ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ
ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ
«ВЫСШАЯ ШКОЛА ЭКОНОМИКИ»

Факультет информатики, математики и компьютерных наук

Татаринов Максим Дмитриевич

ПРИМЕНЕНИЕ МЕТОДОВ ОПТИМИЗАЦИИ НУЛЕВОГО ПОРЯДКА ДЛЯ ТОНКОЙ НАСТРОЙКИ МУЛЬТИМОДАЛЬНЫХ БОЛЬШИХ ЯЗЫКОВЫХ МОДЕЛЕЙ

Курсовая работа

по направлению подготовки *01.04.02 Прикладная математика и информатика*

образовательная программа

«Интеллектуальный анализ данных»

Руководитель

канд. комп. н.

А. В. Демидовский

Нижний Новгород 2025

Оглавление

Введение 3
Глава 1. Теория 6
1.1 Мультимодальные большие языковые модели 6
1.2 Тонкая настройка языковых моделей и её методы 11
1.2.1 Оптимизация нулевого порядка (ZO) 12
1.2.2 MeZO 13
Глава 2. Постановка задачи 16
2.1 Модель 18
2.2 Датасеты 18
Глава 3. Вычислительные эксперименты 21
3.1 Подготовка промптов 22
3.2 Качество дообучения 24
3.3 Использование памяти 26
3.4 Сходимость и стабильность обучения 27
Заключение 31
Список литературы 33
Приложение А 37

Введение

В последние годы наблюдается стремительное развитие крупных языковых моделей (LLM), обладающих способностью понимать и генерировать текст на уровне, приближенном к человеческому. Первоначально их использовали преимущественно для обработки текстовой информации, однако недавно в фокус исследований и приложений оказались мультимодальные модели (MLLM, или MM-LLM), способные работать не только с текстом, но и с изображениями, видео, аудио и другими типами данных [1, 2].

Базовые предобученные языковые модели могут не всегда полностью соответствовать требованиям конкретного случая использования. Адаптация предобученных моделей к специализированным задачам традиционно осуществляется посредством дообучения, или тонкой настройки (fine-tuning). Дообучение с использованием методов первого порядка, таких как стохастический градиентный спуск (SGD) [3] или Adam [4] используется чаще всего. Однако с увеличением масштабов моделей проблема потребления оперативной памяти становится всё более острой. Так, [5] показывают, что дообучение модели OPT-13B с использованием полного набора параметров или с помощью методов, ориентированных на параметрическую эффективность (PEFT), требует соответственно в 12 и 6 больше памяти, чем режим работы модели в режиме инференса. Это объясняется необходимостью кэширования активаций во время прямого прохода, а также хранения градиентов и состояний оптимизатора при обратном распространении ошибки (backpropagation) [6]. Высокая ресурсоёмкость дообучения моделей, связанная с большим потреблением оперативной памяти, приводит к необходимости использования дорогостоящего оборудования для выполнения тонкой настройки.

В связи с вышеописанными ограничениями активно развиваются методы адаптации, основанные на подходе, не требующем обратного распространения. К таким методам относится оптимизация нулевого порядка (ZO). Несмотря на

то, что исследования методов нулевого порядка ведутся уже несколько десятилетий [7, 8], лишь в последнее время их начали применять для дообучения LLM, что позволило значительно снизить затраты вычислительной памяти. Так, Memory-Efficient Zeroth-Order Optimizer (MeZO), предложенный [5], демонстрирует использование памяти сопоставимое с режимом инференса. При этом, существует недостаток исследований, посвящённых применению методов нулевого порядка к мультимодальным языковым моделям и соответствующим им более сложным задачам.

Таким образом, **актуальность** данного исследования обусловлена высокой требовательностью к памяти традиционных методов дообучения мультимодальных LLM. Методы оптимизации нулевого порядка позволяют значительно снизить эти затраты, что открывает возможность их адаптации даже на более дешёвом железе.

Целью настоящей работы является применение оптимизации нулевого порядка для тонкой настройки мультимодальных больших языковых моделей, а также проведение сравнительного анализа с традиционными методами оптимизации.

Из поставленной цели были сформулированы следующие **задачи**:

1. Провести обзор современных мультимодальных больших языковых моделей и рассмотреть основные архитектурные решения.
2. Изучить теоретические основы оптимизации нулевого порядка и исследовать особенности их применения для дообучения нейронных сетей.
3. Реализовать метод нулевого порядка для адаптации мультимодальной модели.
4. Провести экспериментальное сравнение эффективности методов оптимизации нулевого порядка и традиционных методов дообучения на ряде специализированных датасетов.
5. Проанализировать полученные результаты, выявив преимущества и недостатки подхода.

Объектом исследования является процесс дообучения предобученных мультимодальных больших языковых моделей для решения специализированных задач.

Предметом исследования являются реализация и применение методов оптимизации нулевого порядка для эффективной тонкой настройки мультимодальных моделей с минимальными затратами памяти.

Для решения поставленных задач будут использованы следующие **методы**:

1. Метод анализа и синтеза для создания теоретической базы для данного исследования на основе литературы,
2. Методы программирования для написания алгоритмов программы и обучения модели,
3. Методы нулевого порядка для оптимизации параметров модели без использования обратного распространения ошибки,
4. Методы глубокого обучения для дообучения мультимодальных языковых моделей на специализированных датасетах.

Новизна настоящей работы состоит в том, что в ней рассматривается применение методов оптимизации нулевого порядка к тонкой настройке мультимодальных больших языковых моделей по сравнению с традиционными методами.

Работа организована следующим образом: в первой главе рассматриваются теоретические аспекты мультимодальных больших языковых моделей, а также методы их тонкой настройки; во второй главе кратко описывается постановка задачи; в третьей главе приводятся результаты вычислительных экспериментов и их анализ. В заключении содержатся выводы о проделанной работе.

Глава 1. Теория

1.1. Мультимодальные большие языковые модели

В последние годы большие языковые модели (Large Language Models, LLM) стали одной из центральных тем в области искусственного интеллекта (ИИ). Их популярность резко возросла благодаря способности генерировать и понимать текст на уровне, близком к человеческому. Начало этому тренду положил выпуск ChatGPT в 2022 году, который привлек внимание как научного сообщества, так и широкой общественности. LLM, основанные на архитектуре Transformer, такие как серия GPT от OpenAI (начиная с GPT-1 в 2018 году) [9], Google Gemini/BARD, Anthropic CLAUDE и открытые модели, такие как LLaMA от Meta [1] или PaLM от Google [10], продемонстрировали свою универсальность в задачах обработки текста, включая генерацию текста, суммирование, машинный перевод и создание контента в рамках генеративного ИИ. Однако с развитием технологий и ростом объема данных стало очевидно, что одномодальные системы, работающие исключительно с текстом, не способны полностью удовлетворить потребности сложных реальных задач.

На этом фоне возникли мультимодальные большие языковые модели (Multimodal Large Language Models, MLLM) [11], которые расширяют возможности LLM, интегрируя обработку данных различных типов, таких как текст, изображения, видео и аудио. MLLM представляют собой значительный шаг вперед в развитии ИИ, поскольку позволяют создавать более полное и точное представление информации за счет синергии между различными модальностями. Это открывает новые горизонты для приложений, таких как генерация описаний изображений (image captioning), создание видео по текстовым запросам, обработка аудио с учетом контекста и многие другие. В отличие от традиционных одномодальных систем, MLLM способны не только анализировать данные отдельно по каждой модальности, но и находить сложные взаимосвязи между ними, что де-

лает их незаменимыми для задач, требующих комплексного мультимодального анализа.

Исторический контекст и развитие MLLM

История развития MLLM берет начало с попыток объединить достижения в области компьютерного зрения и обработки естественного языка (Natural Language Processing, NLP). Ранние подходы к мультимодальной обработке информации были вдохновлены методами машинного перевода. В таких системах изображение сначала преобразовывалось в промежуточное представление с помощью сверточной нейронной сети (CNN), а затем это представление передавалось в рекуррентную нейронную сеть (RNN) для генерации текстового описания [12]. Однако такие методы были вычислительно затратными и часто давали неточные результаты. Со временем исследователи начали искать более эффективные способы интеграции модальностей.

Одним из ключевых прорывов стало появление модели Contrastive Language-Image Pre-training (CLIP) [13], представленной в 2021 году. CLIP, обученная на 400 миллионах пар изображений и текстов, продемонстрировала способность к zero-shot классификации, то есть могла выполнять задачи без дополнительного обучения на конкретных данных. Этот подход стал основой для многих современных MLLM, поскольку позволил эффективно сопоставлять визуальные и текстовые представления. BLIP-2 [14], представленный в 2023 году, продемонстрировал подход к объединению визуальных и текстовых представлений через двухэтапное обучение. В нём Querying Transformer (Q-Former) извлекает ключевые визуальные особенности из заранее обученного и неизменяемого кодировщика изображений, а затем адаптирует их для взаимодействия с замороженной языковой моделью, обеспечивая тем самым более глубокое и точное сопоставление между изображениями и текстом.

Примеры современных MLLM

Среди современных MLLM можно выделить несколько ключевых моделей, которые демонстрируют разнообразие подходов к интеграции модальностей. LLaVA (Large Language and Vision Assistant) [1], представленная в 2023 году, сочетает визуальный энкодер CLIP с языковой моделью Vicuna для общего понимания связи между изображениями и текстом. Эта модель была дообучена на наборе данных MS-COCO, что позволило ей эффективно обрабатывать задачи, связанные с визуально-языковым взаимодействием. LLaVA 1.5 добавила улучшения в виде более мощного энкодера CLIP-ViT-L и проекционного слоя на основе многослойного перцептрона (MLP), что повысило ее точность.

MiniGPT4 [15], разработанная в 2023 году исследователями из KAUST, представляет собой открытую альтернативу GPT-4 от OpenAI. Эта модель использует замороженные визуальные компоненты (Q-Former и ViT) и языковую модель Vicuna, соединяя их с помощью одного проекционного слоя.

mPLUG-OWL[16], представленная в 2023 году исследователями из Alibaba DAMO Academy, предлагает новый подход к выравниванию изображений и текста, замораживая визуальную модель только на втором этапе обучения. Используя LLaMA-7B как языковую модель и ViT-L/14 как визуальный энкодер, mPLUG-OWL демонстрирует высокую производительность в задачах визуального понимания, хотя также сталкивается с проблемами галлюцинаций.

Семейство открытых мультимодальных моделей LLaVA-OneVision [2], представленное в конце 2024 года, справляется с обработкой одиночных изображений, мультизвуковых последовательностей и видеосигналов Архитектура модели объединяет в себе следующие компоненты: языковую модель Qwen-2, визуальный энкодер SigLIP, обеспечивающий извлечение признаков изображений и видео, и двухслойный MLP-проектор для проекции визуальных признаков в языковое пространство эмбеддингов. Экспериментальные результаты демонстрируют, что подход обеспечивает переносимость знаний между различными сценари-

ями, что подтверждается успешным применением модели в рамках задач видеоанализа и кросс-модального обучения.

Устройство мультимодальных языковых моделей

Рис. 1. Архитектура MLLM

Популярная схема устройства MLLM включает три основных блока [1, 2]:

1. Визуальный энкодер. Для анализа изображений применяется визуальный энкодер, который преобразует входное изображение (Xv) в набор визуальных признаков (Zv). Такие признаки могут быть извлечены как из промежуточных слоёв трансформера, так и из его финальных слоёв, что обеспечивает учёт пространственного расположения элементов и глубину представления.

2. Проектор. Специальная проекционная сеть играет ключевую роль в объединении текстовых и визуальных представлений. Обычно для этой цели используется лёгкий модуль (например, двухслойный MLP или простая линейная трансформация), который отображает визуальные признаки (Zv) в то же пространство эмбеддингов, что и текстовые представления языковой модели. Итогом является

формирование последовательности «визуальных токенов» (Hv), сопоставимых с текстовыми.

3. Языковая модель. В основе системы лежит языковая модель, предобученная на огромном корпусе текстов, такие как Vicuna или Qwen-2. Она отвечает за генерацию ответов и формирование связного текста. Языковая модель принимает на вход как текстовые инструкции (Xq), так и визуальные токены (Hv), объединяя их для генерации ответа (Xa).

Обучение мультимодальных моделей является многоэтапным процессом, направленным на выравнивание представлений разных модальностей и адаптацию модели к конкретным задачам.

1. Первоначальным этапом является выравнивание представлений между модальностями. На этом этапе замораживаются веса как визуального энкодера, так и языковой модели, а обучается только проекционный слой. Это позволяет эффективно преобразовать визуальные данные в формат, совместимый с языковой моделью, поскольку визуальные токены приводятся в форму, максимально приближенную к распределению текстовых эмбеддингов.

2. После этого начинается основной этап обучения, в рамках которого как правило уже все компоненты модели (в том числе LLM, визуальный энкодер и проектор) подвергаются дообучению. На данном этапе задача заключается в тонкой настройке модели, чтобы она могла эффективно связывать информацию из изображения с текстовыми запросами. При этом обучающие примеры могут варьироваться от простых описаний изображений до более сложных многозадачных сценариев, что содействует появлению «новых» способностей в рамках мультимодального взаимодействия. Размер и сложность входных данных постепенно увеличиваются, что позволяет модели адаптироваться к обработке изображений более высокого разрешения и большему числу визуальных токенов без резкого скачка вычислительной нагрузки.

1.2. Тонкая настройка языковых моделей и её методы

Потенциал, предоставляемый заранее обученными LLM и MLLM, огромен. Однако эти базовые модели не всегда могут идеально соответствовать специфическим требованиям каждого конкретного случая использования. В некоторых случаях может требоваться более персонализированный вывод, а в других – отсутствует важный контекст или специализированные знания, которые не были получены на этапе первоначального обучения. Чтобы в полной мере использовать потенциал заранее обученных базовых моделей для конкретных случаев применения, можно прибегнуть к тонкой настройке модели (файн-тьюнинг, от англ. fine-tuning).

Тонкая настройка – это процесс изменения некоторых параметров заранее обученной базовой модели путем дальнейшего обучения на меньшем, специализированном для определенной задачи наборе данных с целью оптимизировать производительность модели для конкретного случая. Полная тонкая настройка предполагает корректировку всех параметров заранее обученной модели на специализированном наборе данных, чтобы модель лучше соответствовала специфическим требованиям целевого случая использования. Хотя такой подход обеспечивает всестороннюю настройку, он требует значительного объема обучающих данных. Преимущество полной тонкой настройки заключается в способности модели улавливать тонкости конкретной области, однако этот процесс может быть вычислительно затратным [5]. В связи с этим, в последние годы наблюдается рост интереса к методам тонкой настройки, которые требуют меньшего объема вычислительных ресурсов и памяти.

Методы параметрически-эффективной тонкой настройки (Parameter-Efficient Fine-Tuning, PEFT), например LoRA [17], обновляют лишь небольшую часть параметров модели. Хотя такие подходы позволяют дообучать большие языковые модели (LLM) при относительно низких затратах памяти и вычисли-

тельных ресурсов, в определённых случаях более предпочтительной оказывается полная тонкая настройка, позволяющая задействовать всю модель [18].

В условиях ограниченных вычислительных ресурсов важным становится использование методов полной тонкой настройки с низким потреблением памяти. Так, подход *Low-Memory Optimization (LOMO)* [19] отказывается от хранения градиентов при использовании стохастического градиентного спуска (SGD) [3, 20], однако градиенты вычисляются на каждом шаге обучения. Альтернативой является оптимизация нулевого порядка (Zeroth-Order Optimization, ZO), которая позволяет оценивать градиенты, используя только прямые проходы через модель. Такой подход требует объёма памяти, сопоставимого с режимом инференса.

В работе [5] предложен *Memory-Efficient Zeroth-Order Optimizer (MeZO)* для дообучения LLM исключительно на основе прямых проходов без обратного распространения ошибки. Достижение высоких результатов MeZO объясняется тем, что в предобученных глубоких нейронных сетях матрицы Гессе обладают небольшим локальным эффективным рангом [21—23]. Метод вычисляет оценку градиента всего за два прямых прохода, используя рандомизированное возмущение параметров, что существенно снижает задержки между вычислениями на GPU по сравнению с методами типа LOMO.

1.2.1. Оптимизация нулевого порядка (ZO)

Оптимизация нулевого порядка давно изучается в задачах выпуклой и сильно выпуклой оптимизации. В качестве классического оценивателя градиента используется метод одновременного стохастического приближения с возмущениями (Simultaneous Perturbation Stochastic Approximation, SPSA) [7]. Он позволяет заменить вычисление градиента в традиционном SGD на оценку, основанную исключительно на прямых проходах.

Рассмотрим размеченный датасет $\mathcal{D} = \{(x_i, y_i)\}_{i \in |\mathcal{D}|}$, мини-батч $\mathcal{B} \subset \mathcal{D}$, размера B и функцию потерь $\mathcal{L}(\theta; \mathcal{B})$ для модели с параметрами $\theta \in \mathbb{R}^d$. Тогда

оценка градиента методом SPSA определяется следующим образом:

$$\widehat{\nabla}\mathcal{L}(\theta;\mathcal{B}) = \frac{\mathcal{L}(\theta+\epsilon z;\mathcal{B}) - \mathcal{L}(\theta-\epsilon z;\mathcal{B})}{2\epsilon}\, z.$$

Здесь $z \in \mathbb{R}^d$ выбирается из нормального распределения, $z \sim \mathcal{N}(0, I)$, где I — единичная матрица, а ϵ представляет масштаб возмущения.

Метод SPSA требует всего двух прямых проходов через модель для оценки градиента, после чего полученная оценка используется для обновления параметров по схеме:

$$\theta_{t+1} = \theta_t - \eta\, \widehat{\nabla}\mathcal{L}(\theta;\mathcal{B}_t),$$

где $\mathcal{B}_t$ — мини-батч на шаге t, а η — скорость обучения.

1.2.2. MeZO

Алгоритм ZO-SGD в классической реализации требует вдвое больше памяти, чем при инференсе, из-за необходимости хранить вектор $z \in \mathbb{R}^d$. В работе [5] предложена память-эффективная версия этого алгоритма, названная MeZO. Имплементация представлена в виде алгоритма 1. На каждом шаге генерируется случайное начальное значение s, которое используется для сброса генератора случайных чисел и повторной выборки нужных элементов z при каждом из четырех применений в алгоритме. Такая реализация «на месте» позволяет MeZO сократить объем памяти до уровня, эквивалентного затратам на инференс.

Кроме того, авторами отмечается, что раздельное возмущение каждого параметра может быть слишком затратным по времени для больших моделей. В практической реализации предлагается возмущать целые весовые матрицы вместо отдельных скаляров, что ускоряет обучение, но увеличивает необходимый объем памяти на размер крупнейшей матрицы, например на размер матрицы словарных эмбеддингов для OPT-66B (0.86 ГБ).

Algorithm 1 MeZO

Require: параметры модели $\theta \in \mathbb{R}^d$, функция потерь $\mathcal{L} : \mathbb{R}^d \to \mathbb{R}$, количество шагов T, масштаб возмущения ϵ, размер батча B, скорость обучения η_t

for $t = 1, \ldots, T$ **do**
  Выбрать батч $\mathcal{B} \subset \mathcal{D}$ и случайный сид s
  $\theta \leftarrow$ ВозмущениеПараметров(θ, ϵ, s)
  $\ell_+ \leftarrow \mathcal{L}(\theta; \mathcal{B})$
  $\theta \leftarrow$ ВозмущениеПараметров$(\theta, -2\epsilon, s)$
  $\ell_- \leftarrow \mathcal{L}(\theta; \mathcal{B})$
  $\theta \leftarrow$ ВозмущениеПараметров(θ, ϵ, s) ▷ Возврат параметров к исходному состоянию
  projected_grad $\leftarrow (\ell_+ - \ell_-)/(2\epsilon)$
  Сбросить генератор случайных чисел с сидом s ▷ Для выборки z
  for all $\theta_i \in \theta$ **do**
    $z \sim \mathcal{N}(0, 1)$
    $\theta_i \leftarrow \theta_i - \eta_t \cdot$ projected_grad $\cdot\, z$
  end for
end for

Algorithm 2 ВозмущениеПараметров

Require: θ, ϵ, s

Сбросить генератор случайных чисел с сидом s ▷ Для выборки z
for all $\theta_i \in \theta$ **do**
  $z \sim \mathcal{N}(0, 1)$
  $\theta_i \leftarrow \theta_i + \epsilon \cdot z$ ▷ Изменение параметров на месте
end for
return θ

Кроме минимального использования памяти во время обучения, преимуществом MeZO является возможность сохранять чекпойнты, включающие толь-

ко сид s и оценку градиента. Такой чекпойнт позволяет восстановить состояние модели из изначальных параметров, но занимая значительно меньше места, чем традиционные чекпойнты, содержащие веса модели.

Недостатками MeZO согласно [5] является то, что для достижения хорошей производительности требуется большое количество итераций, а также грамотно составленный промпт, так как метод работает только в тех условиях, когда траектория оптимизации ведет себя достаточно благоприятсвующе.

Среди попыток улучшить MeZO можно выделить MeZO-SVRG [24], в котором происходит уменьшение дисперсии оценок градиента, однако MeZO-SVRG потребляет больше памяти, чем MeZO, из-за необходимости хранения копий параметров и оценок градиента, рассчитанных на полный батч.

Выводы

Таким образом, в настоящее время разрабатывается множество мультимодальных больших языковых моделей MLLM, которые состоят из визуального энкодера, проекционного слоя и языковой модели. Вначале при разработке таких моделей обучается проекционный слой, обеспечивающий выравнивание визуальных и текстовых представлений, а затем производится адаптация различных компонентов модели – от отдельно языковой модели до полной архитектуры с визуальным энкодером. Традиционные методы тонкой настройки, такие как Adam, требуют значительных затрат памяти из-за необходимости вычисления градиентов через обратное распространение ошибки. Алгоритмы нулевого порядка, такие как MeZO, демонстрируют потенциал значительного уменьшения использования памяти и могут быть применены для оптимизации MLLM.

Глава 2. Постановка задачи

В данной работе рассматривается задача тонкой настройки предобученных мультимодальных больших языковых моделей (MLLM) с использованием методов оптимизации нулевого порядка (ZO, см. раздел 1.2.1), что позволяет значительно сократить потребление памяти по сравнению с традиционными методами, такими как AdamW [4, 25].

Этап 1. Подготовка и формулировка задачи. Пусть имеется предобученная мультимодальная языковая модель M с параметрами $\theta \in \mathbb{R}^d$, способная обрабатывать как текстовую, так и визуальную информацию. В качестве обучающих данных используется выборка $D = \{(I_i, T_i, Y_i)\}_{i=1}^N$, которая охватывает несколько уровней сложности и типов задач, где:

- I_i – изображение, представляющее визуальную информацию;
- T_i – текстовый промпт, содержащий текстовую информацию;
- $Y_i = (y_{i1}, y_{i2}, \ldots, y_{iL_i'})$ – требуемый эталонный ответ (последовательность токенов).

Этап 2. Имплементация метода и дообучение. Данная часть состоит в адаптации существующего пайплайна обучения модели M для интеграции алгоритма нулевого порядка MeZO, что требует переписывания части обучающей логики, а также проведения сравнения с традиционными методами настройки. Так, в алгоритме MeZO для датасета

$$\mathcal{D} = \{(x_i, y_i)\}_{i=1}^{|\mathcal{D}|},$$

для мини-батча $\mathcal{B} \subset \mathcal{D}$ размера B и функции потерь $\mathcal{L}(\theta; \mathcal{B})$ для модели с параметрами $\theta \in \mathbb{R}^d$ оценка градиента производится следующим образом:

$$\widehat{\nabla}\mathcal{L}(\theta; \mathcal{B}) = \frac{\mathcal{L}(\theta + \epsilon z; \mathcal{B}) - \mathcal{L}(\theta - \epsilon z; \mathcal{B})}{2\epsilon} z,$$

где $\epsilon > 0$ – величина пертурбации, а $z \in \mathbb{R}^d$ – случайный вектор. Подробное описание метода приведено в алгоритме 1.

При тонкой настройке модели применяется метод принудительного обучения (teacher forcing). Функция потерь определяется как кросс-энтропия между логитами, генерируемыми языковой моделью, и истинным ответом. Более формально, если $Y_i = (y_{i1}, y_{i2}, \ldots, y_{iL'_i})$ – эталонная последовательность токенов, то функция потерь записывается как

$$L(\theta) = -\sum_{i=1}^{N} \sum_{j=1}^{L'_i} \log p\Big(y_{ij} \,\big|\, I_i,\, T_i,\, y_{i1}, \ldots, y_{i,j-1};\, \theta\Big).$$

При этом токены, относящиеся к первоначальному промпту, исключаются из суммирования.

На этапе инференса для формирования ответа используется жадное декодирование:

$$\hat{y}_{ij} = \arg\max_{k}\, p\Big(k \,\big|\, I_i,\, T_i,\, \hat{y}_{i1}, \ldots, \hat{y}_{i,j-1};\, \theta\Big).$$

Этап 3. Экспериментальное сравнение и оценка. Для оценки эффективности предлагаемого метода планируется проведение сравнительного анализа по следующим направлениям:

1. **Качество дообучения.** Оценка точности генерируемых ответов для каждого датасета при использовании разных конфигураций модели: дообучение только языковой модели (LM); дообучение в комбинации с проекционным слоем (LM+Projector); дообучение всей архитектуры (LM+Projector+Vision Encoder). В качестве эталонного сравнения используются результаты дообучения методом AdamW.
2. **Сходимость и стабильность обучения.** Анализ кривых функции потерь, выявление особенностей сходимости и устойчивости оптимизационного процесса.
3. **Потребление памяти.** Измерение пикового использования памяти для каждого подхода.

2.1. Модель

В качестве мультимодальной большой языковой модели для экспериментов используется LLaVA-OneVision [2], а именно llava-onevision-qwen2-0.5b-si (версия с 0.5B параметров) и llava-onevision-qwen2-7b-si (версия с 7B параметров). Версия Single-Image (SI) выбрана, так как набор датасетов, используемый в настоящей работе, сфокусирован на одиночных изображениях, и подразумевает, что модель предобучена на обработку одиночных изображений, а не на видео. Архитектура LLaVA-OneVision представлена на рисунке 2.

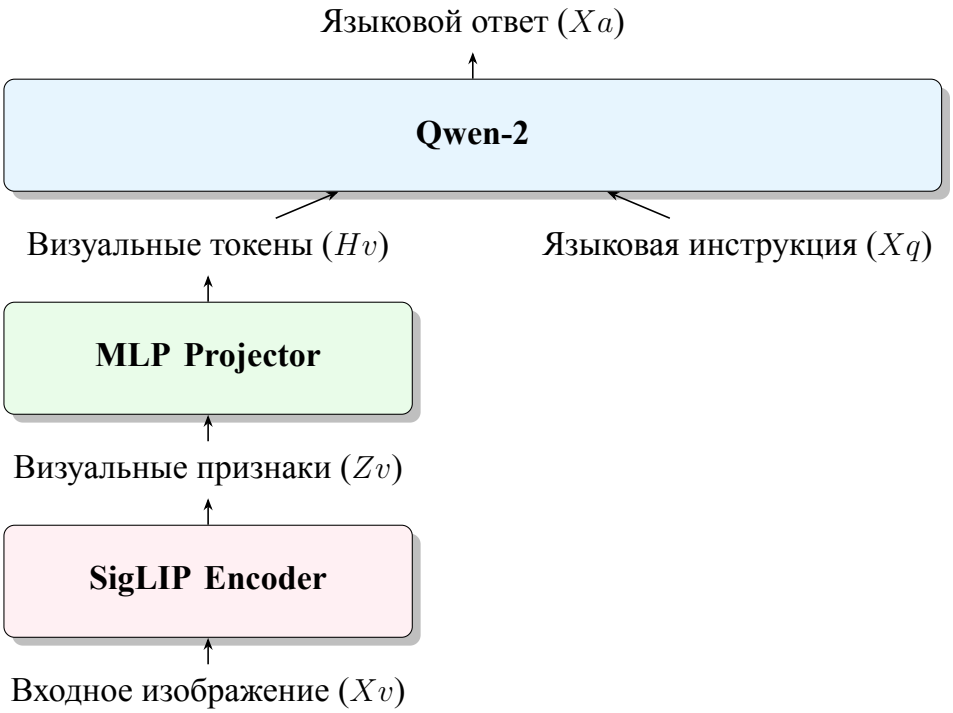


Рис. 2. Архитектура LLaVA-OneVision

2.2. Датасеты

Набор датасетов представлен разными задачами различной сложности.

Br35H - это классификационный датасет, содержащий 3000 изображений MRI мозга, поровну разделённых на 2 класса: здоровые и с опухолями.

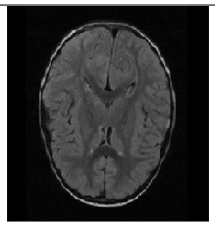
Ключ	**Содержание**
Изображение	
Ответ	No

Таблица 1. Пример вхождения в датасет Br35H

We-Math - это бенчмарк с задачей множественного выбора, содержащий 1740 визуальных математических проблем [26]. Каждый пример состоит из изображения математической задачи, текстового вопроса и 5 вариантов ответа, из которых только один правильный.

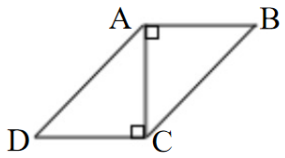
Ключ	**Содержание**
Изображение	
Вопрос	As shown in the diagram, two identical isosceles right triangles are combined to form a parallelogram. The perimeter is reduced by 20 cm. What is the length of AB=AC= ( ) cm?
Варианты ответа	10 cm; 20 cm; 5 cm; 4 cm; No correct answer
Ответ	10 cm

Таблица 2. Пример вхождения в датасет We-Math

ChartQA - это бенчмарк, включающий 9,6 тысяч вопросов, написанных человеком, а также 23,1 тысыч вопросов, сгенерированных на основе описаний диаграмм, написанных человеком [27]. Каждый пример состоит из изображения графика или диаграммы, текстового вопроса и правильного ответа.

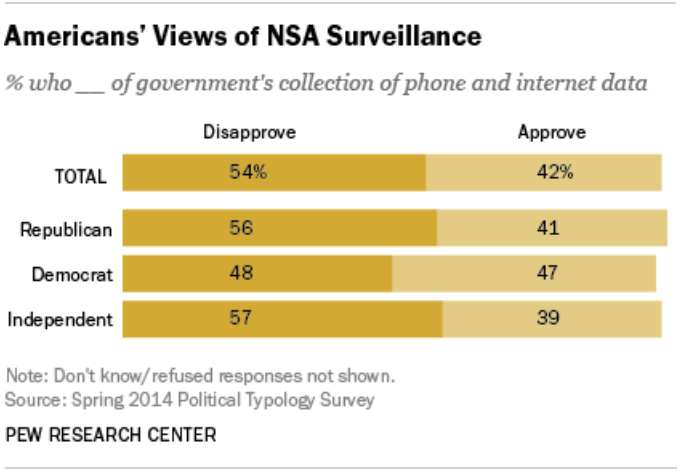
Ключ	**Содержание**
Изображение	
Вопрос	Which party is least likely to approve NSA surveillance?
Ответ	Independant

Таблица 3. Пример вхождения в датасет ChartQA

Глава 3. Вычислительные эксперименты

Аналогично [5] для датасетов создавались обучающие выборки, состоящие из 1000 примеров, тестовые выборки, состоящие из 1000 примеров, и валидационные из 500. В случае недостатка примеров в датасете, тестовая и валидационная выборки были созданы из меньшего количества примеров. Если в датасете не было разделения на обучающую и тестовую выборки, то они были созданы случайным образом. Изображения были приведены к размеру 384x384 пикселей (равноценно 729 визуальным токенам), если превышали это разрешение.

Для модели LLaVA-OneVision-Qwen2-0.5B-SI запуски проводились на NVIDIA RTX 3090 24GB. Для модели LLaVA-OneVision-Qwen2-7B-SI запуски проводились на NVIDIA A100 80GB.

Если не указано иное, использовались следующие гиперпараметры:

Параметр	Значение
Шаг обучения (learning rate)	1e-6
Весовое затухание (weight decay)	0.0
bfloat16	Да
Степень пертубации (zo_eps)	0.001
Планировщик скорости обучения (learning rate scheduler)	Постоянный (constant)

Таблица 4. Параметры запуска MeZO и AdamW

Для экспериментов с AdamW использовался размер батча 1 и 3000 шагов обучения. Для экспериментов с MeZO использовался размер батча 4. Из-за сообщаемой в [5] меньшей скорости сходимости MeZO, количество шагов обучения было увеличено до 5000.

Более подробный список гиперпараметров приведён в приложении 9.

3.1. Подготовка промптов

Так как качество MeZO сильно зависит от качества постановки промпта [5], то изначально мы рассматриваем несколько вариантов шаблона промпта, который может быть использован, тестируя их после обучения в течение одной эпохи (250 шагов) на датасете *Br35H*.

- Промпт 1:
  <image>\nQuestion: Are there signs of a tumor in the image? Yes or No?\nAnswer:\s
- Промпт 2:
  <image>\nQuestion: Are there signs of a tumor in the image? Yes or No?\nAnswer:
- Промпт 3:
  <image>\nQuestion: Are there signs of a tumor? Yes or No?\nAnswer:
- Промпт 4:
  <image>\nQuestion: Are there signs of a tumor in the image? Yes or No?
- Промпт 5:
  <image>\nAre there signs of a tumor in the image? Yes or No?\nAnswer:
- Промпт 6:
  <image> Are there signs of a tumor in the image? Yes or No? Answer:
- Промпт 7:
  <image>\nAre there signs of a tumor in the image? Respond with Yes or No.\nAnswer:

Так, результаты тестирования различных промптов для датасета *Br35H* представлены в таблице 5.

Промпт	**Zero-shot (%)**	**MeZO (%)**
Промпт 1	61.1	78.2
Промпт 2	66.0	52.3
Промпт 3	65.9	59.7
Промпт 4	72.0	51.4
Промпт 5	**62.6**	**82.6**
Промпт 6	52.9	78.1
Промпт 7	50.5	67.3

Таблица 5. Результаты проверки различных промптов

Мы выбираем промпт 5, так как он даёт лучшие результаты при дообучении в течение одной эпохи и является одним из самых производительных для необученной модели (zero-shot).

Мы пользуемся данным шаблоном и для других датасетов, адаптируя его под конкретную формулировку задачи. Так, для датасета *We-Math* в промпт добавлены варианты ответа, а для датасета ChartQA – указание на необходимость найти ответ в изображении.

Выбранные промпты для других датасетов представлены в таблице 6.

Датасет	**Промпт**
Br35H	<image>\nAre there signs of a tumor in the image? Yes or No?\nAnswer:
We-Math	<image>\n<question> Options: <options>\nAnswer:
ChartQA	<image>\nFind the answer in the image for: <question>\nAnswer:

Таблица 6. Промпты для различных датасетов

3.2. Качество дообучения

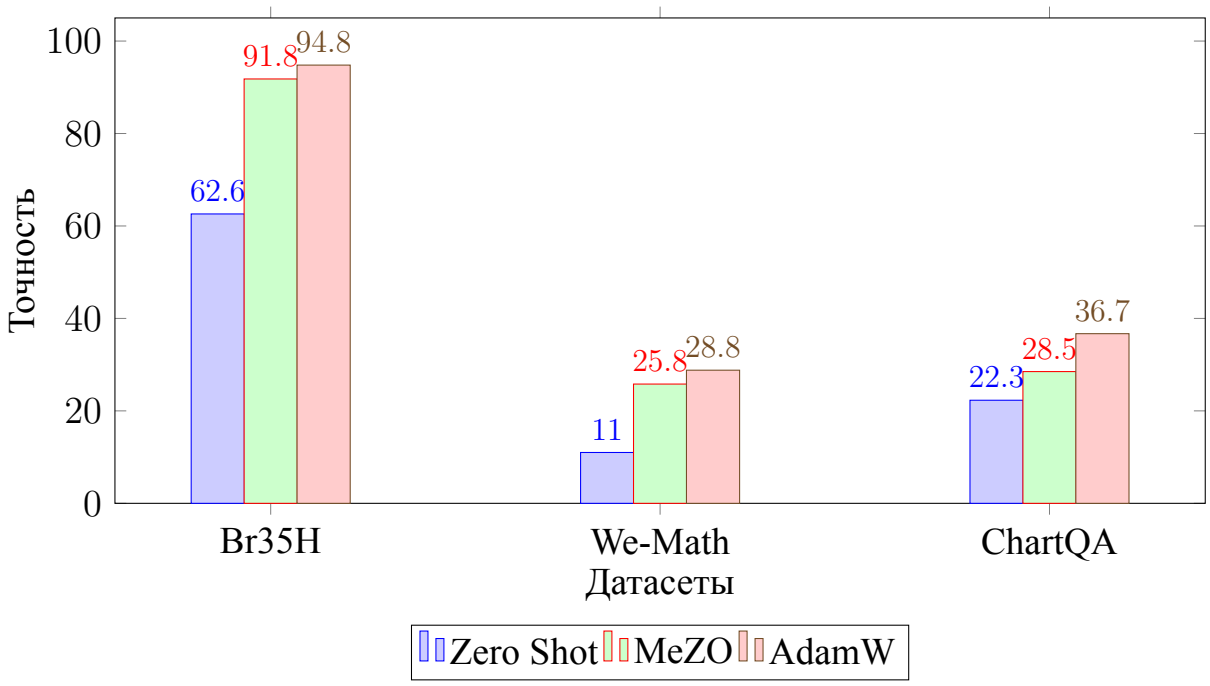


Рис. 3. Общее сравнение результатов MeZO и AdamW на датасетах Br35H, We-Math и ChartQA с моделью LLaVA-OneVision-Qwen2-0.5B-SI.

Как видно из рисунка 3, результаты MeZO ухудшаются по мере усложнения задачи. На простом датасете Br35H метод MeZO обеспечивает прирост, составляющий около 91% от прироста, полученного методом AdamW, тогда как для самого сложного датасета ChartQA этот показатель снижается до примерно 43%.

Более подробные результаты тестирования MeZO и AdamW на датасетах Br35H, We-Math и ChartQA представлены в таблице 7.

Метод	**Br35H**	**We-Math**	**ChartQA**
Zero-shot	62.6	11.0	22.3
AdamW (только LM)	94.3	28.6	35.6
AdamW (LM+Projector)	94.1	28.6	35.9
AdamW (LM+Projector+VisEnc)	**94.8**	**28.8**	**36.7**
MeZO (только LM)	**91.8**	**25.8**	28.0
MeZO (LM+Projector)	91.2	24.8	**28.5**
MeZO (LM+Projector+VisEnc)	91.2	24.4	27.1

Таблица 7. Подробные результаты тестирования MeZO и AdamW на датасетах Br35H, We-Math и ChartQA с использованием разных конфигураций модели LLaVA-OneVision-Qwen2-0.5B-SI.

Как видно из таблицы 7, AdamW показывает лучшие результаты при дообучении всей модели (LM+Projector+VisEnc), когда MeZO демонстрирует лучшее поведение при дообучении только языковой модели (LM) или языковой модели с проекционным слоем (LM+Projector). Однако, разница в обучении разных конфигураций AdamW незначительна и составляет в среднем 0.6% между конфигурациями LM и LM+Projector+VisEnc. Можно предположить, что это связано с высокой предобученностью кодировщика изображений и проекционного слоя.

Так как разница дообучения в различных конфигурациях модели незначительна, в условиях ограниченных вычислительных ресурсов, для варианта модели с 7B параметров запуски проводились только для конфигурации LM.

Метод	**Br35H**	**We-Math**	**ChartQA**
Zero-shot	94.8	19.0	47.7
AdamW (только LM)	96.4	50.4	49.4
MeZO (только LM)	95.2	36.4	45.8

Таблица 8. Результаты тестирования MeZO и AdamW на датасетах Br35H, We-Math и ChartQA с использованием разных конфигураций модели.

Результаты тестирования на модели LLaVA-OneVision-Qwen2-7B-SI представлены в таблице 8 и в целом подтверждают результаты для датасетов We-Math и ChartQA, полученные на модели LLaVA-OneVision-Qwen2-0.5B-SI. MeZO имеет значительное отставание в задаче We-Math, а худшим результатом является ChartQA. Для датасета Br35H MeZO показывает прирост в 0.4% относительно необученной модели, что является незначительным улучшением и составляет 25% от прироста, полученного методом AdamW. Можно предположить, что это связано с высокой предобученностью модели на данной задаче.

3.3. Использование памяти

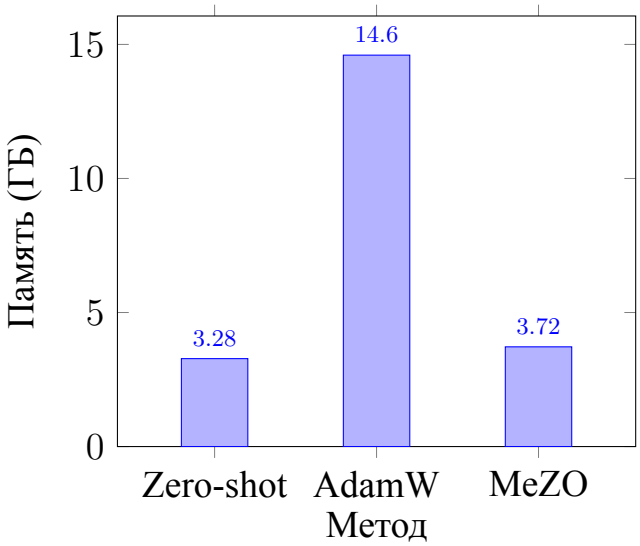


Рис. 4. Сравнение использования памяти для методов Zero-shot, AdamW и MeZO для модели LLaVA-OneVision-Qwen2-0.5B-SI при использованиии максимальной длины токенов 2048 и размера батча 1.

Как видно из рисунка 4, MeZO использует память почти на уровне инференса и сокращает использование памяти по сравнению с AdamW почти в 4 раза.

Для проверки памяти у методов MeZO и AdamW использовалась команда max_memory_allocated() из библиотеки torch.cuda.

3.4. Сходимость и стабильность обучения

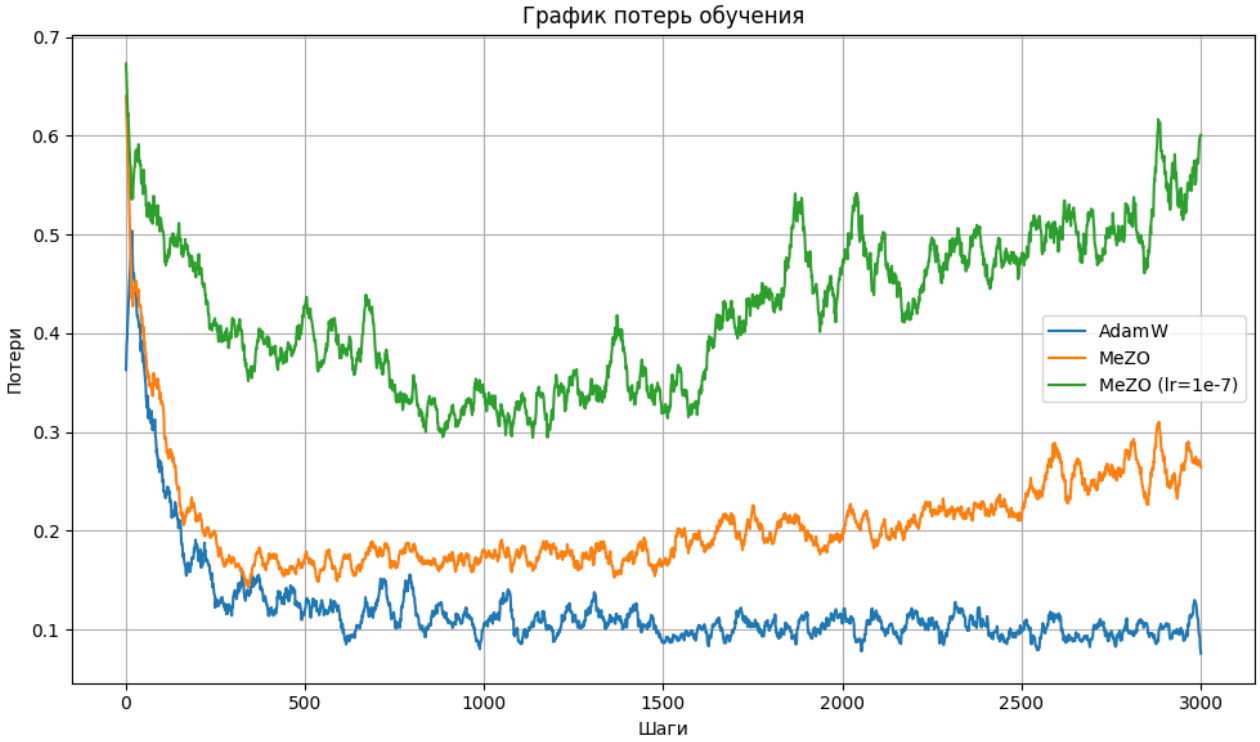


Рис. 5. График потерь обучения для MeZO и AdamW на датасете We-Math для модели LLaVA-OneVision-Qwen2-0.5B-SI

На рисунке 5 представлено сравнение графика потерь методов дообучения.

Метод MeZO демонстрирует разрыв в сходимости по сравнению с AdamW и не может достичь тех же минимальных значений потерь, что и AdamW.

Вторым важным наблюдением является нестабильность обучения при использовании MeZO. График показывает, что потери имеют тенденцию к увеличению после начального снижения. При этом, попытка снижения шага обучения (lr=1e-7) для MeZO не приводит к улучшению результатов.

На рискунке 6 представлено сравнение графика потерь для модели LLaVA-OneVision-Qwen2-7B-SI.

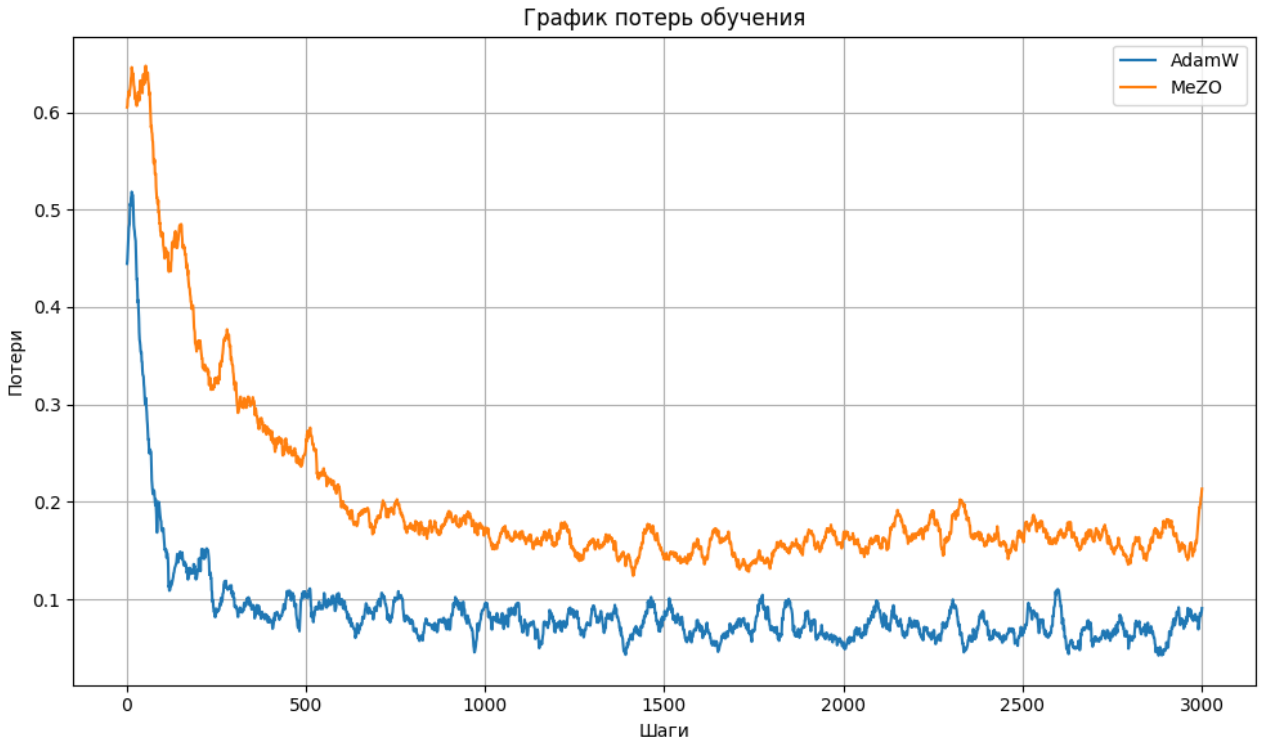


Рис. 6. График потерь обучения для MeZO и AdamW на датасете We-Math для модели LLaVA-OneVision-Qwen2-7B-SI

Как видно из рисунка 6, разрыв в сходимости MeZO относительно AdamW сохраняется и для модели с 7B параметров. Кроме того, заметна меньшая скорость сходимости MeZO по сравнению с моделью с 0.5B параметров.

Также был проведён эксперимент с изменением размера батча.

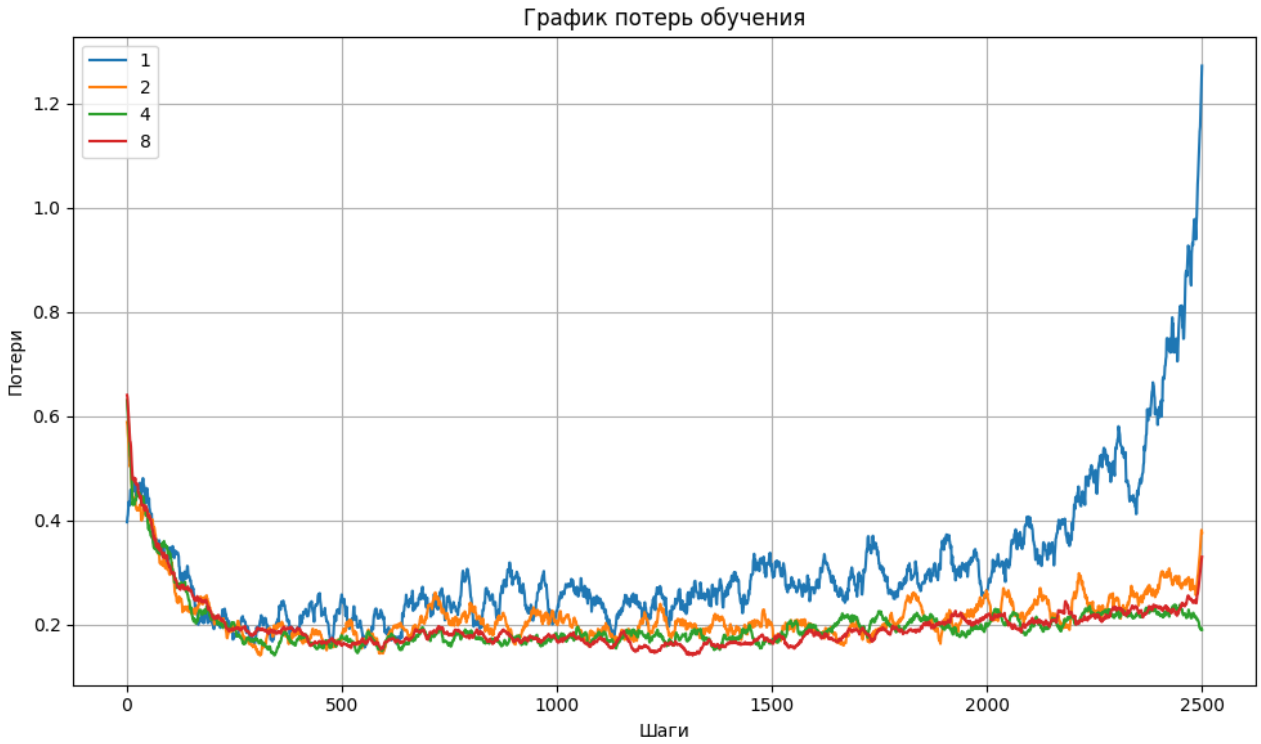


Рис. 7. График потерь обучения для MeZO на датасете We-Math с разными размерами батча для модели LLaVA-OneVision-Qwen2-0.5B-SI

[24] утверждает о большей нестабильности MeZO на меньших размерах батча. На рисунке 7 представлено сравнение MeZO с различными размерами батча. Как видно из рисунка, это утверждение подтверждается. При размере батча 1 заметна сильная нестабильность и расхождение метода. Для размера батчей выше 2 эффект становится малозаметным, но разрыв в сходимости относительно AdamW остаётся.

Выводы

В данной главе были проведены всесторонние вычислительные эксперименты, направленные на оценку эффективности методов тонкой настройки мультимодальных моделей с использованием оптимизации нулевого порядка (метод MeZO) по сравнению с традиционным методом (AdamW). Основные выводы, полученные в ходе экспериментов, можно сформулировать следующим образом:

- **Качество дообучения.** По результатам тестирования на датасетах *Br35H*, *We-Math* и *ChartQA* видно, что применение MeZO позволяет улучшить точность моделей относительно необученного состояния. Однако модели, дообученные методом оптимизации нулевого порядка, могут серьёзно уступать версиям, настроенным с помощью AdamW (например, для датасета *ChartQA* и модели LLaVA-OneVision-Qwen2-0.5B-SI точность MeZO составляет 28.5% против 36.7% при использовании AdamW).
- **Потребление памяти.** Одним из ключевых преимуществ метода MeZO является его крайне низкое потребление оперативной памяти. Было показано, что использование памяти при дообучении методом MeZO соответствует режиму инференса (около 3.7 ГБ) и значительно меньше, чем при использовании AdamW (14.6 ГБ).
- **Сходимость и стабильность обучения.** Анализ кривых функции потерь продемонстрировал, что MeZO характеризуется менее стабильной сходимостью по сравнению с AdamW. Наблюдался разрыв в поведении функции потерь, а параметры оптимизации оказались чувствительны к выбору гиперпараметров (таких как размер батча и шаг обучения). Эксперименты с изменением размера батча подтвердили, что при малых размерах батча метод демонстрирует повышенную нестабильность.
- **Роль промпт-инжиниринга.** Дополнительный анализ показал, что качество итоговых результатов существенно зависит от корректной формулировки промпта.

Заключение

В данной работе было рассмотрено применение методов оптимизации нулевого порядка для тонкой настройки мультимодальных больших языковых моделей, что позволяет существенно снизить требования к оперативной памяти по сравнению с традиционными методами, основанными на вычислении градиентов через обратное распространение ошибки.

Основные итоги исследования можно сформулировать следующим образом:

- Был произведён анализ архитектурных решений мультимодальных моделей и подходов к тонкой настройке.
- Представлена теоретическая база метода SPSA и его применения для оценки градиентов без необходимости обратного распространения. Память-эффективная версия метода (MeZO) позволяет обновлять параметры модели, используя только прямые проходы, что значительно уменьшает объём требуемой памяти.
- Экспериментальная часть работы включала сравнение методов оптимизации MeZO и AdamW на ряде датасетов (Br35H, We-Math, ChartQA). Результаты показали, что:
  - Для более эффективного обучения MeZO требуется тщательная настройка промптов, что может быть трудоёмко и не всегда представляется возможным в реальных приложениях.
  - Для простых датасетов по точности MeZO демонстрирует конкурентоспособные результаты, хотя и немного уступает AdamW.
  - Для сложных задач, таких как ChartQA, MeZO показывает значительное снижение точности по сравнению с AdamW.

- MeZO потребляет значительно меньше оперативной памяти, что особенно актуально при работе с крупномасштабными моделями.
- Сходимость метода MeZO характеризуется неустойчивостью, особенно на небольших размерах батча.

Таким образом, хотя MeZO демонстрирует потенциал для обучения MLLM в условиях ограниченных вычислительных ресурсов, метод требует дальнейших исследований для повышения стабильности и качества обучения.

Перспективными направлениями будущей работы для улучшение стабильности и сходимости метода MeZO можно считать применение техник уменьшения дисперсии, адаптивной настройки гиперпараметров. Например, в [5] отмечалась возможность применения метода MeZO-Adam для сложных задач в качестве направления для будущих исследований.

Код, использованный во время выполнения настоящей работы, выложен в открытый доступ на сайте GitHub и может быть воспроизведен. (URL: https://github.com/tatarinovst2/course-work-5) Также в репозиторий LLaVA-NeXT, на основе которого проводились эксперименты, добавлен пулл-реквест с реализацией MeZO. (URL: https://github.com/LLaVA-VL/LLaVA-NeXT/pull/469)

Список литературы

1. *Liu H.*, *Li C.*, *Wu Q.*, *Lee Y. J.* Visual Instruction Tuning // CoRR. — 2023. — T. abs/2304.08485.
2. *Li B.* [et al.]. LLaVA-OneVision: Easy Visual Task Transfer // Transactions on Machine Learning Research. — 2025.
3. *Robbins H.*, *Monro S.* A Stochastic Approximation Method // The Annals of Mathematical Statistics. — 1951. — Сент. — Т. 22, № 3. — С. 400—407.
4. *Kingma D.*, *Ba J.* Adam: A Method for Stochastic Optimization // International Conference on Learning Representations. — 2014. — Дек.
5. *Malladi S.*, *Gao T.*, *Nichani E.*, *Damian A.*, *Lee J. D.*, *Chen D.*, *Arora S.* Fine-Tuning Language Models with Just Forward Passes // Thirty-seventh Conference on Neural Information Processing Systems. — 2023.
6. *Rumelhart D. E.*, *Hinton G. E.*, *Williams R. J.* Learning representations by back-propagating errors // Nature. — 1986. — Т. 323, № 6088. — С. 533—536.
7. *Spall J.* Multivariate stochastic approximation using a simultaneous perturbation gradient approximation // IEEE Transactions on Automatic Control. — 1992. — Т. 37, № 3. — С. 332—341.
8. *Ghadimi S.*, *Lan G.* Stochastic First- and Zeroth-Order Methods for Nonconvex Stochastic Programming // SIAM Journal on Optimization. — 2013. — Т. 23, № 4. — С. 2341—2368.
9. *Brown T.* [et al.]. Language Models are Few-Shot Learners // Advances in Neural Information Processing Systems. Т. 33. — Curran Associates, Inc., 2020. — С. 1877—1901.
10. *Chowdhery A.* [et al.]. PaLM: scaling language modeling with pathways // J. Mach. Learn. Res. — 2023. — Янв. — Т. 24, № 1.

11. *Li C.*, *Gan Z.*, *Yang Z.*, *Yang J.*, *Li L.*, *Wang L.*, *Gao J.* Multimodal Foundation Models: From Specialists to General-Purpose Assistants // Found. Trends. Comput. Graph. Vis. — Hanover, MA, USA, 2024. — Май. — Т. 16, № 1/2. — С. 1—214.

12. *Ghandi T.*, *Pourreza H.*, *Mahyar H.* Deep Learning Approaches on Image Captioning: A Review // ACM Comput. Surv. — New York, NY, USA, 2023. — Окт. — Т. 56, № 3.

13. *Radford A.* [et al.]. Learning Transferable Visual Models From Natural Language Supervision // Proceedings of the 38th International Conference on Machine Learning. Т. 139. — PMLR, 18--24 Jul.2021. — С. 8748—8763.

14. *Li J.*, *Li D.*, *Savarese S.*, *Hoi S.* BLIP-2: bootstrapping language-image pre-training with frozen image encoders and large language models // Proceedings of the 40th International Conference on Machine Learning. — Honolulu, Hawaii, USA : JMLR.org, 2023.

15. *Zhu D.*, *Chen J.*, *Shen X.*, *Li X.*, *Elhoseiny M.* MiniGPT-4: Enhancing Vision-Language Understanding with Advanced Large Language Models // The Twelfth International Conference on Learning Representations. — 2024.

16. *Ye Q.* [et al.]. mPLUG-Owl: Modularization Empowers Large Language Models with Multimodality // ArXiv. — 2023. — Т. abs/2304.14178.

17. *Hu E. J.*, *Shen Y.*, *Wallis P.*, *Allen-Zhu Z.*, *Li Y.*, *Wang S.*, *Wang L.*, *Chen W.* LoRA: Low-Rank Adaptation of Large Language Models // International Conference on Learning Representations. — 2022.

18. *Sun X.*, *Ji Y.*, *Ma B.*, *Li X.* A Comparative Study between Full-Parameter and LoRA-based Fine-Tuning on Chinese Instruction Data for Instruction Following Large Language Model. — 2023.

19. *Lv K.*, *Yan H.*, *Guo Q.*, *Lv H.*, *Qiu X.* AdaLomo: Low-memory Optimization with Adaptive Learning Rate // Findings of the Association for Computational Linguistics: ACL 2024. — Bangkok, Thailand : Association for Computational Linguistics, 08.2024. — C. 12486—12502.

20. *Shamir O.*, *Zhang T.* Stochastic Gradient Descent for Non-smooth Optimization: Convergence Results and Optimal Averaging Schemes // Proceedings of the 30th International Conference on Machine Learning. T. 28. — Atlanta, Georgia, USA : PMLR, 17--19 Jun.2013. — C. 71—79.

21. *Ghorbani B.*, *Krishnan S.*, *Xiao Y.* An Investigation into Neural Net Optimization via Hessian Eigenvalue Density // Proceedings of the 36th International Conference on Machine Learning. T. 97. — PMLR, 09--15 Jun.2019. — C. 2232—2241.

22. *Sagun L.*, *Evci U.*, *Guney V. U.*, *Dauphin Y.*, *Bottou L.* Empirical Analysis of the Hessian of Over-Parametrized Neural Networks. — 2018.

23. *Yao Z.*, *Gholami A.*, *Keutzer K.*, *Mahoney M. W.* PyHessian: Neural Networks Through the Lens of the Hessian // 2020 IEEE International Conference on Big Data (Big Data). — IEEE. 2020. — C. 581—590.

24. *Gautam T.*, *Park Y.*, *Zhou H.*, *Raman P.*, *Ha W.* Variance-reduced zeroth-order methods for fine-tuning language models // Proceedings of the 41st International Conference on Machine Learning. — Vienna, Austria, 2024.

25. *Loshchilov I.*, *Hutter F.* Decoupled Weight Decay Regularization // International Conference on Learning Representations. — 2019.

26. *Qiao R.* [et al.]. We-Math: Does Your Large Multimodal Model Achieve Human-like Mathematical Reasoning? // CoRR. — 2024. — T. abs/2407.01284.

27. *Masry A.*, *Long D.*, *Tan J. Q.*, *Joty S.*, *Hoque E.* ChartQA: A Benchmark for Question Answering about Charts with Visual and Logical Reasoning // Findings of the Association for Computational Linguistics: ACL 2022. — Dublin, Ireland : Association for Computational Linguistics, 05.2022. — C. 2263—2279.

28. *Tatarinov M. D.* Application of Zeroth-Order Optimization Methods for Fine-Tuning of Multimodal Large Language Models. — URL: https://github.com/tatarinovst2/course-work-5.

29. *Tatarinov M. D.* Application of Zeroth-Order Optimization Methods for Fine-Tuning of Multimodal Large Language Models. — URL: https://github.com/LLaVA-VL/LLaVA-NeXT/pull/469.

Приложение А

Параметр	**Значение**
Vision encoder	google/siglip-so400m-patch14-384
Vision tower learning rate	1e-6
MM projector type	mlp2x_gelu
Vision select layer	-2
Use image start/end tokens	False
Use image patch token	False
Group by modality length	True
Image aspect ratio	anyres_max_9
Image grid pinpoints	(1x1),...,(6x6)
Patch merge type	spatial_unpad
Precision	bfloat16
Per device train batch size	4
Gradient accumulation steps	1
Learning rate	1e-6
Weight decay	0.0
LR scheduler type	constant
TF32	False
Gradient checkpointing	False
Lazy preprocess	True
Torch compile	False
Dataloader drop last	True
Frames upper bound	32

Таблица 9. Полный список общих параметров для MeZO и AdamW

Параметр	**Значение**
Attention implementation	sdpa
Model max length	1792

Таблица 10. Полный список общих параметров для MeZO и AdamW. Продолжение

Для MeZO использовались:

Параметр	**Значение**
ZO epsilon	1e-3
ZO number of directions	1

Таблица 11. Параметры MeZO