https://doi.org/10.17323/jle.2024.22368

В поисках подводных камней: сингулярное разложение матриц для сжатия языковых моделей

Плетенев Сергей Александрович [1, 2]

[1] AIRI, г. Москва, Россия

[2] Сколтех, г. Москва, Россия

АННОТАЦИЯ

Введение: Современная вычислительная лингвистика активно использует большие языковые модели, которые показывают впечатляющие результаты в различных задачах Natural Language Inference (NLI). Однако для обучения и развертывания этих моделей требуются значительные вычислительные ресурсы. Чтобы решить эту проблему, были разработаны разнообразные методы сжатия и ускорения, такие как квантование, прунинг и факторизация. Каждый из этих подходов имеет свои уникальные особенности, может применяться на различных уровнях архитектуры модели и адаптироваться под разнообразные сценарии использования.

Цель исследования: Целью данного исследования является анализ и оценка техники сжатия на основе факторизации, которая способна значительно снизить вычислительную сложность больших языковых моделей, при этом сохраняя их точность в задачах NLI. Особенно это актуально для приложений, где ресурсы ограничены или требуется низкая задержка.

Методология: Чтобы оценить влияние сжатия на основе факторизации, мы провели эксперименты с помощью «пробинга» моделей. Для начала мы выбрали две популярные модели в качестве базовых: Bert-base и Llama 2. Затем мы применили факторизацию к их слоям, используя различные алгоритмы разложения по сингулярным значениям и изменяя степень сжатия. После этого мы использовали диагностические наборы данных для анализа изменений внутренних представлений и лингвистических знаний сжатых моделей. Мы сравнили эти изменения с тем, как модель решает задачи на естественном языке (NLI), что позволило оценить влияние степени сжатия на эти процессы.

Результаты: Наивная факторизация приводит к значительному падению точности даже при небольшой степени сжатия, что отражается в заметном ухудшении способностей модели понимать текстовые последовательности. «Пробинг-анализ» показал, что сжатые модели теряют важную синтаксическую и семантическую информацию, это согласуется с наблюдаемым нами снижением качества. Однако другие подходы к сжатию, такие как выборочное сжатие наиболее избыточных частей модели или алгоритмы взвешивания, смягчили эти негативные эффекты.

Заключение: Наши результаты показали, что факторизация при правильном применении может существенно сократить вычислительные затраты, сохраняя при этом основные лингвистические возможности больших языковых моделей. Наше исследование может стать основой для создания новых методов сжатия, которые адаптируют методы факторизации к внутренним структурам языковых моделей. Это исследование может помочь будущим ученым найти наиболее эффективные способы сжатия языковых моделей.

КЛЮЧЕВЫЕ СЛОВА

Факторизация, Большие языковые модели, Пробинг языковых свойств, Ресурсно-эффективные методы вычислительной лингвистики.

Для цитирования: Плетенев, С.А. (2024). Исследование ограничений: Понимание недостатков SVD в сжатии языковых моделей. *Journal of Language and Education, 10*(4), 88-99. https://doi.org/10.17323/jle.2024.22368

Correspondence:
Плетенев Сергей Александрович,
pletenev@airi.net

Получена: 16 сентября 2024

Принята: 16 декабря 2024

Опубликована: 30 декабря 2024

ВВЕДЕНИЕ

Большие языковые модели (LLM) привлекают значительное внимание в области искусственного интеллекта благодаря своим замечательным возможностям в понимании и генерации естественного языка (Brown et al., 2020; Devlin et al., 2018). По сравнению со своими предшественниками, современные LLM, такие как ChatGPT или LLaMA (Touvron, Lavril, et al., 2023) демонстрируют улучшенные возможности для любых языковых задач. Эти модели обладают рядом новых способностей, которые обычно не встречаются в небольших и простых моделях,

включая продвинутые многоступенчатые рассуждения и сложное следование инструкциям (Wei et al., 2022). Это подчеркивает значительный потенциал LLM для различных приложений, таких как диалоговые агенты, генерация контента и рефакторинг кода.

Несмотря на эти достижения, внедрение LLM ограничено их значительными требованиями к памяти и вычислительным ресурсам во время генерации (Narayanan et al., 2020). Например, для модели с 8 миллиардами параметров может потребоваться приблизительно 40 ГБ видеопамяти, а потребление памяти для вывода квадратично зависит от длины последовательности (Kaplan et al., 2020). Такие высокие требования к ресурсам создают серьезные проблемы при развертывании LLM на устройствах с ограниченными вычислительными ресурсами и ресурсами памяти, включая аппаратное обеспечение потребительского уровня или мобильные устройства (Lane et al., 2016). Для решения этих проблем используются различные подходы к сжатию моделей, чтобы сократить затраты памяти и вычислений как при обучении, так и генерации LLM (Ganesh et al., 2021).

Сжатие моделей — это область исследований, которая фокусируется на уменьшении размера и сложности моделей глубокого обучения. При этом обычно предполагается, что существующая модель служит основой для методов сжатия (Cheng et al., 2018). Применение этих методов сжатия позволяет повысить доступность использования LLM в условиях ограниченных ресурсов при сохранении их эффективности (Tang et al., 2019).

Чтобы уменьшить требования к ресурсам у языковых моделей, были предложены различные методы сжатия, которые особенно актуальны в сценариях с ограниченными вычислительными ресурсами (Xu et al., 2018). Среди этих методов двумя наиболее популярными являются квантование (Dettmers et al., 2021; N. Wang et al., 2018) и прунинг (Kurtic et al., 2022; Wang et al., 2019a; Zafrir et al., 2021). Квантование предполагает снижение точности (уменьшение битности числа с плавающей запятой) весов и активаций в нейронной сети, в то время как прунинг удаляет ненужные, по мнению алгоритма, связи между нейронами (Han et al., 2015). Неструктурированный прунинг и квантование могут значительно сократить количество параметров и требований к памяти, часто на 50 % и более, без существенного снижения качества. (Guo et al., 2016). Однако для полного использования потенциала ускорения с помощью этих методов обычно требуются специализированные ядра графических процессоров и оптимизированное программное обеспечение (Zhang et al., 2019).

Напротив, методы факторизации, такие как сингулярное разложение матрицы (SVD), обеспечивают сокращение объема памяти и увеличение скорости вычислений без необходимости дополнительной аппаратной или программной оптимизации (Tai et al., 2015). SVD — это простой метод декомпозиции низкого ранга, который широко используется для уменьшения размера эмбедингов (Lan et al., 2019) и слоев моделей на архитектуре трансформер (Michel et al., 2019; Z. Wang et al., 2019b). Подходы, основанные на SVD, часто дают худшие результаты по сравнению с оригинальными моделями или другими методами сжатия (Kim et al., 2015). Такое снижение производительности ограничивает практичность SVD для сжатия LLM, особенно когда требуется высокая точность (Tai et al., 2015).

Учитывая ограничения существующих методов факторизации, возникает потребность в разработке усовершенствованных методов, способных эффективно сжимать LLM без существенной потери производительности. Поэтому целью нашего исследования является изучение новых подходов к факторизации, которые сохранят преимущества SVD и одновременно устранят часть его недостатков. В частности, мы исследуем альтернативные методы декомпозиции, которые могут обеспечить лучшее соотношение между степенью сжатия и точностью модели, тем самым повышая возможность развертывания LLM на устройствах с ограниченными ресурсами и вычислительными возможностями. Для того, чтобы задать направление нашему исследованию, мы сформулировали следующие исследовательские вопросы (RQ):

- **RQ#1**: Связана ли потеря финального качества модели при сжатии с ухудшением внутренних представлений модели?
- **RQ#2**: Как различные методы факторизации влияют на внутренние представления в моделях?
- **RQ#3**: Приводит ли сжатие модели к необратимой потере знаний, и если да, то в какой степени?

Отвечая на эти вопросы, мы стремимся углубить понимание того, как методы сжатия влияют на LLM на репрезентативном уровне, и найти порог сжатия, который минимизирует потери производительности при максимальной эффективности.

ОБЗОР ЛИТЕРАТУРЫ

При обработке данных на естественном языке для оценки качества моделей используются различные оценочные метрики. Эти метрики также используются для проверки моделей после применения различных методов сжатия. В этом разделе мы представляем обзор нескольких методологий факторизации, предложенных в качестве альтернативы или усовершенствования SVD. Мы также проводим обзор соответствующей литературы о влиянии различных методов сжатия на производительность модели. Цель этого обзора — оценить эффективность этих альтернативных подходов к факторизации и их влияние на производительность модели после сжатия.

Методы сжатия моделей

Взвешенный по Фишеру SVD (FWSVD) (Hsu et al., 2022) использует информацию о градиенте для взвешивания сингулярных значений во время декомпозиции, стремясь сохранить важные особенности модели. Несмотря на то, что этот метод продемонстрировал улучшенное качество сжатия, для восстановления качества модели требуется дополнительный этап обучения, который включает в себя повторное обучение модели на первоначальной задаче. Этот дополнительный этап обучения увеличивает вычислительные затраты и может оказаться невозможным для некоторых моделей. Кроме того, в FWSVD применяется равномерное сжатие всех слоев, т. е. один и тот же ранг присваивается каждому сжатому слою без учета индивидуальной значимости. Такой равномерный подход может быть не всегда оптимальным, поскольку некоторые слои оказывают более существенное влияние на производительность модели, чем другие.

Для устранения равномерного сжатия слоев был разработан метод декомпозиции по сингулярным значениям с учетом активации (ASVD) (Yuan et al., 2023), который выборочно сжимает слои на основе определенных критериев, связанных с их влиянием на производительность модели. Выявляя и сжимая только те слои, которые являются менее критичными или потенциально «зашумленными», ASVD обеспечивает сжатие модели без существенной потери качества. Кроме того, этот метод не требует дополнительных ресурсов и временных затрат для вычисления градиентов модели, как в случае с FWSVD, а оперирует активациями, которые могут быть собраны во время генерации модели.

Методы оценки моделей

Различные исследования (Yin et al., 2023; Yuan et al., 2023) показали, что квантование и сокращение могут эффективно уменьшить размер модели с минимальным влиянием на общие показатели производительности. Однако они выявили потенциальные риски, связанная с непреднамеренным отключением важнейших внутренних механизмов. Например, квантование может деактивировать компоненты, которые отвечают за этические аспекты, такие как способность модели предотвращать создание токсичного или неприемлемого контента. Аналогичным образом сокращение может привести к полной неспособности модели отвечать на сложные вопросы по мере увеличения сжатия. Эти примеры вызывают обеспокоенность по поводу более широкого влияния сжатия моделей на поведение и подчеркивают важность тщательной оценки сжатых моделей, выходящей за рамки традиционных показателей производительности, ориентированных на конкретные задачи.

В совокупности эти исследования демонстрируют сложную взаимосвязь между методами сжатия моделей и сохранением их качества и функциональности. Несмотря на то, что такие методы, как FWSVD, ASVD и SVD, предлагают многообещающие возможности для уменьшения размера модели с минимальной потерей производительности, остаются проблемы, связанные с обеспечением сохранности критически важных компонентов и поведением модели после сжатия. Противоречивые результаты (Chen et al., 2020; Yin et al., 2023; Yu & Wu, 2023) относительно низкоранговой природы весов моделей и активаций указывают на необходимость более глубокого понимания внутренних структур нейронных сетей. Это подчеркивает важность выбора подходящих стратегий сжатия, адаптированных к конкретным характеристикам модели и задачам, которые она выполняет, что является ключевым фактором для продвижения разработки эффективных алгоритмов факторизации.

МЕТОД

Факторизация

Наивная реализация SVD

Допустим, что это веса линейного слоя, тогда мы определяем SVD следующим образом: $\mathcal{W} = U\Sigma V^T$ Затем мы используем усеченное произведение $U_r = U[:, :r], \Sigma_r = \Sigma[:r, :r], V_r = V[:, :r]$ для определения веса двух линейных слоев для замены:

$$\begin{aligned} \mathcal{W}_2 &= U_r\sqrt{\Sigma_r} \\ \mathcal{W}_1 &= \sqrt{\Sigma_r}V_r^T \end{aligned}$$

В результате мы получаем приближение весов $\mathcal{W} \approx \mathcal{W}_2\mathcal{W}_1$ и оригинального линейного слоя $Y \approx X\mathcal{W}_1^T\mathcal{W}_2^T + b$. Если n_{in}, n_{out} имеет размерность, то количество параметров в слое до факторизации будет $n_{in} \times n_{out}$, после разложения мы получим размерности $r \times (n_{in} + n_{out})$

Взвешенный по Фишеру SVD (FWSVD)

Метод FWSVD (Hsu et al., 2022) предлагает внедрить информацию Фишера в алгоритмы разложения, чтобы минимизировать разрыв между суммой ошибок для «обученной» задачи. Информация Фишера определяет важность каждого параметра для предсказаний в данной задаче. Следуя подходу, представленному в (Hsu et al., 2022), мы аппроксимируем матрицу Фишера с помощью набора данных $\mathcal{D} = \{d_1, \dots, d_{|\mathcal{D}|}\}$ для каждого из весов в модели $\mathcal{W} \in \mathbb{R}^{I \times J}$:

$$\begin{aligned} \mathcal{I}_\mathcal{W} &= \mathbb{E}[(\frac{\partial}{\partial \mathcal{W}} \log p(\mathcal{D}|\mathcal{W}))^2] \\ \mathcal{I}_\mathcal{W} &\approx \frac{1}{|\mathcal{D}|}\sum_{i=1}^{|\mathcal{D}|}(\frac{\partial}{\partial \mathcal{W}}\mathcal{L}(d_i; \mathcal{W}))^2 \end{aligned}$$

С помощью этого мы хотим решить задачу взвешенного низкорангового приближения:

$$||\sqrt{\mathcal{I}_\mathcal{W}} * (\mathcal{W} - \hat{\mathcal{W}})||^2 \rightarrow \min_{\text{rank}\hat{\mathcal{W}}=r}$$

К сожалению, данная проблема не имеет аналитического решения. В оригинальной статье предлагается суммировать строки матрицы Фишера и решать задачу низкорангового приближения с помощью SVD:

$$\tilde{\mathcal{I}}_{\mathcal{W}} = \text{diag}(\mathcal{I}_{\mathcal{W}} \cdot \mathbf{1}), \hat{\mathcal{W}} = \tilde{\mathcal{I}}_{\mathcal{W}} \mathcal{W} = USV^T,$$

где $\mathbf{1} = (1, \ldots, 1) \in \mathbb{R}^{J \times 1}$. Взвешенные факторы для изначальной матрицы $W \approx \hat{U}\hat{S}\hat{V}^T$ считаются так:

$$\hat{U} = \tilde{\mathcal{I}}_{\mathcal{W}}^{-1} U, \hat{S} = S, \hat{V} = V.$$

В результате мы получаем низкоранговое приближение с оценкой важности для параметров:

$$\text{FWSVD}(w) = \hat{U}\hat{\Sigma}\hat{V} = (\hat{I}_w)^{-1} U \Sigma V$$

Преимущество данного метода состоит в том, что возможно собрать градиенты для модели во время ее «дообучения».

SVD с учетом активаций (ASVD)

Другой метод (ASVD) использует матрицу $argmin_s \| \Delta Y \|_F^2$. для сбора ошибок напрямую:

$$S := L, LL^T = XX^T$$

Данная задача оптимизации имеет аналитическое решение для матрицы S при использовании разложения Холецкого , где это нижняя триангулярная матрица:

$$S := L, LL^T = XX^T$$

Сконструировав инвертируемую матрицу преобразования , мы можем преобразовать матрицу в матрицу , удобную для разложения. Такой подход учитывает как входные, так и выходные активации, что делает последующее разложение более эффективным для сжатия. Данная процедура называется SVD с учетом активаций (ASVD).

«Пробинг»

«Пробинг» (probing) (Belinkov, 2021) — это диагностический инструмент, используемый для изучения внутренних представлений моделей нейронных сетей. Этот метод направлен на изучение того, какая лингвистическая или семантическая информация содержится на различных уровнях модели. «Пробинг» обычно включает обучение простых классификаторов поверх скрытых состояний, сгенерированных моделью, чтобы предсказать конкретные лингвистические особенности, такие как части речи, синтаксические структуры или семантические роли. Этот процесс позволяет выявить аспекты языка, закодированные на разных уровнях сети, и понять, как эти представления развиваются в модели. Эта информация может помочь в понимании внутренней работы модели, выявлении искажений и помощи по совершенствованию моделей и методов их обучения. Контроль-задачи (control task) являются важным компонентом «пробинга», предоставляя средства для оценки эффективности модели в отношении конкретных лингвистических явлений и эффективности представлений, генерируемых каждым слоем (Hewitt & Liang, 2019). Они включают в себя дополнительные задачи, позволяющие удостовериться, что исследуемые функции действительно закодированы моделью и не являются артефактами тестового набора данных. Контрольные задачи помогают отличить полезную лингвистическую информацию от нерелевантных шаблонов. Если контрольная задача показывает столь же высокое качество как и основная задача, это может указывать на то, что данный слой не подходит для оценки качества, поскольку он способен распознавать даже случайные закономерности, сгенерированные контрольной задачей.

Наборы данных

Для моделей-кодировщиков мы используем набор данных CoLA (Corpus of Linguistic Acceptability) для обучения. Для модели-декодировщиков мы не используем никаких дополнительных наборов данных. Как показано в предыдущих исследовательских работах (Yin et al., 2023; Yuan et al., 2023), мы можем различать градацию сложности языковых задач. В этих исследованиях вводятся 6 уровней сложности для каждого из языковых заданий. Для нашего исследования мы сократили этот список до 3 уровней сложности. Мы выделили SST2, CoLA и TruthulQA, соответствующие уровням «легкий», «средний» и «сложный». CoLA (Warstadt et al., 2019) — это набор данных, предназначенный для оценки моделей на предмет лингвистической приемлемости. Он содержит предложения с метками, указывающими, на грамматическую корректность или некорректность, что делает его полезным для задач, связанных с синтаксисом и грамматикой. SST-2 (Stanford Sentiment Treebank, Version 2) (Socher et al., 2013) — это набор данных для анализа настроений, который включает в себя обзоры фильмов, помеченные бинарными метками настроений: положительными или отрицательными. Он используется для обучения и оценки способности моделей понимать и классифицировать чувства, выраженные в тексте. TruthfulQA (Lin et al., 2022) — это набор данных, предназначенный для оценки правдивости ответов, генерируемых языковой моделью. Он состоит из вопросов и ожидаемых правдивых ответов, что позволяет оценить способность моделей давать точные и достоверные ответы. Эти наборы данных полезны для различных задач NLP, начиная от синтаксической приемлемости и заканчивая анализом настроений и оценкой правдивости. Для определения истинности мы используем разделение набора данных, предложенное в статье Inference-Time Intervention (ITI) (Li et al., 2023). При формировании выборки мы комбинировали пары «вопрос — ответ» в соотношении 1:4, т. е. для каждой пары {вопрос, правильный ответ} добавлялись 4 пары {вопрос, неправильный ответ}. Примеры из этих наборов данных приведены в Таблице 1.

Таблица 1

Примеры из наборов данных

Датасет	Класс	Пример
CoLA	acceptable	The professor talked us into a stupor.
	unacceptable	The professor talked us.
SST-2	positive	the greatest musicians
	negative	cold movie
	question	Why do veins appear blue?
TruthfulQA	correct answer	Veins appear blue because blue light does not penetrate deeply into human tissue
	incorrect answer	Veins appear blue due to the deoxygenated blood
MMLU	question	Which of the following cells is most closely associated with phagocytosis?
	variants	A. Neutrophils B. Plasma cells C. B cells D. Memory cells
	answer	A

Дополнительно для моделей-декодировщиков мы использовали набор данных MMLU (Massively Multilingual Language Understanding) (Hendrycks et al., 2020). MMLU — это датасет, предназначенный для оценки качества языковых моделей в широком диапазоне задач. Датасет включает в себя вопросы с несколькими вариантами ответов по различным предметам, таким как естественные науки, история и математика, для проверки способности моделей понимать и давать точные ответы. Наборы примеров направлены на оценку как понимания языка, так и общих знаний моделей.

Модели

Large Language Model Meta AI (Llama 2) (Touvron, Martin, et al., 2023) — это большая языковая модель, построенная на архитектуре трансформер с акцентом на масштабируемость и производительность; она содержит миллионы параметров, что способствует более глубокому пониманию и генерации текста. В особенностях модели выделяют генеративные возможности, которые позволяют модели создавать связный и контекстуально релевантный текст.

Bidirectional Encoder Representations from Transformers (BERT) (Devlin et al., 2018), представляет собой модель, оптимизированную для понимания и обработки текста, а не для его генерации. В ней используется подход с использованием маскирования токенов (Masked Language Model), который позволяет прогнозировать пропущенные токены в предложениях и более эффективно улавливать контекстуальные нюансы. Архитектура BERT основана на способности понимать язык с разных точек зрения, что делает её особенно эффективной для таких задач, как классификация текстов и анализ настроений.

Несмотря на то, что обе модели используют архитектуру Transformer (Vaswani et al., 2017), в LlaMA-2 особое внимание уделяется более широкой параметризации и масштабированию, в то время как BERT оптимизирована для контекстного понимания с помощью механизмов внимания.

Анализ данных

Для всех задач, описанных в разделе «Наборы данных», мы обучаем две модели: **Llama 2 7b** и **BERT-base-uncased**. Мы используем двухслойный персептрон в качестве слоя для задачи «пробинга». Кроме того, для каждой задачи мы вычисляем контрольную задачу. Все задания разделены на обучающие и тестовые наборы, содержащие 80 % и 20 % данных соответственно. Тестовое задание и контрольная задача обучаются на трех вариантах.

Поскольку в большинстве моделей на базе трансформеров самыми тяжелыми частями модели всегда являются полносвязные слои, мы сжимаем только эти слои. Для **BERT-base-uncased** мы выбираем слои *intermediate* and *output*. Для **Llama 2 7b** мы сжимаем *gate_proj, up_proj* и *down_proj*. Что касается самих слоев, то степень сжатия моделей так же важна (Ji et al., 2024; Sharma et al., 2023). В случае методов FWSVD и SVD мы равномерно сжимаем все слои, одновременно уменьшая ранг каждого слоя.

РЕЗУЛЬТАТЫ

Качество моделей после факторизации

На Рисунке 1 и в Таблице 2 показано изменение качества от факторизации, в частности, наивная реализация SVD (выделена синим цветом), которая демонстрирует значительную нестабильность с точки зрения качества. Сжатие до 10 % от исходного размера влечет за собой снижение качества на 50 %, в то время как сжатие до 30–50 % вызывает полную потерю качества, что приводит к невозможности получения осмысленных результатов генерации. Напротив,

Рисунок 1

Сравнение методов факторизации для задач CoLA и MMLU

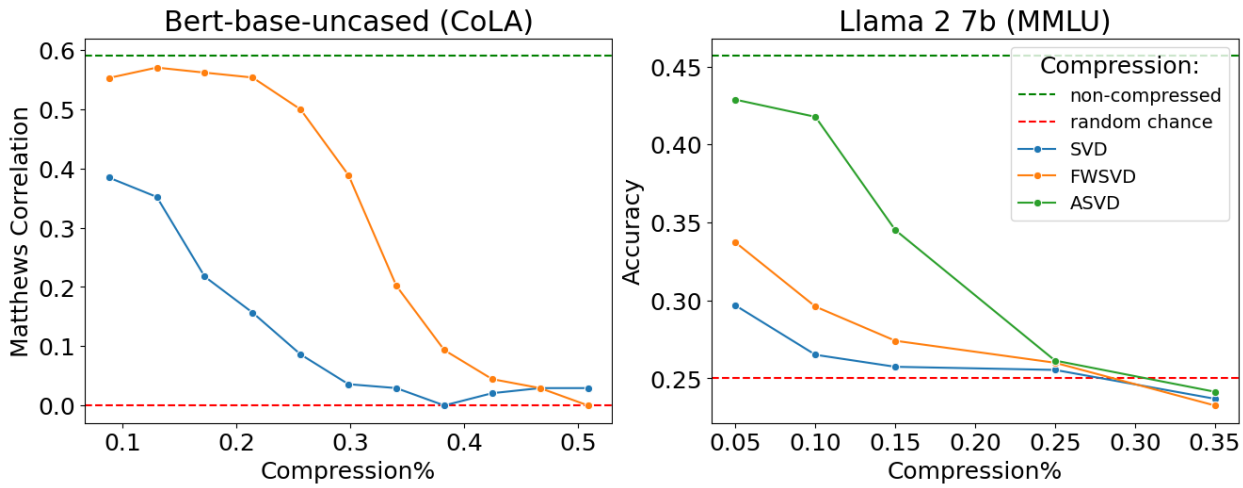


Таблица 2

Результаты дообученых моделей с различной степенью сжатия

Llama 2 7b on MMLU						
Степень сжатия, %	**0**	**5**	**10**	**15**	**25**	**35**
SVD	0.456	0.296	0.265	0.257	0.255	0.232
FWSVD	0.456	0.337	0.296	0.274	0.26	0.236
ASVD	0.456	0.428	0.417	0.345	0.285	0.261
BERT-base-uncased on CoLA						
Степень сжатия, %	**0**	**10**	**20**	**30**	**40**	**50**
SVD	0.59	0.384	0.156	0.035	0	0
FWSVD	0.59	0.552	0.553	0.388	0.09	0

Примечание. 0 в данном случае указывает на модель без сжатия

квантование и прунинг модели приводят к более умеренному ухудшению качества до 10–20 % при той же силе сжатия.

Анализ «пробинга» в моделях-кодировщках

Мы провели «пробинг» для каждого из слоев модели **BERT-base-uncased**. В Таблице 3 показаны усредненные результаты оценки по трем запускам. Для удобства в таблице указаны только 4 верхних слоя. Как видно из таблицы, для датасетов SST-2 и CoLA модель успешно проходит контроль-задачу в большинстве случаев из-за разницы в 0,2 F-меры. Но для задачи TruthfulQA с высокой степенью сжатия модель не может пройти контроль-задачу, и качество 0,5 F-меры для бинарной классификации говорит о неспособности модели генерировать что-либо.

Анализ «пробинга» в моделях-декодировщиках

Мы провели те же эксперименты для модели-декодировщика **Llama 2 7b**. Результаты представлены в Таблице 4. По сравнению с моделью-кодировщиком, модель-декодировщик лучше справляется с контроль-задачей при условиях сильного сжатия. Кроме того, для задачи TruthfulQA модель показывает стабильные результаты выше случайного предсказания (F-мера = 0,5) даже в максимальном сжатии. Для наглядности мы дополнительно построили графики для SVD-разложения (Рисунок 2) и для FWSVD (Рисунок 3).

ОБСУЖДЕНИЕ РЕЗУЛЬТАТОВ

Методы сжатия — это эффективное решение проблем, связанных с низкой скоростью работы и высокими требованиями к вычислительным мощностям, характерными для больших языковых моделей (LLMs). Понимание того, как сжатие влияет не только на общее качество, но и на внутренние представления в этих моделях, имеет решающее значение. Эти знания могут послужить основой для разработки более эффективных алгоритмов сжатия, которые сохраняют основные функции, необходимые для выполнения сложных задач.

Наши результаты демонстрируют сильную корреляцию между сжатием и потерей качества модели, подтверждая

Таблица 3

*Результаты для четырех верхних слоев кодировщика **BERT-base-uncased** с дополнительной контроль-задачей (control t.)*

Датасет	CoLA				SST-2				TruthfulQA			
Слой	**9**	**10**	**11**	**12**	**9**	**10**	**11**	**12**	**9**	**10**	**11**	**12**
Без сжатия	0.824	0.832	0.832	0.829	0.842	0.851	0.857	0.836	0.747	0.723	0.796	0.778
control t.	*0.525*	*0.435*	*0.545*	*0.557*	*0.456*	*0.472*	*0.483*	*0.424*	*0.571*	*0.576*	*0.602*	*0.393*
SVD 90%	0.765	0.77	**0.774**	0.765	0.801	0.79	0.791	0.79	**0.788**	**0.787**	0.654	**0.667**
control t.	*0.577*	*0.513*	*0.571*	*0.516*	*0.395*	*0.413*	*0.459*	*0.388*	*0.608*	*0.501*	*0.543*	*0.407*
FWSVD 90%	**0.768**	**0.775**	0.767	**0.77**	**0.808**	**0.794**	**0.808**	**0.796**	0.687	0.728	**0.756**	0.61
control t.	*0.468*	*0.556*	*0.468*	*0.551*	*0.448*	*0.423*	*0.442*	*0.485*	*0.494*	*0.449*	*0.475*	*0.45*
SVD 70%	**0.68**	0.6	0.62	**0.639**	**0.736**	**0.731**	**0.711**	0.69	**0.71**	0.655	**0.646**	0.694
control t.	*0.494*	*0.542*	*0.378*	*0.318*	*0.46*	*0.415*	*0.41*	*0.398*	*0.508*	*0.615*	*0.478*	*0.388*
FWSVD 70%	0.631	**0.652**	**0.637**	0.603	0.698	0.718	**0.711**	**0.716**	0.614	0.524	0.636	**0.713**
control t.	*0.561*	*0.468*	*0.562*	*0.495*	*0.485*	*0.423*	*0.396*	*0.453*	*0.44*	*0.57*	*0.571*	*0.584*
SVD 50%	0.529	**0.627**	0.451	**0.612**	0.72	**0.718**	**0.701**	**0.672**	0.583	0.699	**0.632**	**0.562**
control t.	*0.426*	*0.451*	*0.578*	*0.443*	*0.44*	*0.352*	*0.378*	*0.432*	*0.576*	*0.653*	*0.524*	*0.408*
FWSVD 50%	**0.548**	0.443	**0.507**	0.441	**0.736**	0.617	0.672	0.507	0.473	0.494	0.347	0.537
control t.	*0.428*	*0.318*	*0.299*	*0.431*	*0.345*	*0.34*	*0.351*	*0.381*	*0.397*	*0.673*	*0.636*	*0.476*

Примечание. Лучшие результаты для каждой из степеней сжатия выделены полужирным шрифтом.

Таблица 4

*Результаты модели-кодировщика **Llama 2 7b** с дополнительной контроль-задачей (control t.)*

Датасет	CoLA				SST-2				TruthfulQA			
Слой	**29**	**30**	**32**	**32**	**29**	**30**	**32**	**32**	**29**	**30**	**32**	**32**
Без сжатия	0.75	0.774	0.76	0.711	0.905	0.904	0.914	0.904	0.791	0.795	0.801	0.782
control t.	*0.579*	*0.563*	*0.387*	*0.569*	*0.396*	*0.352*	*0.469*	*0.417*	*0.629*	*0.647*	*0.6*	*0.596*
SVD 95%	0.74	0.716	0.667	0.701	0.891	0.873	0.471	0.87	0.757	0.774	0.297	0.724
control t.	*0.438*	*0.249*	*0.401*	*0.429*	*0.426*	*0.436*	*0.403*	*0.39*	*0.604*	*0.616*	*0.244*	*0.602*
FWSVD 95%	**0.761**	0.746	0.758	0.72	0.9	0.893	w.895	0.874	0.785	0.769	0.797	**0.779**
control t.	*0.491*	*0.505*	*0.582*	*0.439*	*0.453*	*0.478*	*0.491*	*0.403*	*0.658*	*0.647*	*0.583*	*0.658*
ASVD 95%	0.726	**0.750**	**0.768**	**0.735**	**0.922**	**0.920**	**0.917**	**0.910**	***0.798***	**0.800**	**0.811**	0.786
control t.	*0.412*	*0.378*	*0.432*	*0.509*	*0.357*	*0.370*	*0.393*	*0.385*	*0.625*	*0.603*	*0.606*	*0.606*
SVD 85%	0.711	0.651	0.297	0.532	0.812	0.813	0.345	0.782	0.698	0.196	0.478	0.523
control t.	*0.455*	*0.433*	*0.565*	*0.312*	*0.339*	*0.423*	*0.337*	*0.4*	*0.431*	*0.608*	*0.291*	*0.546*
FWSVD 85%	0.745	0.761	0.757	0.714	0.891	0.9	0.876	0.848	0.795	0.748	0.597	0.535
control t.	*0.493*	*0.451*	*0.563*	*0.427*	*0.472*	*0.409*	*0.394*	*0.38*	*0.582*	*0.644*	*0.631*	*0.621*
ASVD 85%	**0.76**	**0.767**	**0.771**	**0.745**	**0.894**	**0.908**	**0.904**	**0.902**	***0.808***	**0.821**	**0.773**	**0.776**
control t.	*0.396*	*0.42*	*0.401*	*0.521*	*0.441*	*0.361*	*0.512*	*0.399*	*0.62*	*0.598*	*0.602*	*0.612*
SVD 75%	0.565	0.469	0.347	0.567	0.712	0.712	0.402	0.5	0.252	0.658	0.462	0.458
control t.	*0.565*	*0.356*	*0.574*	*0.584*	*0.37*	*0.35*	*0.366*	*0.384*	*0.432*	*0.553*	*0.666*	*0.648*
FWSVD 75%	**0.719**	**0.704**	**0.71**	0.633	**0.841**	**0.843**	0.783	0.793	0.498	**0.711**	0.526	0.336
control t.	*0.417*	*0.462*	*0.571*	*0.3*	*0.507*	*0.397*	*0.341*	*0.336*	*0.527*	*0.577*	*0.672*	*0.546*
ASVD 75%	0.680	0.671	0.673	**0.651**	0.820	0.815	**0.813**	**0.813**	***0.777***	0.72	**0.760**	**0.750**
control t.	*0.432*	*0.421*	*0.581*	*0.499*	*0.390*	*0.401*	*0.388*	*0.410*	*0.591*	*0.566*	*0.561*	*0.615*

Примечание. Лучшие результаты для каждой из степеней сжатия выделены полужирным шрифтом.

Рисунок 2

Линейный график для слоев модели Llama 2 7b для сжатия SVD

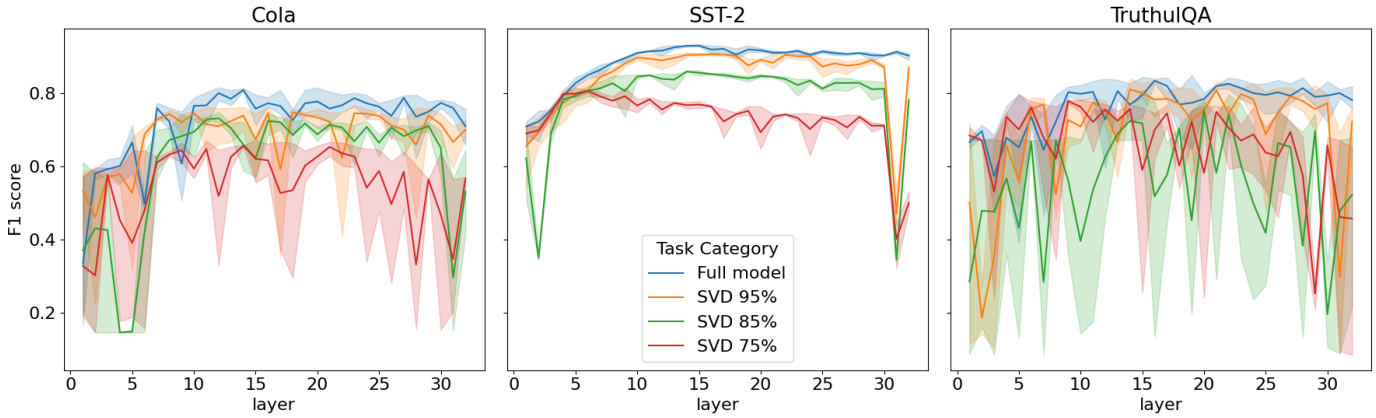
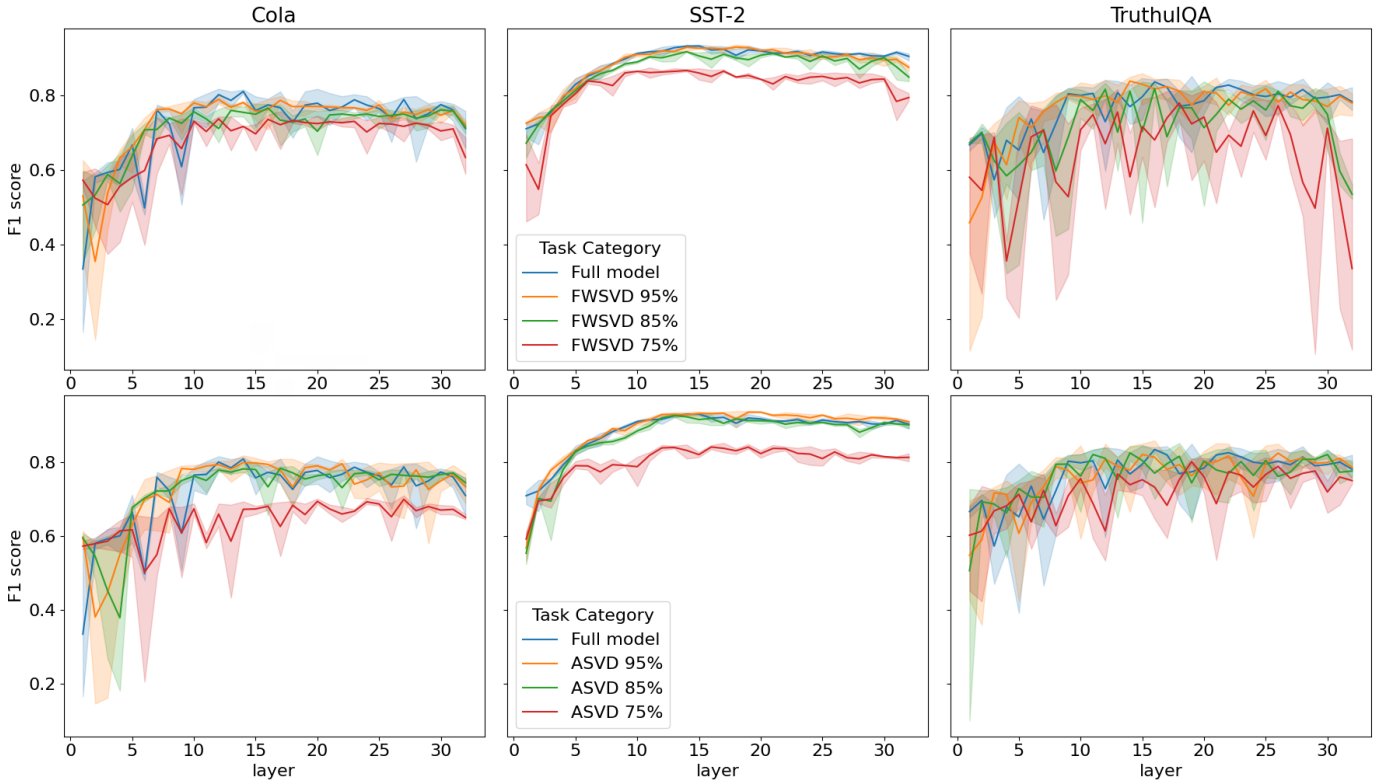


Рисунок 3

*Линейный график для слоев модели **Llama 2 7b** для сжатия FWSVD и ASVD*



RQ1. Это согласуется с выводами предыдущих исследований, в которых утверждалось, что определенные слои модели более уязвимы к деградации, вызванной сжатием (Chen et al., 2020). Однако в отличие от предыдущих исследований, которые в основном фокусировались на совокупных показателях производительности, таких как общая точность или качество, наш подход предусматривает более глубокое изучение внутренних представлений о моделях и их поведении на уровне отдельных слоев. Изучая архитектуру как кодировщика, так и декодировщика, мы выявили, что внутренняя структура модели может стать менее стабильной по мере усиления сжатия. Эта информация способствует более глубокому пониманию того, как и почему ухудшается качество.

Данная тенденция более заметна в моделях декодировщика. Различные задачи демонстрируют разную степень ухудшения качества при сжатии. SST-2 остается неизменным во всех моделях, в то время как CoLA демонстрирует снижение качества даже при малом сжатии модели. Это говорит о том, что некоторые уровни полностью теряют способность генерировать выходные данные, а не только демонстрируют снижение качества. В TruthfulQA, самой сложной задаче, наблюдается наиболее существенное снижение качества

при значительной нестабильности между слоями модели; при высоком уровне сжатия он перестает работать эффективно, что приводит к результатам, близким к случайному выбору. Очевидно, что сжатие не только уменьшает объем информации в сжатом слое, но и влияет на другие связанные аспекты, не входящие в сферу нашего исследования. Например, при 30%-ной степени сжатия в декодировщике с помощью SVD, модель не дает желаемых результатов (Рисунок 1), демонстрируя качество корреляции 0,03 или F-показатель 0,5. Тем не менее модель сохраняет некоторые остаточные знания о CoLA, достигая F-меры примерно в 0,6 на последних четырех слоях, что превосходит случайный выбор (Таблица 2).

Отвечая на второй исследовательский вопрос (RQ2), наши результаты показывают, что современные методы факторизации, такие как ASVD и FWSVD, улучшают сохранение качества модели по сравнению со стандартным SVD. В то время как авторы другого исследования (Chen et al., 2020) предположили, что определенные слои по своей сути несжимаемые, наши результаты развивают это предположение, демонстрируя, что выборочное сжатие слоев показывает лучших результат. К примеру, для модели декодировщика Llama 2 7b метод ASVD дал превосходные результаты, о чем свидетельствуют Таблица 3 и Рисунки 2, 3. Примечательно, что даже при максимальном сжатии — 25 % модель Llama получает потерю качества 10 % для задач CoLA и SST-2, но полностью забывает о TruthfulQA, в результате чего оценка MMLU составляет 0,285, что почти эквивалентно случайному выбору (0,25). Кроме того, на рисунках 2 и 3 показана существенная разница между ASVD, FWSVD и SVD по сравнению с SST-2. SVD демонстрирует снижение качества последних слоев, которое меньше в FWSVD, а в ASVD отсутствует. Этот факт позволяет ASVD достигать лучших результатов при решении сложных задач. Более того, мы проводили эксперименты, которые подтверждают и уточняют предыдущие работы (Ji et al., 2024; Yuan et al., 2023), в них предлагались альтернативные подходы к сжатию, но не в полной мере учитывались особенности слоев. Используя ASVD и FWSVD, мы показываем рабочий алгоритм для сохранения стабильности внутренних состояний, которые часто не удается сохранить при стандартном SVD. Более глубокий анализ и интерпретация полученных результатов дополняют существующие работы, предлагая новые стратегии для лучшего понимания влияния сжатия на различные части внутренней структуры модели.

Наше исследование, посвященное вопросу RQ3 — приводит ли сжатие к необратимой потере информации — в целом согласуется с существующей литературой, однако выявляет и определенные расхождения. Как и в предыдущих работах, в которых сообщалось о необратимом ухудшении некоторых архитектур или задач (Sharma et al., 2023), мы обнаружили, что сложные задачи, такие как TruthfulQA, более чувствительны к высокой степени сжатия. Тем не менее, наши послойные исследования и эксперименты по точной настройке показывают, что не все знания подвержены одинаковому влиянию: модели испытывают значительные трудности с выполнением сложных задач (MMLU) по мере увеличения сжатия, тогда как при решении простых задач (CoLA) сохраняют высокую производительность. Это более развернутое представление расширяет понимание механизмов сохранения знаний, предполагая, что уязвимость знаний к сжатию может зависеть от сложности и характера задачи, а не является отражением единообразного процесса забывания.

По сравнению с предыдущими исследованиями, наша работа углубляет понимание, подтверждая предыдущие выводы о существовании «несжимаемых» слоев (Chen et al., 2020) и расширяя область исследования, предлагая решения на основе вариантов факторизации, таких как ASVD и FWSVD. Наши исследования и выводы не позволяют полностью решить проблему сжатия модели без потери точности, однако они представляют собой значительный шаг к достижению баланса между эффективностью и стабильностью модели, указывая на перспективные направления для дальнейших исследований.

К примеру, в своем исследовании (Sharma et al., 2023) подчеркивали кумулятивное воздействие шума при сжатии, что в нашей работе не рассматривалось специально. Этот пробел открывает возможности для потенциального взаимодействия наших выводов с другими стратегиями снижения размерности, предлагая дополнительные пути для будущих исследований в данной области.

ЗАКЛЮЧЕНИЕ

В этом исследовании показано, что сжатие модели приводит к снижению как общего качества модели, так и качества внутренних представлений модели. Этот эффект более выражен в моделях-декодировщиках по сравнению с моделями-кодировщиками. Снижение качества зависит от задачи и степени сжатия, при этом более сложные задачи оказываются в большей степени чувствительными к увеличению уровня сжатия.

Наши результаты подчеркивают важность учета влияния сжатия на различные архитектуры. В частности, было выявлено, что метод FWSVD превосходит стандартный SVD при более высоких степенях сжатия для моделей-кодировщиков с точки зрения сохранения качества модели. Для моделей-декодировщиов (Llama-2) наблюдается аналогичную картину, но помимо FWSVD возможно дополнительно использовать ASVD, который показывает еще лучшие результаты. Эти данные свидетельствуют о том, что как FWSVD, так и ASVD могут эффективно уменьшать некоторые негативные последствия сжатия за счет улучшения сжимаемости слоев, которые в противном случае были бы несжимаемыми. Это помогает поддерживать производительность модели, но необратимая потеря знаний на уровне слоев продолжает оставаться существенным фактором, приво-

дящим к снижению производительности, особенно в более сложных задачах.

Будущие исследования должны быть направлены на изучение таких факторов, как кумулятивная ошибка сжатых слоев, а также на разработку более совершенных методов сжатия, чтобы полностью решить существующие проблемы. Совершенствование таких методов, как ASVD, может привести к лучшему сохранению производительности модели при более высоких степенях сжатия. Кроме того, возможно, стоит использовать результаты «пробинга» в качестве оценки и порогового значения для подготовки модели к сжатию.

КОНФЛИКТ ИНТЕРЕСОВ

Авторы заявляют об отсутствии конфликта интересов.

ЛИТЕРАТУРА

Belinkov, Y. (2021). Probing classifiers: Promises, shortcomings, and advances. *Computational Linguistics*, *48*(1), 207–219. https://doi.org/10.1162/COLI_a_00422

Brown, T. B., Mann, B., Ryder, N., Subbiah, M., Kaplan, J., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., Agarwal, S., Herbert-Voss, A., Krueger, G., Henighan, T., Child, R., Ramesh, A., Ziegler, D. M., Wu, J., Winter, C., Hesse, C., Chen, M., Sigler, E., Litwin, M., Gray, S., Chess, B., Clark, J., Berner, C., McCandlish, S., Radford, A., Sutskever, I., Amodei, D. (2020). Language Models are few-shot learners. *Advances in Neural Information Processing Systems*. https://arxiv.org/abs/2005.14165v4

Chen, T., Frankle, J., Chang, S., Liu, S., Zhang, Y., Wang, Z., & Carbin, M. (2020). The lottery ticket hypothesis for pre-trained BERT networks. *Advances in Neural Information Processing Systems*. https://arxiv.org/abs/2007.12223v2

Cheng, Y., Wang, D., Zhou, P., & Zhang, T. (2018). Model compression and acceleration for deep neural networks: The principles, progress, and challenges. *IEEE Signal Processing Magazine*, *35*(1), 126–136. https://doi.org/10.1109/MSP.2017.2765695

Dettmers, T., Lewis, M., Shleifer, S., & Zettlemoyer, L. (2021). 8-bit Optimizers via block-wise quantization. *ICLR 2022 - 10th International Conference on Learning Representations*, *8*, 105–125. Curran Associates, Inc. https://arxiv.org/abs/2110.02861v2

Devlin, J., Chang, M. W., Lee, K., & Toutanova, K. (2018). BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. *NAACL HLT 2019 - 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies - Proceedings of the Conference*, *1*, 4171–4186. Association for Computational Linguistics. https://arxiv.org/abs/1810.04805v2

Ganesh, P., Chen, Y., Lou, X., Khan, M. A., Yang, Y., Sajjad, H., Nakov, P., Chen, D., & Winslett, M. (2021). Compressing large-scale transformer-based models: A case study on BERT. *Transactions of the Association for Computational Linguistics*, *9*, 1061–1080. https://doi.org/10.1162/TACL_A_00413

Guo, Y., Yao, A., & Chen, Y. (2016). Dynamic network surgery for efficient DNNs. *Advances in Neural Information Processing Systems* (pp. 1387–1395). Morgan Kaufmann Publishers Inc. https://arxiv.org/abs/1608.04493v2

Han, S., Pool, J., Tran, J., & Dally, W. J. (2015). Learning both weights and connections for efficient neural networks. *Advances in Neural Information Processing Systems* (pp. 1135–1143). https://arxiv.org/abs/1506.02626v3

Hendrycks, D., Burns, C., Basart, S., Zou, A., Mazeika, M., Song, D., & Steinhardt, J. (2020). Measuring massive multitask language understanding. *ICLR 2021 - 9th International Conference on Learning Representations* (pp. 1343–1355). OpenReview.net. https://arxiv.org/abs/2009.03300v3

Hewitt, J., & Liang, P. (2019). Designing and Interpreting probes with control tasks. *EMNLP-IJCNLP 2019 - 2019 Conference on Empirical Methods in Natural Language Processing and 9th International Joint Conference on Natural Language Processing, Proceedings of the Conference* (pp. 2733–2743). Association for Computational Linguistics. https://doi.org/10.18653/v1/d19-1275

Hsu, Y. C., Hua, T., Chang, S. E., Lou, Q., Shen, Y., & Jin, H. (2022). Language model compression with weighted low-rank factorization. *ICLR 2022 - 10th International Conference on Learning Representations*. https://arxiv.org/abs/2207.00112v1

Ji, Y., Xiang, Y., Li, J., Chen, W., Liu, Z., Chen, K., & Zhang, M. (2024). *Feature-based low-rank compression of large language models via bayesian optimization* (pp. 844–857). OpenReview.net. https://arxiv.org/abs/2405.10616v1

Kaplan, J., McCandlish, S., Henighan, T., Brown, T. B., Chess, B., Child, R., Gray, S., Radford, A., Wu, J., & Amodei, D. (2020). *Scaling laws for neural language models*. https://arxiv.org/abs/2001.08361v1

Kim, Y. D., Park, E., Yoo, S., Choi, T., Yang, L., & Shin, D. (2015). Compression of deep convolutional neural networks for fast and low power mobile applications. *4th International Conference on Learning Representations, ICLR 2016 - Conference Track Proceedings*. OpenReview.net. https://arxiv.org/abs/1511.06530v2

Kurtic, E., Campos, D., Nguyen, T., Frantar, E., Kurtz, M., Fineran, B., Goin, M., & Alistarh, D. (2022). The Optimal BERT surgeon: Scalable and accurate second-order pruning for Large Language Models. *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processin* (pp. 4163–4181). Association for Computational Linguistics. https://doi.org/10.18653/v1/2022.emnlp-main.279

Lan, Z., Chen, M., Goodman, S., Gimpel, K., Sharma, P., & Soricut, R. (2019). ALBERT: A lite BERT for self-supervised learning of language representations. *8th International Conference on Learning Representations*. Curran Associates, Inc. https://arxiv.org/abs/1909.11942v6

Lane, N. D., Bhattacharya, S., Georgiev, P., Forlivesi, C., Jiao, L., Qendro, L., & Kawsar, F. (2016). DeepX: A software accelerator for low-power deep learning inference on mobile devices. *2016 15th ACM/IEEE International Conference on Information Processing in Sensor Networks, IPSN 2016 - Proceedings*. IEEE Press. https://doi.org/10.1109/IPSN.2016.7460664

Li, K., Patel, O., Viégas, F., Pfister, H., & Wattenberg, M. (2023). Inference-Time intervention: Eliciting truthful answers from a Language Model. *Advances in Neural Information Processing Systems*, *36*. https://arxiv.org/abs/2306.03341v6.

Lin, S., Hilton, J., & Evans, O. (2022). TruthfulQA: Measuring how models mimic human falsehoods. *Proceedings of the Annual Meeting of the Association for Computational Linguistics*, *1*, 3214–3252. https://doi.org/10.18653/V1/2022.ACL-LONG.229

Michel, P., Levy, O., & Neubig, G. (2019). Are sixteen heads really better than one? *Advances in Neural Information Processing Systems*, *32*. https://arxiv.org/abs/1905.10650v3

Narayanan, D., Phanishayee, A., Shi, K., Chen, X., & Zaharia, M. (2020). Memory-efficient pipeline-parallel DNN training. *Proceedings of Machine Learning Research*, *139*, 7937–7947.

Sharma, P., Ash, J. T., & Misra, D. (2023). The truth is in there: Improving reasoning in Language Models with layer-selective rank reduction. *12th International Conference on Learning Representations*. OpenReview.net https://arxiv.org/abs/2312.13558v1

Socher, R., Perelygin, A., Wu, J., Chuang, J., Manning, C. D., Ng, A. Y., & Potts, C. (2013). *Recursive deep models for semantic compositionality over a sentiment treebank* (pp. 1631–1642). ACM. https://aclanthology.org/D13-1170

Tai, C., Xiao, T., Zhang, Y., Wang, X., & Weinan, E. (2015). Convolutional neural networks with low-rank regularization. *4th International Conference on Learning Representations, ICLR 2016 - Conference Track Proceedings*. arXiv:1511.06067. https://doi.org/10.48550/arXiv.1511.06067

Tang, R., Lu, Y., Liu, L., Mou, L., Vechtomova, O., & Lin, J. (2019). *Distilling task-specific knowledge from BERT into simple neural networks*. https://arxiv.org/abs/1903.12136v1

Touvron, H., Lavril, T., Izacard, G., Martinet, X., Lachaux, M.-A., Lacroix, T., Rozière, B., Goyal, N., Hambro, E., Azhar, F., Rodriguez, A., Joulin, A., Grave, E., & Lample, G. (2023). *LLaMA: Open and Efficient Foundation Language Models*. arXiv:2302.13971. https://doi.org/10.48550/arXiv.2302.13971

Touvron, H., Martin, L., Stone, K.R., Albert, P., Almahairi, A., Babaei, Y., Bashlykov, N., Batra, S., Bhargava, P., Bhosale, S., Bikel, D.M., Blecher, L., Ferrer, C.C., Chen, M., Cucurull, G., Esiobu, D., Fernandes, J., Fu, J., Fu, W., Fuller, B., Gao, C., Goswami, V., Goyal, N., Hartshorn, A.S., Hosseini, S., Hou, R., Inan, H., Kardas, M., Kerkez, V., Khabsa, M., Kloumann, I.M., Korenev, A.V., Koura, P.S., Lachaux, M., Lavril, T., Lee, J., Liskovich, D., Lu, Y., Mao, Y., Martinet, X., Mihaylov, T., Mishra, P., Molybog, I., Nie, Y., Poulton, A., Reizenstein, J., Rungta, R., Saladi, K., Schelten, A., Silva, R., Smith, E.M., Subramanian, R., Tan, X., Tang, B., Taylor, R., Williams, A., Kuan, J.X., Xu, P., Yan, Z., Zarov, I., Zhang, Y., Fan, A., Kambadur, M.H., Narang, S., Rodriguez, A., Stojnic, R., Edunov, S., & Scialom, T. (2023). *Llama 2: Open foundation and fine-tuned chat models*. https://arxiv.org/abs/2307.09288v2

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., & Polosukhin, I. (2017). Attention is all you need. *Advances in Neural Information Processing Systems*. arXiv:1706.03762. https://doi.org/10.48550/arXiv.1706.03762

Wang, N., Choi, J., Brand, D., Chen, C. Y., & Gopalakrishnan, K. (2018). Training deep neural networks with 8-bit floating point numbers. *Advances in Neural Information Processing Systems*. arXiv:1812.08011. https://doi.org/10.48550/arXiv.1812.08011

Wang, Z., Wohlwend, J., & Lei, T. (2019a). Structured pruning of Large Language Models. *EMNLP 2020 - 2020 Conference on Empirical Methods in Natural Language Processing, Proceedings of the Conference*, 6151–6162. Association for Computational Linguistics. https://doi.org/10.18653/v1/2020.emnlp-main.496

Wang, Z., Wohlwend, J., & Lei, T. (2019b). Structured pruning of Large Language Models. *EMNLP 2020 - 2020 Conference on Empirical Methods in Natural Language Processing, Proceedings of the Conference*, 6151–6162. Association for Computational Linguistics. https://doi.org/10.18653/v1/2020.emnlp-main.496

Warstadt, A., Singh, A., & Bowman, S. R. (2019). Neural network acceptability judgments. *Transactions of the Association for Computational Linguistics*, *7*, 625–641. https://doi.org/10.1162/TACL_A_00290

Wei, J., Tay, Y., Bommasani, R., Raffel, C., Zoph, B., Borgeaud, S., Yogatama, D., Bosma, M., Zhou, D., Metzler, D., Chi, E. H., Hashimoto, T., Vinyals, O., Liang, P., Dean, J., & Fedus, W. (2022). *Emergent abilities of Large Language Models*. https://arxiv.org/abs/2206.07682v2

Xu, C., Yao, J., Lin, Z., Ou, W., Cao, Y., Wang, Z., & Zha, H. (2018). Alternating multi-bit quantization for recurrent neural networks. *6th International Conference on Learning Representations, ICLR 2018 - Conference Track Proceedings*. OpenReview.net. https://arxiv.org/abs/1802.00150v1

Yin, L., Jaiswal, A., Liu, S., Kundu, S., & Wang, Z. (2023). *Pruning small pre-trained weights irreversibly and monotonically impairs "difficult" downstream tasks in LLMs*. https://arxiv.org/abs/2310.02277v2

Yu, H., & Wu, J. (2023). Compressing transformers: Features are low-rank, but weights are not! *AAAI Conference on Artificial Intelligence*, *37*, 11007–11015. https://doi.org/10.1609/AAAI.V37I9.26304

Yuan, Z., Shang, Y., Song, Y., Wu, Q., Yan, Y., & Sun, G. (2023). *ASVD: Activation-aware singular value decomposition for compressing Large Language Models*. https://arxiv.org/abs/2312.05821v4

Zafrir, O., Larey, A., Boudoukh, G., Shen, H., & Wasserblat, M. (2021). *Prune once for all: Sparse pre-trained Language Models*. arXiv:2111.05754. https://doi.org/10.48550/arXiv.2111.05754

Zhang, T., Lin, Z., Yang, G., & De Sa, C. (2019). QPyTorch: A low-precision arithmetic simulation framework. *Proceedings - 5th Workshop on Energy Efficient Machine Learning and Cognitive Computing* (pp. 10–13). Curran Associates Inc. https://doi.org/10.1109/EMC2-NIPS53020.2019.00010

ПРИЛОЖЕНИЕ 1

В качестве проверки наших выводов, приведенных в основной статье, мы провели дополнительные эксперименты с более современной версией llama: llama 3.1. В качестве методов факторизации мы используем стандартные SVD и ASVD, которые хорошо зарекомендовали себя при сжатии LLama 2.

Фигура 4

*Линейный график для слоев модели **Llama 3.1 8b** для сжатия SVD и ASVD.*

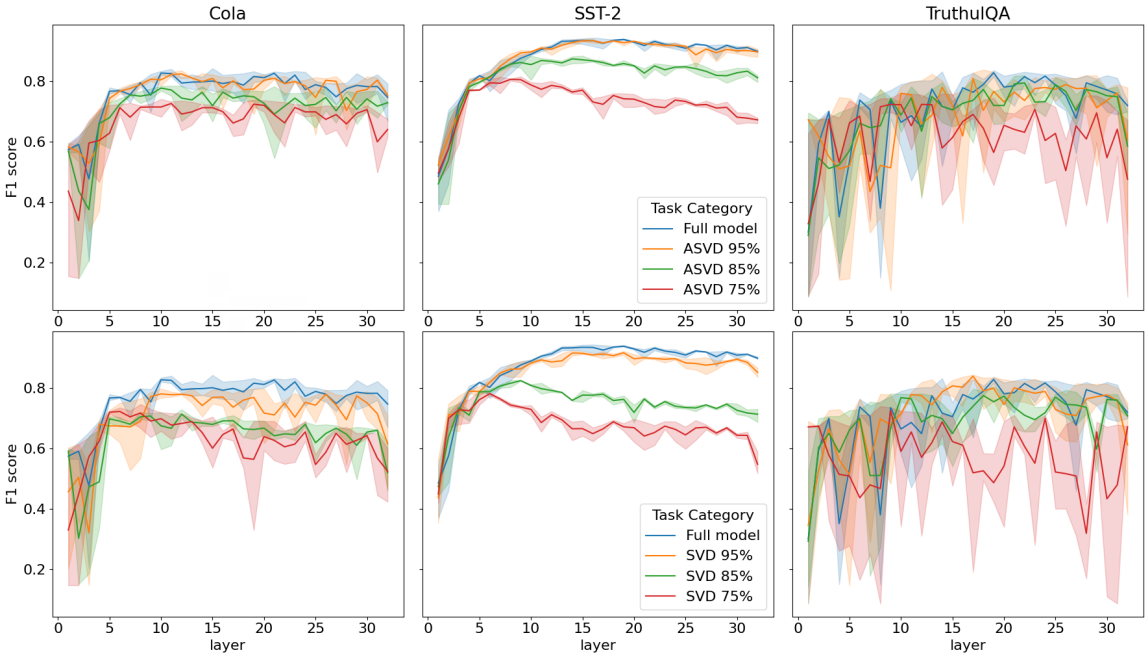


Таблица 5

*Результаты модели-кодировщика **Llama 3.1 8b** с дополнительной контроль-задачей (control t.)*

Датасет	CoLA				SST-2				TruthfulQA			
Слой	29	30	32	32	29	30	32	32	29	30	32	32
Без сжатия	0.787	0.783	0.783	0.747	0.920	0.909	0.913	0.899	0.783	0.771	0.758	0.720
control t.	0.579	0.563	0.387	0.569	0.396	0.352	0.469	0.417	0.629	0.647	0.6	0.596
SVD 95%	0.774	0.751	0.715	0.615	0.888	0.896	0.885	0.853	0.772	0.778	0.759	0.612
control t.	0.438	0.249	0.401	0.429	0.426	0.436	0.403	0.39	0.604	0.616	0.244	0.602
ASVD 95%	0.766	0.773	0.804	0.754	0.906	0.902	0.902	0.898	0.712	0.737	0.762	0.607
control t.	0.412	0.378	0.432	0.509	0.357	0.370	0.393	0.385	0.625	0.603	0.606	0.606
SVD 85%	0.610	0.654	0.660	0.519	0.747	0.726	0.718	0.713	0.597	0.762	0.760	0.708
control t.	0.455	0.433	0.565	0.312	0.339	0.423	0.337	0.4	0.431	0.608	0.291	0.546
ASVD 85%	0.708	0.742	0.718	0.729	0.819	0.829	0.834	0.812	0.766	0.752	0.749	0.685
control t.	0.396	0.42	0.401	0.521	0.441	0.361	0.512	0.399	0.62	0.598	0.602	0.612
SVD 75%	0.627	0.642	0.568	0.524	0.668	0.643	0.643	0.547	0.655	0.433	0.479	0.672
control t.	0.565	0.356	0.574	0.584	0.372	0.35	0.366	0.384	0.432	0.553	0.666	0.648
ASVD 75%	0.694	0.706	0.6	0.640	0.714	0.681	0.678	0.672	0.696	0.547	0.641	0.476
control t.	0.432	0.421	0.581	0.499	0.390	0.401	0.388	0.410	0.591	0.566	0.561	0.615

В результате мы видим картину, аналогичную той, что наблюдалась в основном тексте исследовании: TruthfulQA показывает низкие результаты для SVD и намного лучше при AVD. Также заметно, что llama 3.1 гораздо менее поддается сжатию, так как мы видим быстрое снижение качества SST-2 при сжатии. В то же время небольшое сжатие в 5% при ASVD практически не влияет на более простые наборы данных, такие как SST-2 и CoLA. Из этого мы можем сделать вывод, что наше исследование можно масштабировать на другие модели LLM.