяющая сократить требования к

объему памяти и вычислительным мощностям, минимизируя при этом деградацию качества модели [1].

Проблема вычислительных ресурсов и памяти в LLM

1. Масштаб проблемы

Современные LLM оперируют сотнями миллиардов параметров, традиционно представляемых 32-битными числами с плавающей запятой (FP32). Например, для модели с 671 млрд параметров только для хранения весов требуется около 2.68 ТБ дискового пространства. Даже при использовании формата половинной точности FP16 (16 бит на параметр) или BF16, объем занимаемой памяти остается значительным, что затрудняет развертывание таких моделей на большинстве пользовательских устройств и ограничивает их применение в коммерческих сценариях с жесткими бюджетными рамками. Помимо весов, существенный объем памяти во время инференса занимает так называемый KV-кэш, хранящий промежуточные состояния внимания (attention) для обработанных токенов, что особенно критично для моделей с большим окном контекста.

2. Ограничения традиционных методов сжатия данных

Стандартные алгоритмы сжатия информации общего назначения, такие как ZIP (для архивации данных) или JPEG (для изображений), оказываются малоэффективными или неприменимыми для компрессии LLM по ряду причин:

a) **Чувствительность к изменениям параметров:** В отличие от пикселей изображения или символов текста, где локальные потери могут быть несущественны, параметры LLM обладают сложной семантической нагрузкой и взаимозависимостями. Небольшие изменения в значениях весов могут привести к значительным искажениям в поведении модели.

b) **Требования к скорости доступа:** Веса LLM должны быть доступны для выполнения матричных операций (ключевых для трансформерных архитектур) с минимальной задержкой во время инференса. Методы сжатия, требующие последовательной декомпрессии всего блока данных, здесь неприменимы.

c) **Нелокальное влияние ошибок:** В LLM ошибки, вызванные неточной декомпрессией или представлением даже небольшой группы параметров, могут каскадно распространяться по сети, приводя к существенной деградации итогового результата.

Следовательно, квантование LLM — это не просто сжатие данных, а специфическая для нейронных сетей техника адаптации представления параметров с учетом особенностей их распределений и влияния на вычислительные процессы, нацеленная на сохранение функциональности модели.

Эволюция методов квантования LLM

1. Ранние подходы: Посттренировочное квантование (PTQ)

Первые методы квантования, применяемые к LLM, были преимущественно посттренировочными (Post-Training Quantization, PTQ) и основывались на

преобразовании предварительно обученных весов из FP32/FP16 в целочисленные форматы с низкой битностью.

a) **Линейная (аффинная) квантизация:** Преобразование весов w в квантованные значения q по формуле $w \approx S * q + Z$, где S – масштаб (scale), Z – точка нуля (zero-point). Калибровочный набор данных используется для определения оптимальных S и Z.

b) **Симметричная квантизация:** Упрощенный вариант, где Z=0, что может ускорить вычисления, но менее точно представляет асимметричные распределения весов. Эти подходы позволяли сократить размер модели в 2-4 раза (например, при переходе с FP16 на INT8) и ускорить вычисления, но часто сопровождались заметной потерей качества, особенно при агрессивном квантовании (например, до 4 бит и ниже) без дополнительных эвристик.

2. Современные продвинутые методы PTQ.

Для преодоления ограничений ранних PTQ-методов были разработаны более сложные техники:

a) **Методы, учитывающие распределение и значимость весов/активаций:**

- **GPTQ (Generalized Post-Training Quantization**: Послойное квантование весов с использованием информации из приближенного гессиана для минимизации ошибки квантования [2].
- **AWQ (Activation-aware Weight Quantization)** : Определяет важность весов на основе величины соответствующих им активаций, квантуя менее значимые веса более агрессивно.
- **SpQR (Sparse-Quantized Representation)** : Идентифицирует и сохраняет значения-выбросы (outliers) в более высоком разрешении, в то время как остальные веса квантуются более грубо.

b) **Блочное (групповое) квантование:** Матрицы весов разделяются на небольшие блоки (например, размером 64 или 128 элементов), и для каждого блока вычисляются свои параметры квантования (S и Z). Это позволяет лучше адаптироваться к локальным изменениям в распределении весов. Примером формата, использующего этот подход, является NF4 (NormalFloat4), представленный в QLoRA.

c) **Смешанная точность (Mixed Precision):** Использование различных битностей для разных слоев или даже типов тензоров внутри модели. Например, в форматах llama.cpp (таких как Q3_K_M) тензоры attention.wv могут использовать квантование Q4_K, а остальные — Q3_K, для баланса между размером и качеством.

d) **Экстремальное квантование (1-3 бита):**

- Методы, позволяющие использовать 1-2 бита на параметр (например, IQ2_XS, IQ1_S в llama.cpp, или BitNet b1.58, который предлагает 1.58-битные веса и специализированную архитектуру) [3].
- **QUIP#** : Использует нелинейное квантование и адаптивные проекции для достижения почти безпотерьного сжатия до 2-3 бит на параметр, но может требовать специфической аппаратной поддержки для эффективного инференса.

e) **Квантование KV-кэша:** Помимо весов, активно исследуется квантование KV-кэша (например, до INT8) для снижения пикового потребления памяти во время генерации длинных последовательностей, что особенно актуально для моделей с большим контекстным окном.

3. Квантование с дообучением: Quantization-Aware Training (QAT).

QAT представляет собой подход, при котором процесс квантования имитируется непосредственно во время обучения или дообучения модели.

a) **Принцип работы:** На прямом проходе сети веса и/или активации квантуются, а на обратном проходе градиенты пропускаются через операцию квантования (часто с использованием Straight-Through Estimator, STE), позволяя модели адаптироваться к дискретизированным значениям (Рисунок 1).

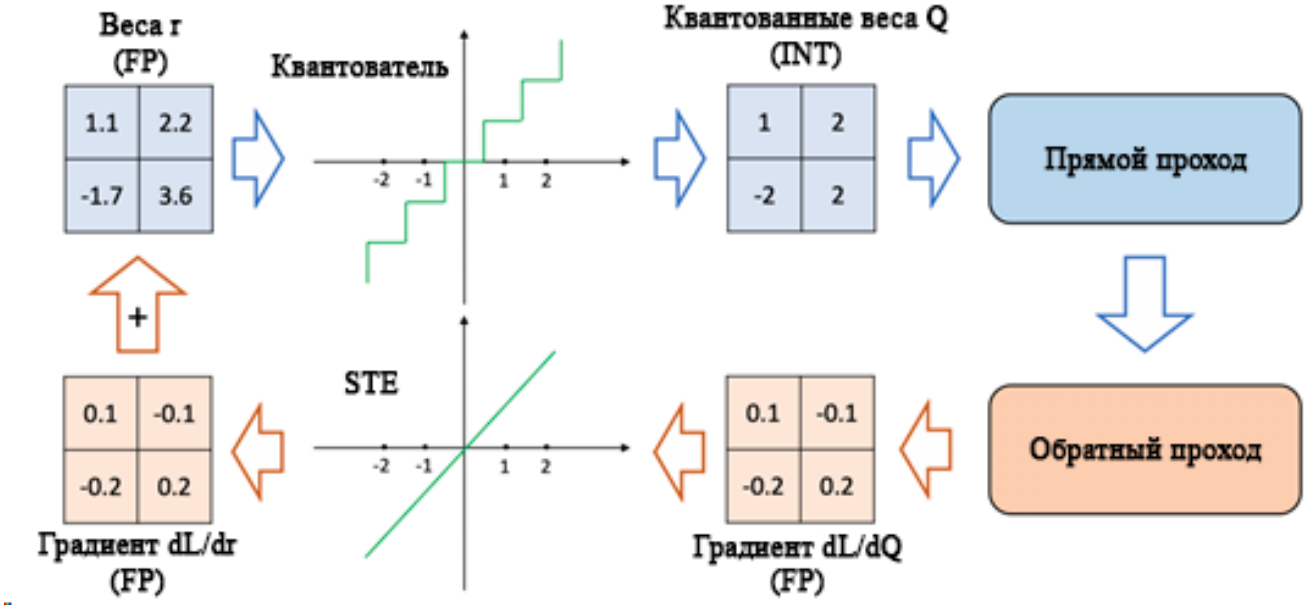

Рисунок 1. Принцип работы квантования с дообучением

b) **Преимущества:** QAT обычно приводит к значительно меньшим потерям точности по сравнению с PTQ, особенно при агрессивных уровнях квантования (например, 4 бита и ниже). Потери могут быть на 50-70% меньше.

c) **Недостатки:** Существенно увеличивает время и вычислительные затраты на обучение (на 30-50% и более), так как требует эмуляции квантования на каждой итерации.

d) **Применение:** Особенно эффективно для моделей, предназначенных для развертывания на устройствах с жесткими ограничениями по ресурсам, где необходимо достичь максимального сжатия при минимальных потерях.

Вычислительные затраты и влияние на производительность

1. Затраты на процесс квантования.

Процесс квантования сам по себе требует вычислительных ресурсов, которые варьируются в зависимости от метода:

a) **Посттренировочное квантование (PTQ):**

- Обычно требует однократного прохода по модели (или ее части) с использованием небольшого калибровочного набора данных для сбора статистики и преобразования весов.
- Для модели класса Llama-3-8B, простые PTQ методы могут занимать от нескольких минут до нескольких часов на современном CPU или GPU. Более сложные PTQ (GPTQ, AWQ) требуют больше времени (несколько часов на GPU).
- Потребление памяти во время PTQ обычно сопоставимо с размером исходной модели плюс память для калибровочных данных и промежуточных вычислений.

b) **Quantization-Aware Training (QAT):**

- Как упоминалось, увеличивает время обучения на 30-50% и более.
- Требует хранения дополнительных состояний для эмуляции квантования и расчета градиентов.
- Для крупных моделей (например, 70B параметров) QAT может потребовать значительных кластерных ресурсов и дней дополнительного обучения.

c) **Методы, сочетающие квантование с адаптацией (например, QLoRA):**

- QLoRA сочетает 4-битное квантование базовой модели (NF4) с дообучением небольшого количества адаптивных параметров (LoRA).
- Это позволяет значительно снизить требования к памяти для дообучения (например, модель 65B может быть дообучена на одном GPU с 48 ГБ VRAM) по сравнению с полным дообучением даже в FP16. Затраты на само квантование здесь — это затраты PTQ для базовой модели.

2. Влияние формата квантования на инференс

Выбор метода и формата квантования оказывает прямое влияние на скорость инференса, качество генерации и аппаратные требования:

a) **Скорость инференса:**

- Переход с FP16 на INT8 может ускорить матричные умножения в 2-3 раза на аппаратном обеспечении с поддержкой INT8-операций (например, тензорные ядра NVIDIA GPU).
- Использование 4-битного квантования (например, Q4_0, NF4) может дать дополнительное ускорение, особенно если оно эффективно реализовано на целевой архитектуре (например, за счет загрузки большего числа весов в кэш или использования специализированных ядер).
- Однако, экстремальное квантование (1-2 бита) не всегда приводит к пропорциональному ускорению, так как может требовать сложных операций деквантования "на лету" или не иметь оптимизированной аппаратной поддержки.

b) **Качество генерации**:

- Переход с FP16 на Q5_0 (5-битное квантование) может увеличить перплексию на валидационном наборе незначительно (чаще всего примерно, не более чем на 2 пункта).

- Более агрессивное квантование (например, Q2_K, ~2.5 бита) может привести к заметному росту перплексии, что может быть критично для некоторых задач.
- Методы вроде QUIP# или продвинутые PTQ (AWQ, SpQR) стремятся минимизировать эту деградацию даже при низких битрейтах.

c) **Аппаратные требования:**

- Модель Llama 70B в FP16 требует около 140 ГБ VRAM. При квантовании в Q4_0 (4-бита) ее размер уменьшается примерно до 35-40 ГБ, что делает возможным запуск на одном или двух высокопроизводительных потребительских GPU.
- Технологии вроде QUIP# или HIGGS (Представляющие собой сочетание агрессивного квантования, вплоть, до 1-2 бит, и оптимизированной архитектуры), нацелены на запуск LLM даже на мобильных устройствах.

Методы оценки снижения качества модели.

Оценка влияния квантования на "интеллектуальные" способности модели является нетривиальной задачей. Используются следующие подходы:

a) **Перплексия (Perplexity, PPL):**

- Основная метрика для языковых моделей, измеряющая неопределенность модели при предсказании следующего токена. PPL = exp(средняя отрицательная логарифмическая вероятность на тестовом наборе)

$$PPL(X) = exp\left\{-\frac{1}{t}\sum_{i}^{t} logp_\theta(x_i|x_{<i})\right\}$$

- Ниже PPL – лучше. Например, для Llama 70B разница в PPL между FP16 и Q5_K_M может составлять менее 0.5%.
- Важно отметить, что перплексия чувствительна к токенизации и набору данных для оценки, поэтому ее следует использовать с осторожностью при сравнении принципиально разных моделей. Однако она остается полезным индикатором изменений внутри одной модели при различных уровнях квантования [4].

b) **Задача-специфичные метрики (Task-specific benchmarks):**

- Наборы данных для оценки общего языкового понимания: GLUE, SuperGLUE, MMLU (Massive Multitask Language Understanding).
- Метрики для задач генерации: BLEU, ROUGE для машинного перевода и суммаризации.
- Метрики для генерации кода: Pass@k (например, на HumanEval).
- Оценка на задачах здравого смысла (commonsense reasoning): HellaSwag, WinoGrande.

c) **Качественный анализ и человеческая оценка:**

- Экспертная оценка связности, релевантности и фактической корректности генерируемого текста.
- Тестирование на сложных, редких или провокационных запросах.

- Анализ склонности к "галлюцинациям" (генерации неверной информации) и противоречиям.
- Слепые тесты с привлечением людей для сравнения ответов квантованной и неквантованной моделей. В современных исследованиях принято использовать комбинацию автоматических метрик и качественного анализа для всесторонней оценки последствий квантования.

Использование нескольких видов тестов дает более реальную оценку того, насколько квантование ухудшает модель. В качестве достаточно наглядного примера снижения качества модели с учетом квантования можно привести следующие бенчмарки (Рисунок 2):

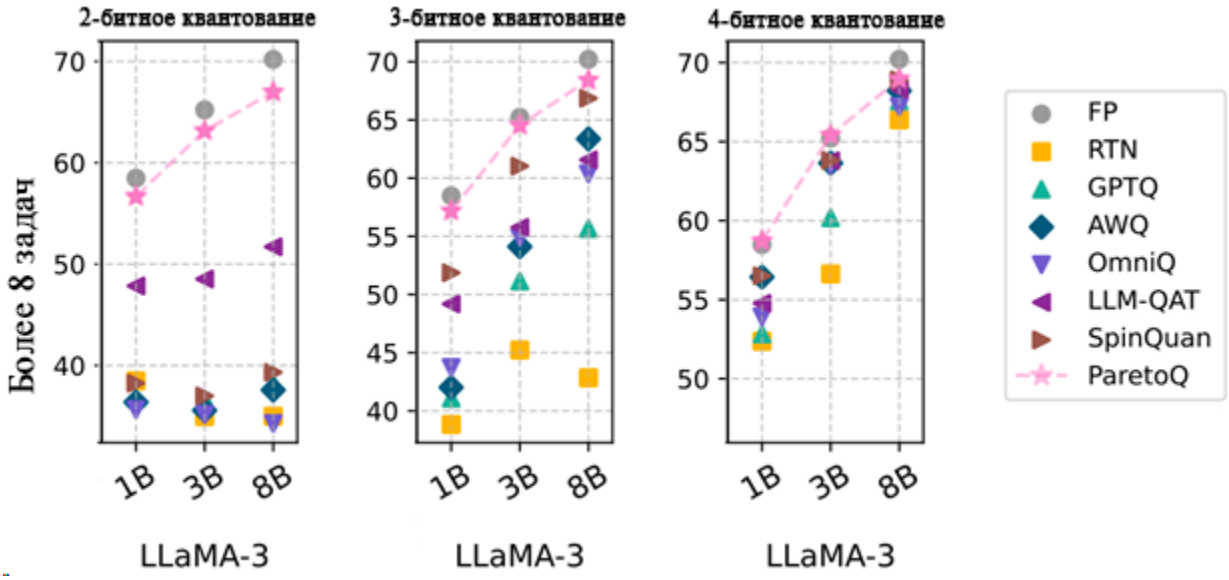

Рисунок 2. Бенчмарки качества моделей

Вызовы и перспективные направления развития

Несмотря на достигнутые успехи, область квантования LLM сталкивается с рядом вызовов. К примеру, потерям при квантовании небольшие модели подвержены гораздо больше чем крупные, из-за плотности знаний, также на любых моделях заметна потеря многоязычия при агрессивном квантовании. Все это открывает новые направления для исследований:

a) **Адаптивное и нелинейное квантование:**

- Разработка методов, динамически изменяющих битность или параметры квантования для разных слоев, блоков или даже отдельных весов в зависимости от их чувствительности или статистических свойств.
- Исследование более сложных, нелинейных функций квантования, выходящих за рамки простой аффинной модели, для лучшего представления сложных распределений весов (как в QUIP#).

b) **Гибридные методы оптимизации:**

- Интеграция квантования с другими техниками компрессии моделей, такими как прореживание (pruning), дистилляция знаний (knowledge distillation) и низкоранговая факторизация (low-rank factorization, LoRA). QLoRA является ярким примером такого синергетического подхода.

c) **Аппаратно-ориентированное квантование (Hardware-aware Quantization):**

- Разработка схем и форматов квантования, оптимизированных под конкретные аппаратные архитектуры (CPU, GPU, NPU, TPU), включая специализированные ускорители (например, Groq).
- Исследование возможностей для LLM с экстремально низкой битностью (например, 1-битные модели типа BitNet), которые могут выполняться с использованием преимущественно битовых операций вместо дорогостоящих умножений.

d) **Квантование для моделей с расширенной памятью/контекстом:**

- Эффективное квантование не только весов, но и механизмов работы с длинными контекстами (например, KV-кэша в моделях типа StreamingLLM или внешних модулей памяти).

e) **Теоретическое понимание и гарантии:**

- Разработка теоретической базы, объясняющей, почему квантование работает, и каковы пределы сжатия без потери существенной информации для различных архитектур и задач.

f) **Автоматизация и поиск оптимальных стратегий квантования (AutoML for Quantization):**

- Применение техник автоматического машинного обучения для поиска оптимальных конфигураций квантования (битности, размеров блоков, калибровочных параметров) для конкретной модели и целевого устройства.

Заключение

Квантование больших языковых моделей эволюционировало из простых техник сокращения битности в сложный набор инструментов и методологий, позволяющих существенно расширить применимость LLM. Современные подходы, включая продвинутые PTQ-методы (GPTQ, AWQ, SpQR), QAT и гибридные решения (QLoRA), демонстрируют возможность достижения эффективного компромисса между размером модели, скоростью инференса и сохранением высокого качества генерации. Дальнейшее развитие этой области, вероятно, будет сосредоточено на создании более адаптивных и аппаратно-ориентированных схем квантования, разработке надежных методов оценки деградации качества и глубокой интеграции квантования с другими парадигмами оптимизации нейронных сетей. Успехи в области квантования открывают перспективы для инноваций в различных сферах применения LLM, способствуя их демократизации и доступности для широкого круга исследователей, разработчиков и конечных пользователей, в том числе на устройствах с ограниченными ресурсами.

Список использованных источников и литературы:

1. Zechun Liu, Changsheng Zhao. ParetoQ: Scaling Laws in Extremely Low-bit LLM Quantization [Электронный ресурс] // arXiv. — 2025. — arXiv:2502.02631. — URL: https://arxiv.org/pdf/2502.02631 (дата обращения: 05.05.2025).

2. Elias Frantar, Saleh Ashkboos: GPTQ: Accurate Post-Training Quantization for Generative Pre-trained Transformers [Электронный ресурс] // arXiv. — 2022. — arXiv:2210.17323. — URL: https://arxiv.org/abs/2210.17323 (дата обращения: 05.05.2025).

3. Wenhua Cheng, Weiwei Zhang: Optimize Weight Rounding via Signed Gradient Descent for the Quantization of LLMs [Электронный ресурс] // arXiv. — 2023. — arXiv:2309.05516. — URL: https://arxiv.org/pdf/2309.05516 (дата обращения: 05.05.2025).

4. Ji Lin, Jiaming Tang: AWQ: Activation-aware Weight Quantization for LLM Compression and Acceleration [Электронный ресурс] // arXiv. — 2023. — arXiv:2306.00978. — URL: https://arxiv.org/abs/2306.00978 (дата обращения: 05.05.2025).

List of references

1. Zechun Liu, Changsheng Zhao. ParetoQ: Scaling Laws in Extremely Low-bit LLM Quantization [Электронный ресурс] // arXiv. — 2025. — arXiv:2502.02631. — URL: https://arxiv.org/pdf/2502.02631 (дата обращения: 05.05.2025).

2. Elias Frantar, Saleh Ashkboos: GPTQ: Accurate Post-Training Quantization for Generative Pre-trained Transformers [Электронный ресурс] // arXiv. — 2022. — arXiv:2210.17323. — URL: https://arxiv.org/abs/2210.17323 (дата обращения: 05.05.2025).

3. Wenhua Cheng, Weiwei Zhang: Optimize Weight Rounding via Signed Gradient Descent for the Quantization of LLMs [Электронный ресурс] // arXiv. — 2023. — arXiv:2309.05516. — URL: https://arxiv.org/pdf/2309.05516 (дата обращения: 05.05.2025).

4. Ji Lin, Jiaming Tang: AWQ: Activation-aware Weight Quantization for LLM Compression and Acceleration [Электронный ресурс] // arXiv. — 2023. — arXiv:2306.00978. — URL: https://arxiv.org/abs/2306.00978 (дата обращения: 05.05.2025).