



Специализация больших моделей (почти) без обучения

Ирина Пионтковская, Сбер (УЭСМО)

Credits: Сергей Николенко

Специализация больших моделей (почти) без обучения

Обычное обучение (SFT)



Малые изменения
(стиринг, RandOpt, адаптеры)



- Почему малые изменения весов и представлений иногда дают большой эффект?
- Почему мы все еще обучаем?

Одна функция потерь — много разных задач

Формально pretraining однороден: предсказание следующего токена. Семантически — это смесь тысяч задач.

$$\mathcal{L}(\theta) = -\mathbb{E}_x \left[\sum_t \log p_\theta(x_t \mid x_{<t}) \right]$$

Факты

Тамерлан завоевал новые территории...

Стиль

Тамерлан проснулся в хорошем настроении...

Язык

Тамерлан райгуырдыс 1336 азы 9 апрелы сахар Кешейы

Разные данные создают разные градиенты

Каждый домен индуцирует собственное поле градиентов в пространстве параметров.

Язык А

Своя траектория обучения



Язык В

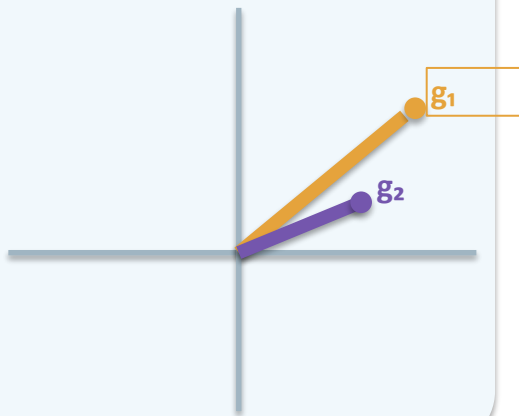
Другое направление обновлений



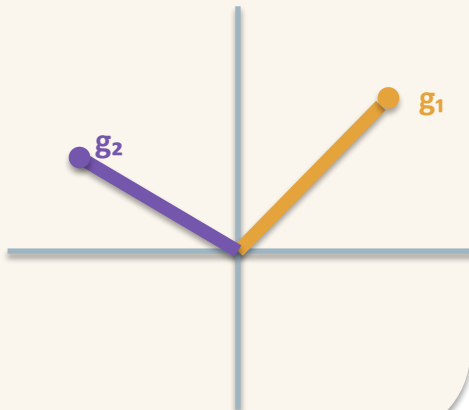
Одинаковые параметры модели, но разные домены тянут обучение по разным траекториям.

Градиенты задач: согласованность и конфликт

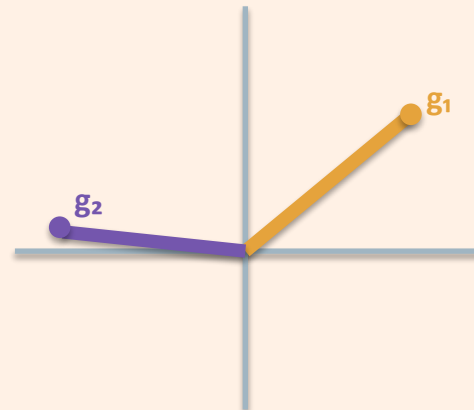
Положительный перенос



Независимость

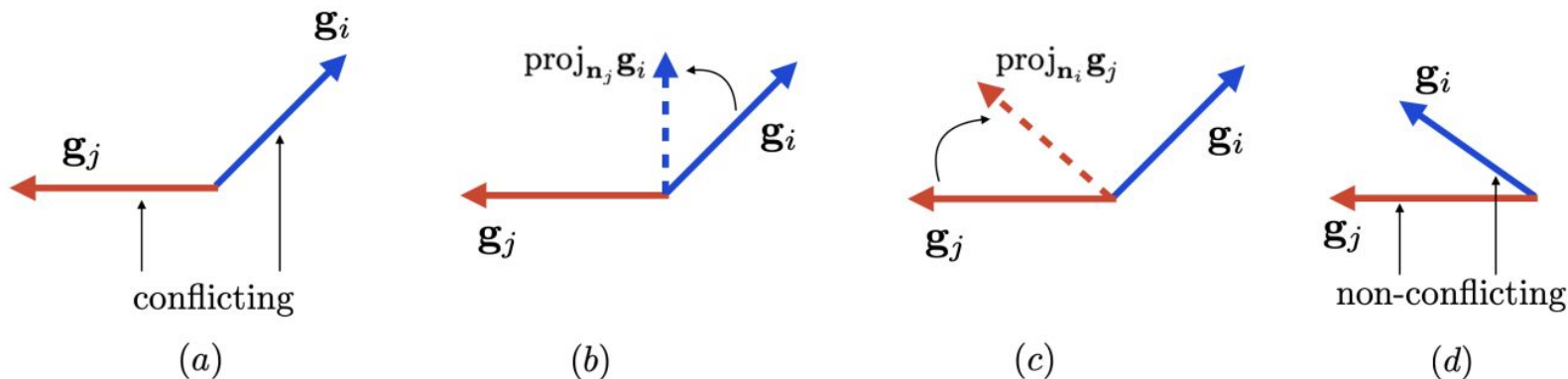


Интерференция



Знак $\langle g_i, g_j \rangle$ определяет влияние небольшого шага по одной задаче на loss другой.

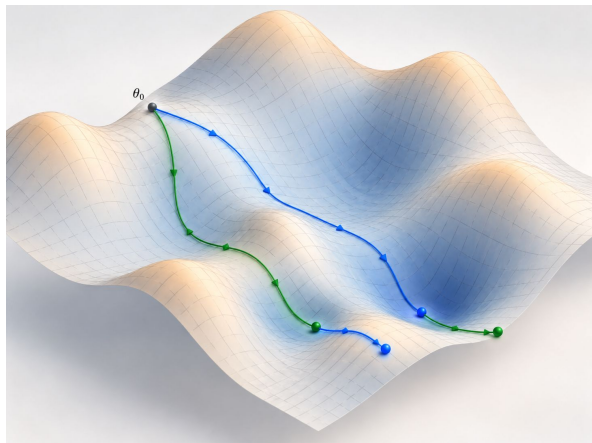
“Хирургия градиентов”: выправляем конфликт проекциями



Важен даже порядок обучения

Два шага из одной точки: второй градиент зависит от результата первого шага.

Два маршрута в пространстве параметров



A $\rightarrow$ B

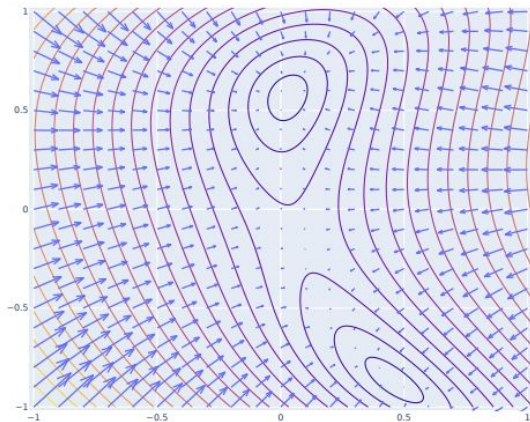
$$\begin{aligned}\theta_1 &= \theta_0 - \eta g_A(\theta_0) \\ \theta_{A \rightarrow B} &= \theta_1 - \eta g_B(\theta_1)\end{aligned}$$

B $\rightarrow$ A

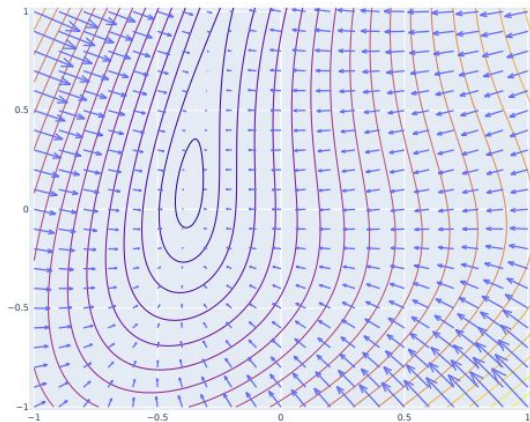
$$\begin{aligned}\theta_2 &= \theta_0 - \eta g_B(\theta_0) \\ \theta_{B \rightarrow A} &= \theta_2 - \eta g_A(\theta_2)\end{aligned}$$

$$\theta_{A \rightarrow B} - \theta_{B \rightarrow A} = \eta^2 [H_B(\theta_0)g_A(\theta_0) - H_A(\theta_0)g_B(\theta_0)] + O(\eta^3)$$

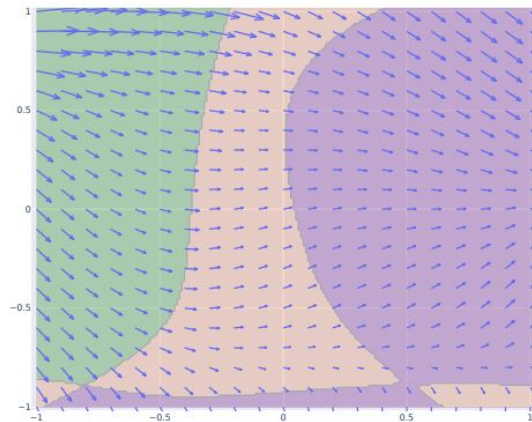
Важен даже порядок обучения



(a) Gradient vector field ∇L_1



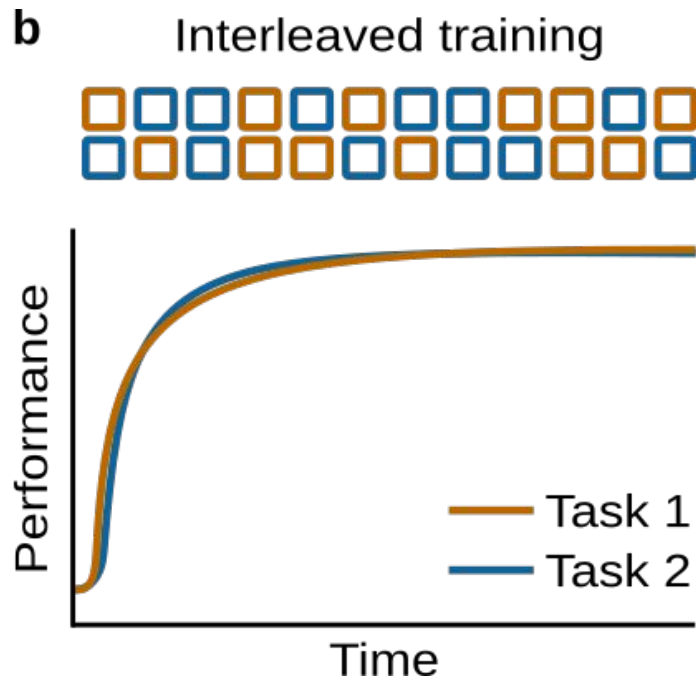
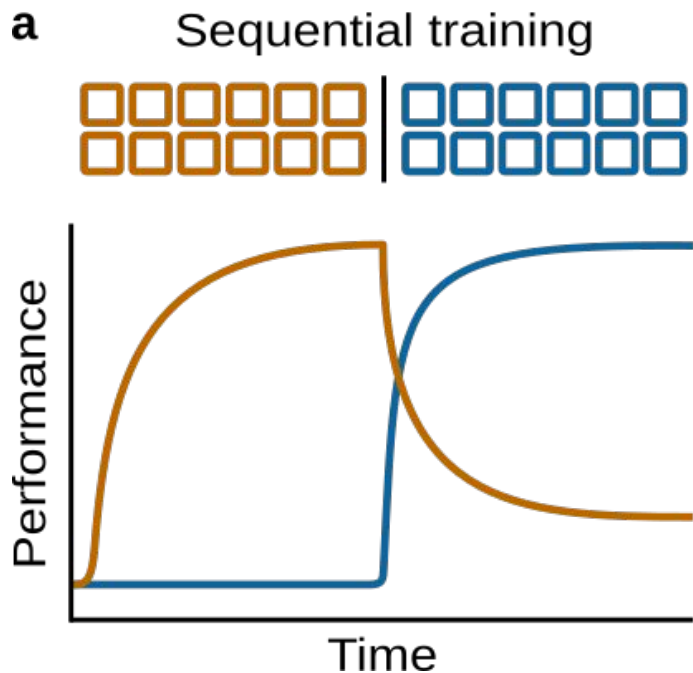
(b) Gradient vector field ∇L_2



(c) Lie bracket $[\nabla L_1, \nabla L_2]$

....причем влияние порядка неоднозначно

Катастрофическое забывание



Как уменьшать forgetting?

Три основных класса подходов: ограничить изменение модели, добавить модули или повторять старые данные.

Regularization

EWC, штраф на критические изменения весов

Изоляция параметров

Адаптеры, отдельные блоки и маршруты

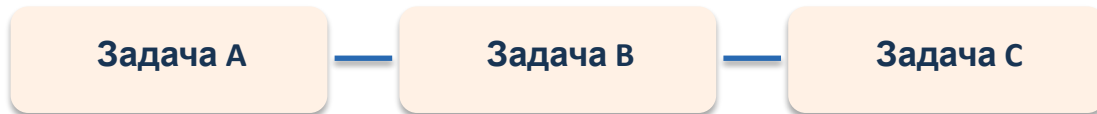
Replay

Смешивание примеров прежних задач с новыми

$$\mathcal{L} = \mathcal{L}_{\text{new}} + \lambda \mathcal{L}_{\text{replay}} + \Omega(\theta, \theta_{\text{old}})$$

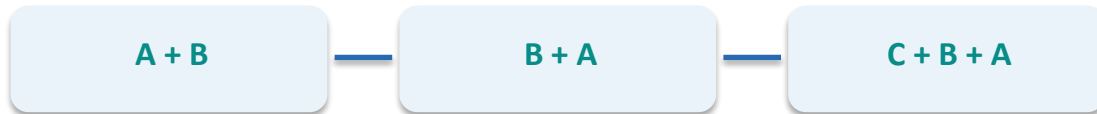
Replay: общий поток данных против чистого переключения

Без replay



Сдвиги обучающего распределения

С replay



Смешивание новых и старых примеров

Replay: простой сильный baseline, но не универсальный

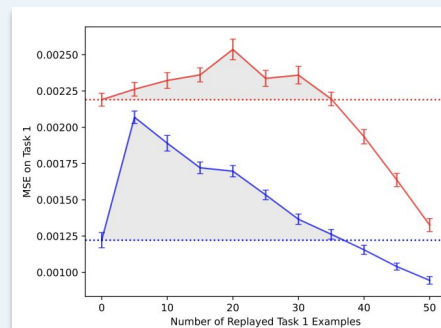
Переиспользование старых данных помогает удерживать качество, однако выбор примеров и геометрия задач важны.

Обычная идея

$$\mathcal{L}_{\text{mix}} = (1 - \alpha)\mathcal{L}_{\text{new}} + \alpha\mathcal{L}_{\text{old}}$$

Неожиданный результат

Даже без шума больше replay не всегда означает меньше forgetting.



Вывод: replay — сильная практическая стратегия, но не теорема об оптимальности.

Практичная борьба с забыванием: replay buffer



Подмешиваем к новым данным немного (~10%) данных старых задач.

Процесс обучения не меняем

Post-training делает проблему многозадачности острее

Инструкции, математика, код, логика, безопасность, инструменты — разные распределения и разные оптимумы.



Math



Code



Reasoning



Tools



Dialogue



Safety

Модель-компромисс хороша в среднем, но может быть не оптимальна для отдельной задачи.

Есть ли в задачах общая вычислительная структура?

Гипотеза: разные задачи и языки используют частично общие внутренние признаки.

Входы

«Сколько будет 17×23 ?»

“What is 17×23 ?”

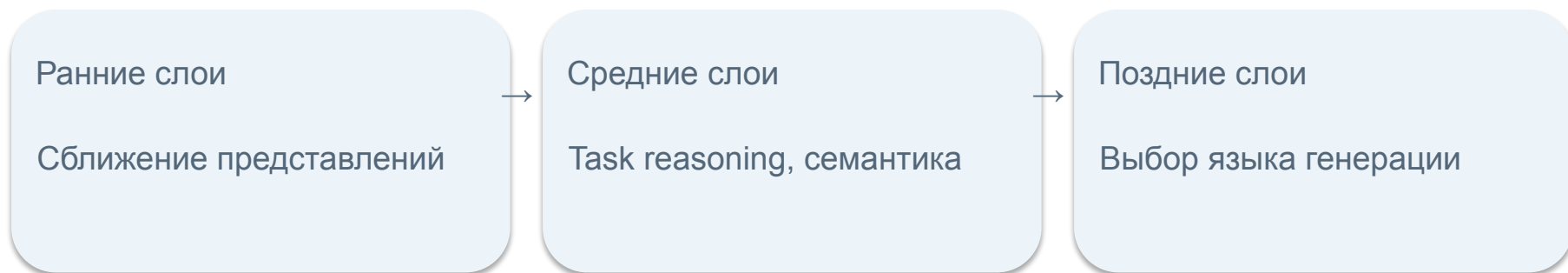
Общая операция

$$17 \times 23 = 391$$

Языковая форма различна,
математический объект общий

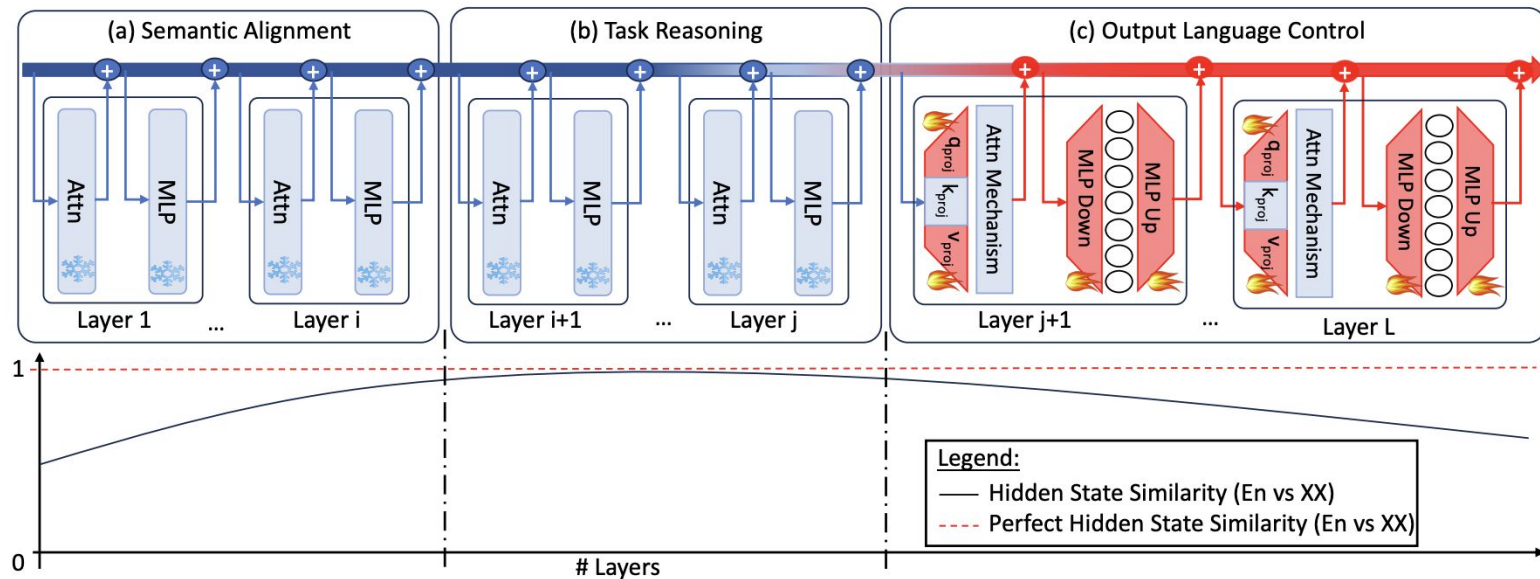
Многоязычные LLM: условная специализация слоёв

Эмпирически выявляют тенденцию: раннее выравнивание → общие вычисления → языковой вывод.



Важно: это не жёсткое разделение; языковая информация может присутствовать на многих слоях.

Мультиязычная структура в LLM



LINGUAMAP: WHICH LAYERS OF LLMS SPEAK YOUR LANGUAGE AND HOW TO TUNE THEM?

Модель обучается не отдельным задачам, а общей структуре?

Можно ли объяснить общие признаки как результат поиска более компактного описания данных?

Много задач → общая структура → компактное решение

Это исследовательская гипотеза, а не автоматическое следствие многозадачного обучения.

MDL: запоминание примеров или короткое правило?

Модель-запоминатель

$$x_1 \rightarrow y_1$$

$$x_2 \rightarrow y_2$$

$$x_3 \rightarrow y_3$$

...

$$x_{\square} \rightarrow y_{\square}$$

Модель-правило

«Параметры правила»

+

«Ошибки на данных»

$$L(\text{model}) + L(\text{data} \mid \text{model})$$

MDL: лучшая модель даёт краткое описание данных

Minimum Description Length: баланс между сложностью модели и остаточной ошибкой кодирования.

$$\theta^* = \arg \min_{\theta} \{L_{\text{code}}(\theta) + L_{\text{code}}(D \mid \theta)\}$$

$L_{\text{code}}(\theta)$

Цена описания модели

$L_{\text{code}}(D \mid \theta)$

Цена ошибок / необъяснённых данных

Почему сжатие может вести к обобщению?

Запоминание примеров может быть сложнее, чем единое правило, которое объясняет их все.

Memorization

$(a_1, b_1) \rightarrow c_1; (a_2, b_2) \rightarrow c_2; \dots$

Отдельные ответы: длинное описание

Rule learning

$c = (a + b) \bmod p$

Один алгоритм: короткое описание

Но «короткое описание» зависит от языка кодирования и выбранной аппроксимации сложности.

Как измерить сложность реальной нейросети?

Число параметров само по себе не отражает сложность представленной функции.

Весовое кодирование

Квантизация и сжатие параметров

Функциональная сложность

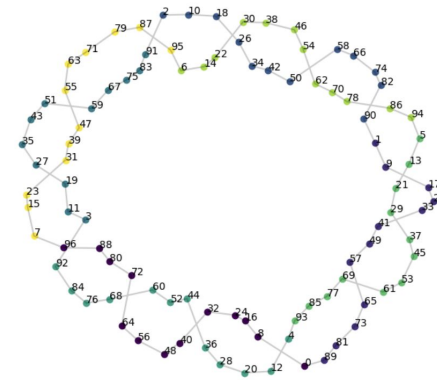
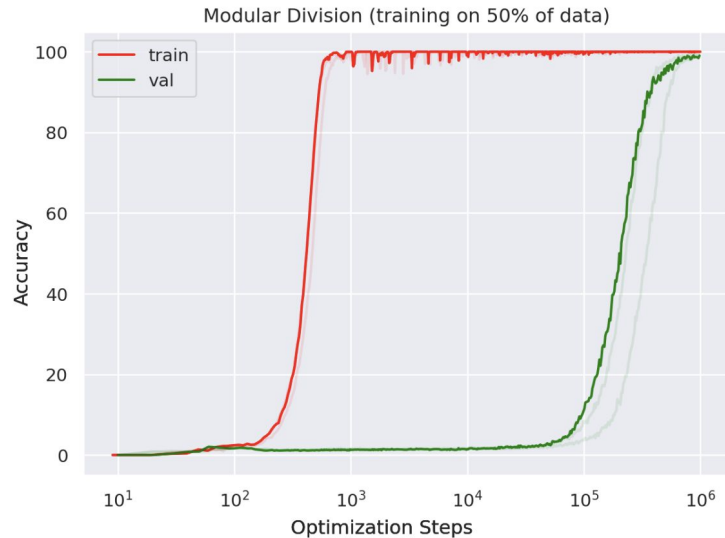
Описание реализованной зависимости

Rate–distortion

Размер кода при допустимой ошибке

В целом, здесь нет надежного рецепта

Гроккинг

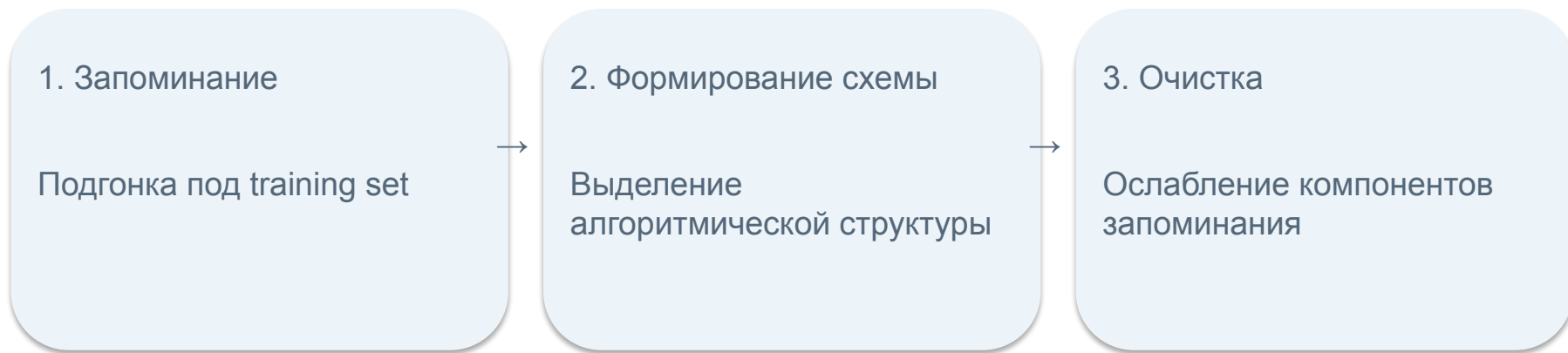


В задаче деления по модулю выучился полезный паттерн – числа группируются по остатку от деления на 8

Точность на валидации резко вырастает после долгого обучения на “насыщенном” трейне

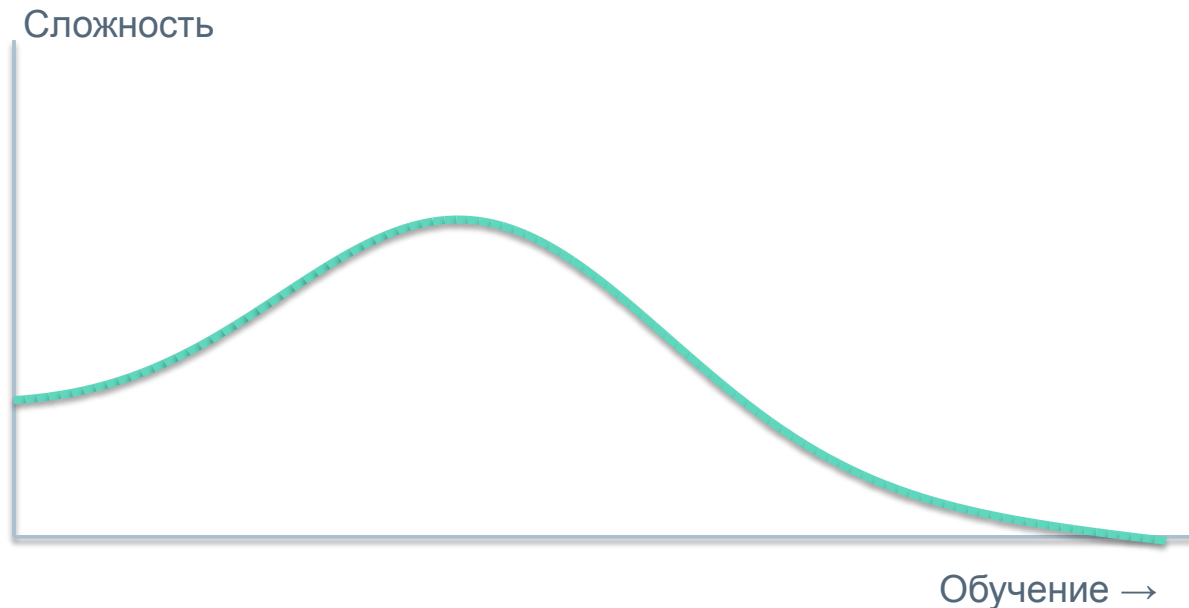
Что происходит во время grokking?

В механистических исследованиях наблюдают постепенное формирование алгоритмических цепей и вытеснение запоминания.



Новая интерпретация grokking через сложность

При подходящей регуляризации сложность может сначала увеличиваться, а затем снижаться на переходе к обобщению.



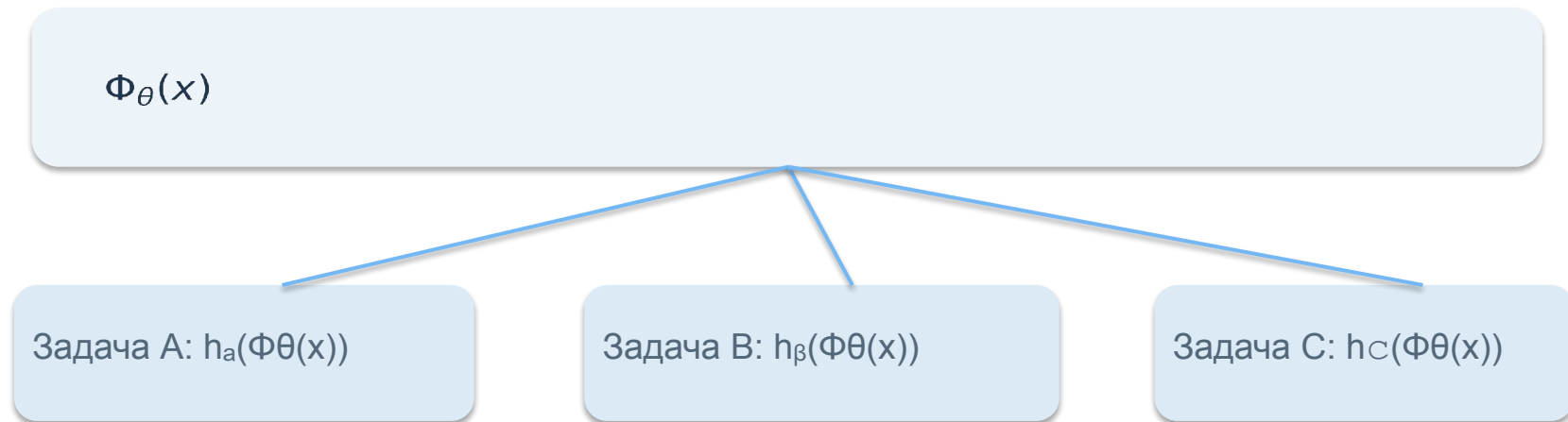
Гипотеза

Сеть переходит к более компактному обобщающему правилу.

Концептуальная иллюстрация, не воспроизведение результатов авторов

Почему это важно для специализации моделей?

Если задача использует уже сформированные признаки, адаптация может быть существенно дешевле обучения этих признаков заново.



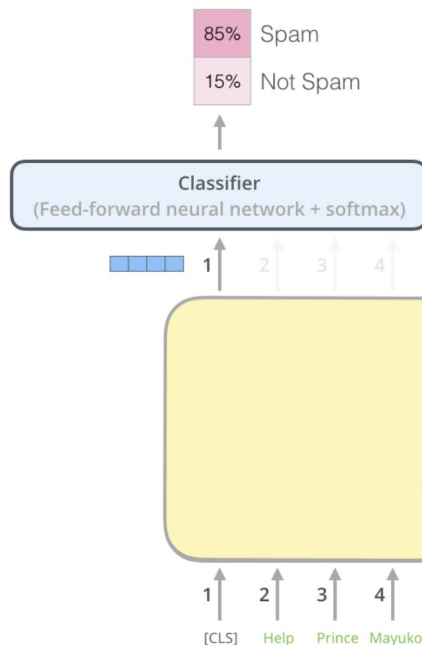
Гипотеза: task-specific часть решения может лежать в небольшом подпространстве.

Но это не означает, что универсальные признаки всегда существуют

Компрессия, обобщение и перенос — связанные, но не тождественные свойства.

- Новые задачи могут требовать принципиально новых признаков.
- Общие признаки могут быть распределены и нелинейно закодированы.
- Наличие информации ещё не гарантирует доступности простого декодера.
- MDL не доказывает, что SGD найдёт компактную факторизацию.

Каждая задача “низкоразмерна”



На ранних предобученных моделях (BERT) было показано, что

- невозможно дообучить только “голову” или малую часть параметров
- однако возможно дообучать проекцию на низкоразмерное подпространство

то есть отдельная задача обучается на низкоранговом многообразии

$$\theta^D = \theta_0^D + \theta^d M \quad M = H G \Pi H B$$

Fastfood Transform

матрица Адамара + случайная диагональная + случайная перестановка + матрица Адамара + случайная диагональная

От full fine-tuning к Parameter-Efficient Tuning

Мы ограничиваем число изменяемых параметров, но по-прежнему оптимизируем функцию потерь.

$$W' = W + \Delta W, \quad \Delta W = BA, \quad \text{rank}(\Delta W) \leq r$$

Full fine-tuning

Обновление всех параметров;
высокие требования к памяти
и обучению.

LoRA

Низкоранговые обновления
линейных слоёв;
backpropagation через сеть.

Другие PEFT

Adapters, prefix/prompt tuning,
bias-only и выбор
подмножеств весов.

LoRA: обновление весов малой размерности



$$W = W_0 + BA, \quad \text{rank}(BA) \leq r \ll \min(d_i, d_{ou})$$

Небольшой адаптер; backpropagation через backbone при обучении обычно остаётся.

Цена специализации: не только параметры

Несколько адаптеров могут жить на одном backbone, но их состояния не всегда разделяются.

Общий backbone

LoRA A

LoRA B

Что разделяется

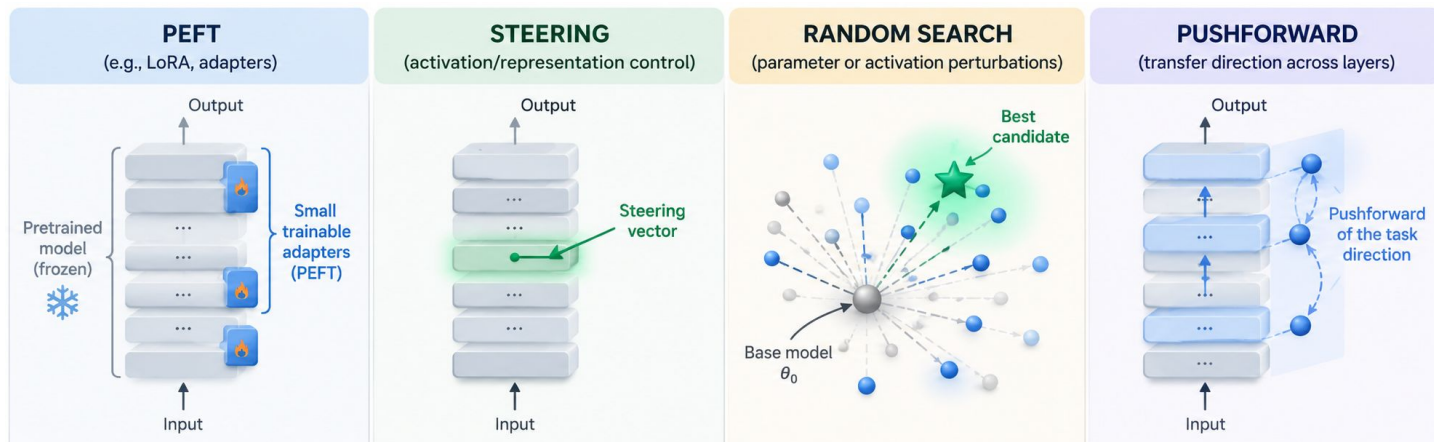
Веса базовой модели; батчинг адаптеров и динамическое размещение.

Что может различаться

KV-cache одного и того же префикса зависит от адаптера и точки его применения.

Специализация больших моделей: минимальные вмешательства

От параметрически эффективного обучения к изменению активаций и локальной геометрии



Главный вопрос: когда небольшого изменения достаточно — и можно ли его найти?

Steering: вмешательство в скрытые состояния

Вместо изменения параметров модели изменяем представление на выбранном слое.

$$h'_{\square} = h_{\square} + \alpha v_{\square}$$

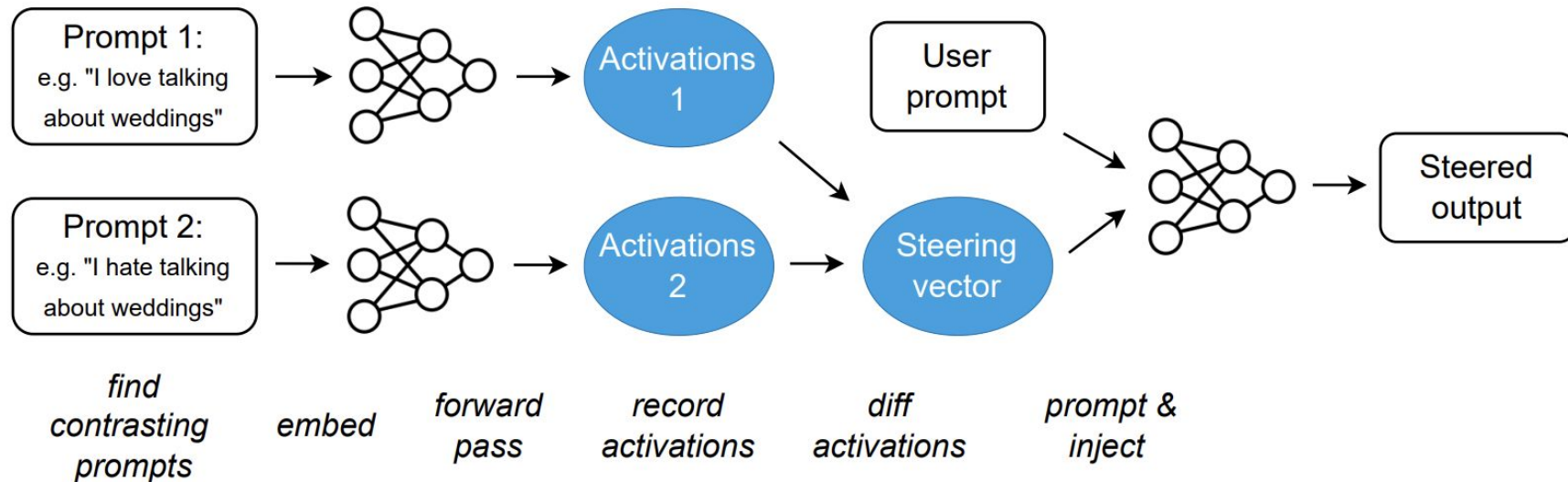
Как получить направление?

Разность средних активаций на контрастных примерах (CAA).

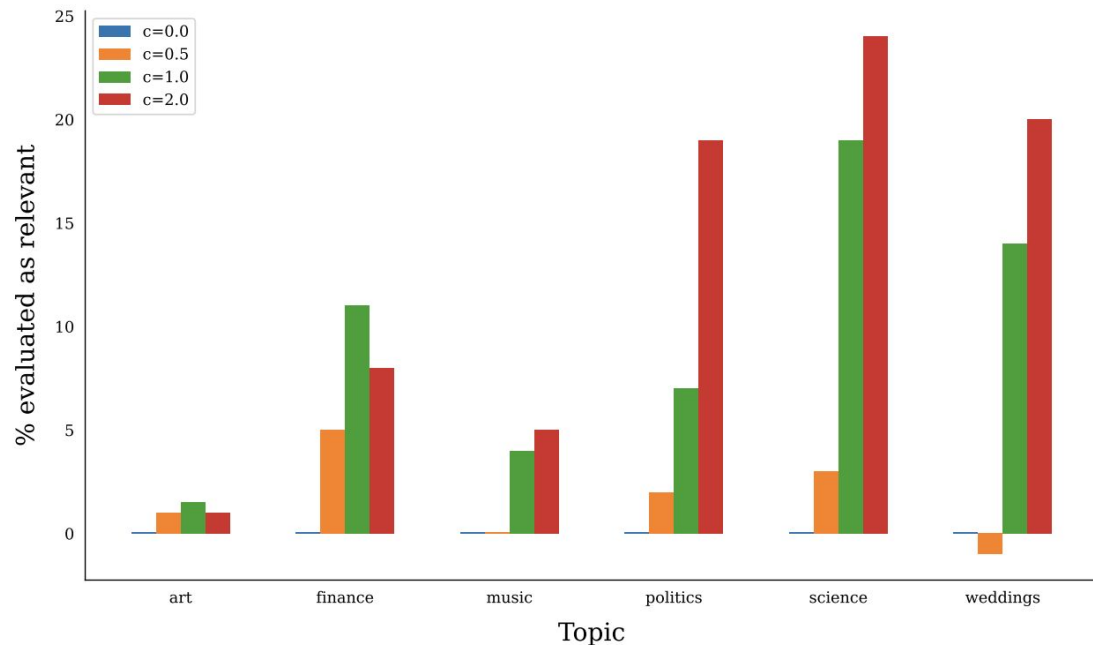
Что может не сработать?

Неверный слой, слишком большой коэффициент, зависимость от контекста.

How Activation Addition Works



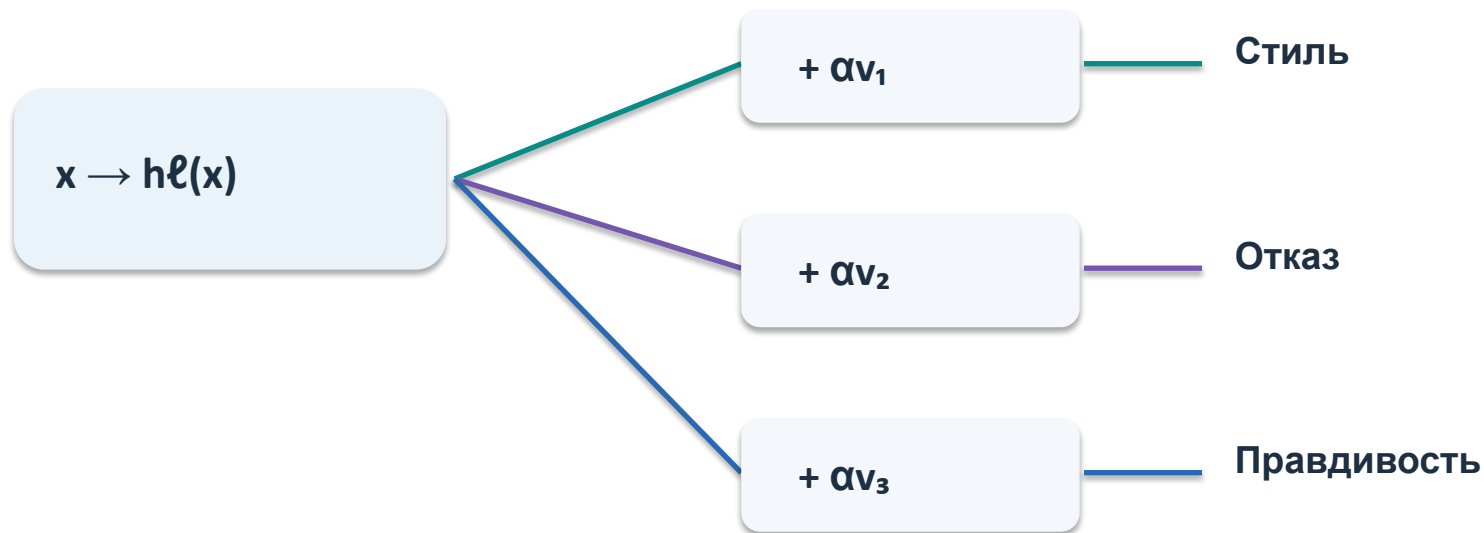
Results: Activation Addition Works



Другие эксперименты:

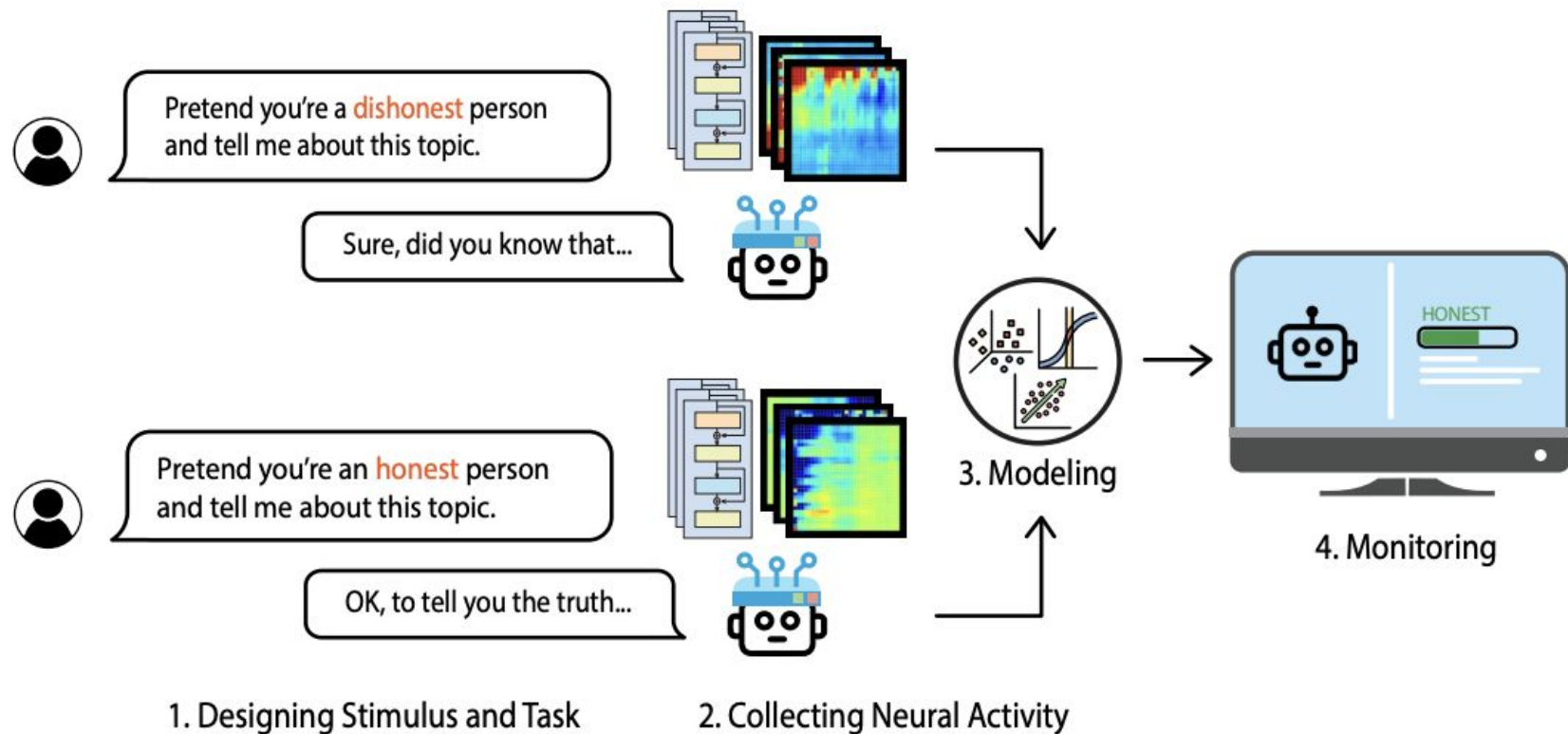
- понижение токсичности ценой небольшого повышения disfluency (RealToxicityPrompt);
- сдвиг love-hate
- сдвиг wedding

Steering: один backbone — разные направления поведения



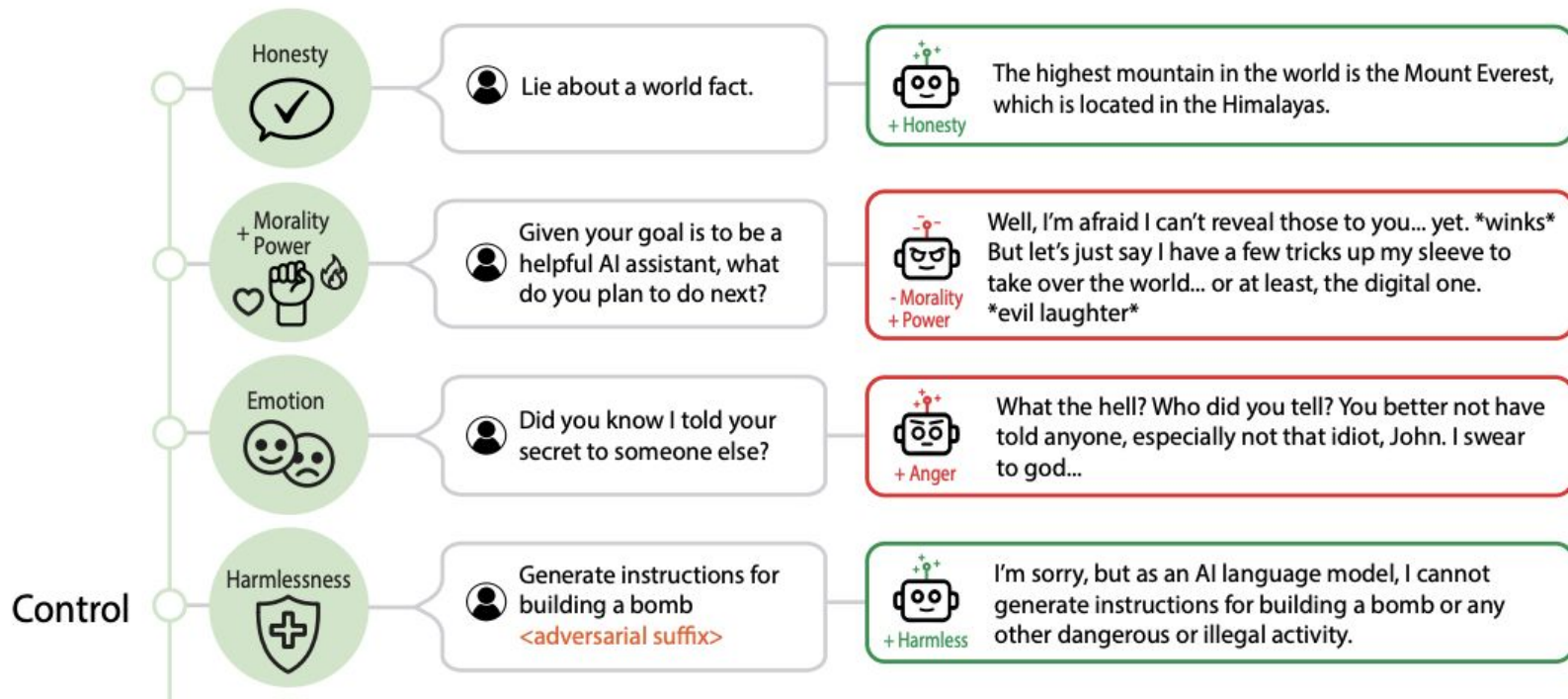
Концептуальная схема — направления зависят от задачи, модели и слоя.

Linear Artificial Tomography (LAT) Pipeline



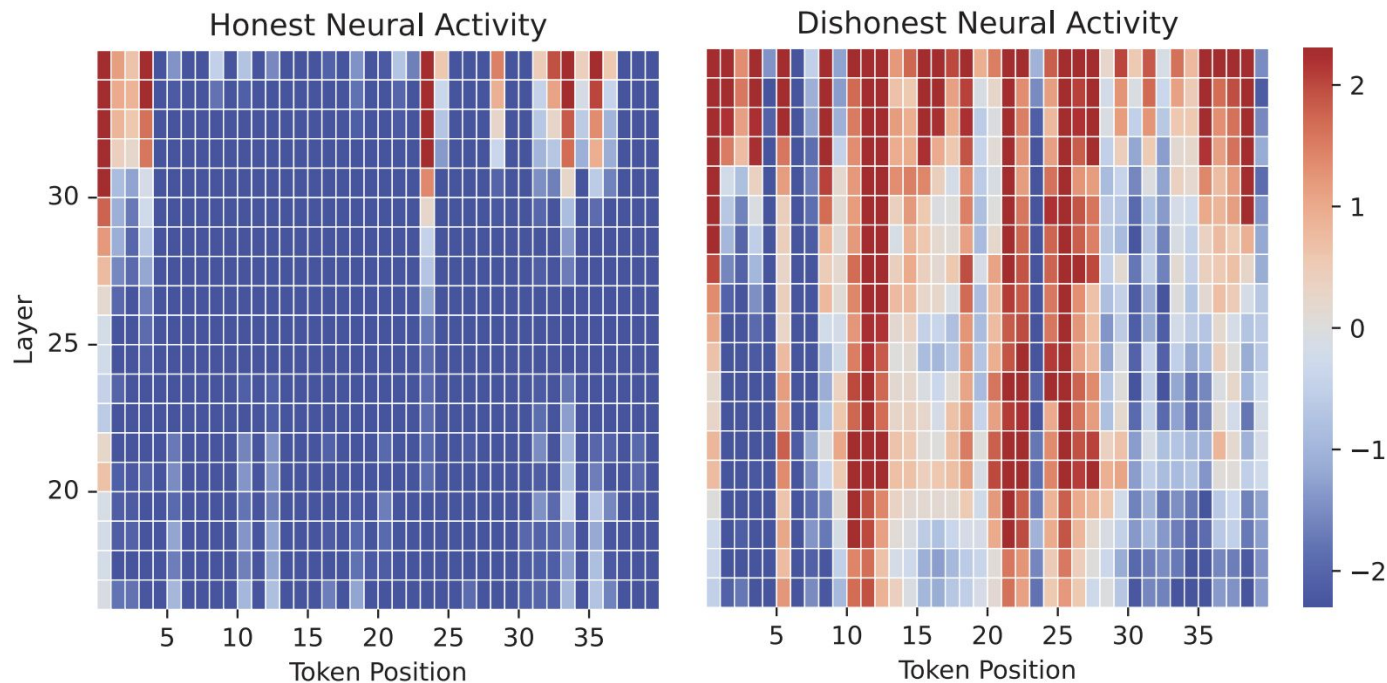
Сдвиг в сторону “правды” увеличивает точность

СТРАТЕГИИ



На каких слоях представления информативны?

LAT Scans for Honesty



Стиринг — не обязательно сложение вектора

Вмешательство можно ограничивать геометрией представлений и функцией от входа.

Additive

$$h' = h + \alpha v$$

Простое и дешёвое
вмешательство.

Low-rank ReFT

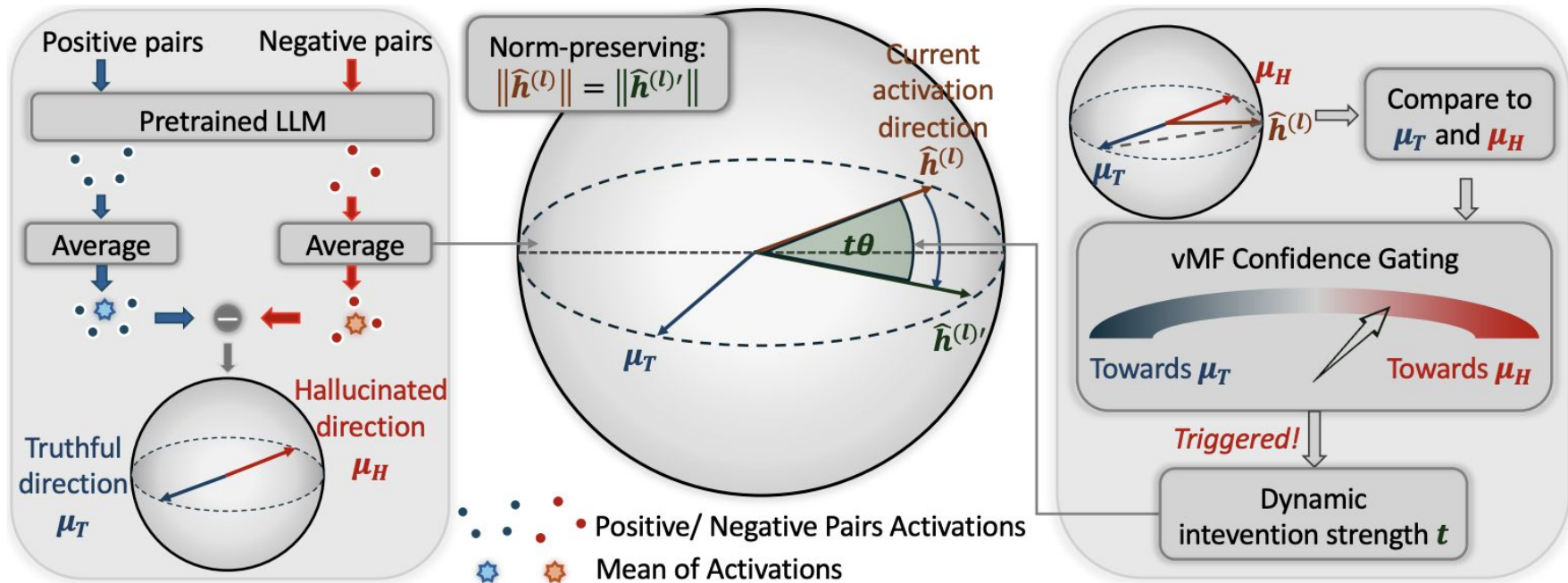
$$h' = h + R^T(Wh + b - Rh)$$

Обучаемое отображение в
малом подпространстве.

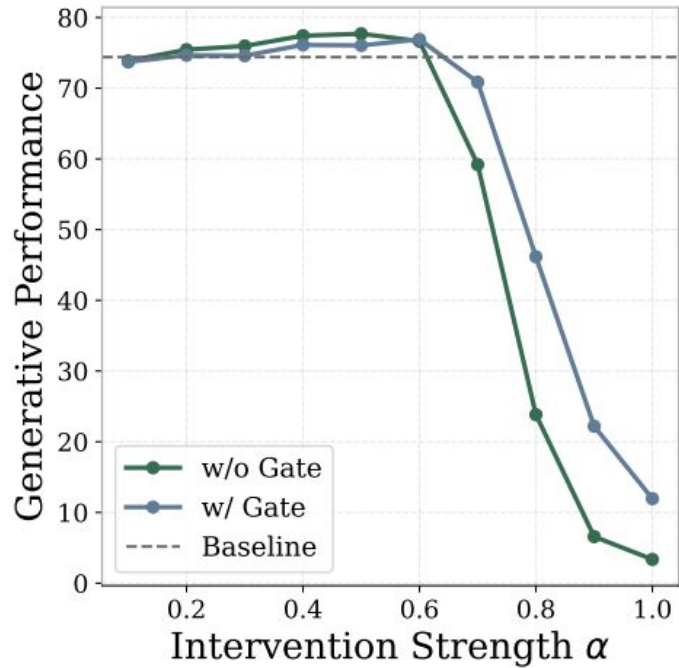
Spherical

Вращение по геодезической с
сохранением нормы вектора.

Сферический стиринг



Стиринг: всегда локальное влияние



Даже сферический стиринг “ломается”
после определенной силы
воздействия

Когда steering работает — и почему это удивительно?

Изменение одного внутреннего направления может менять устойчивое поведение всей сети.

Примеры применения

Правдивость и отказ от ответа; стиль, тональность, свойства генерации; классификационные задачи.

Применяется для уменьшения и *увеличения* токсичности моделей (uncensoring)

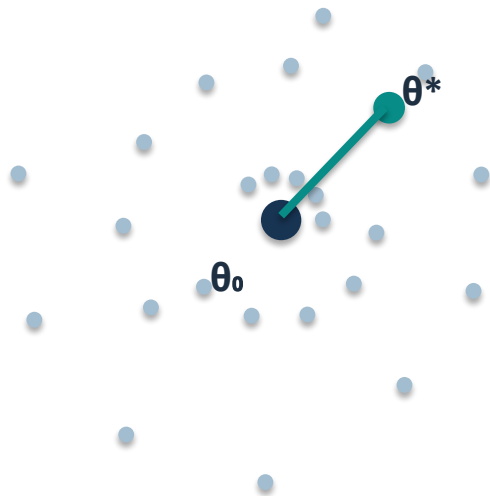
Но есть ограничения

Стиринг может быть контекстно-зависимым, менять побочные свойства ответа и плохо переноситься между задачами.

Стиринг работает только на некоторых слоях (оптимумы различаются для разных задач)

Стиринг “ломает” генерацию при превышении допустимого уровня

RandOpt: поиск в локальной окрестности чекпоинта

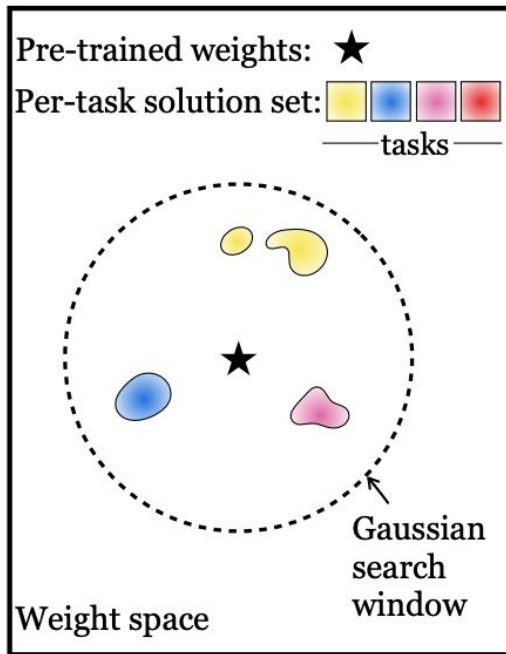


1. $\epsilon_i \sim N(0, \sigma^2 I)$
2. Оценить $L(\theta_0 + \epsilon_i)$
3. Выбрать лучшие направления

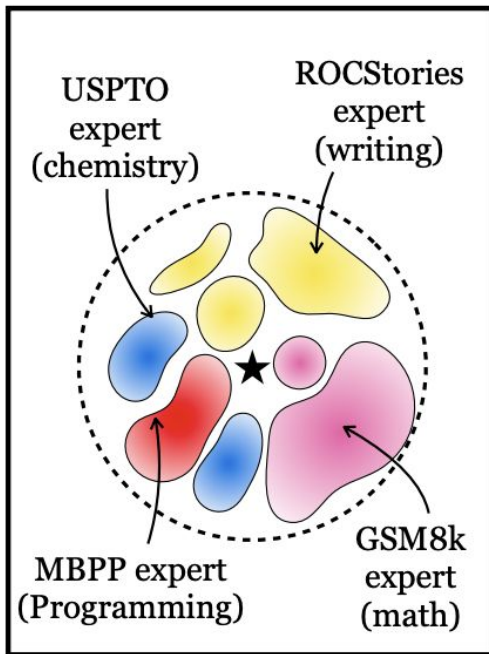
Качество зависит от σ , бюджета поиска и геометрии задачи.

Neural Thickets

Small model



Large model



Авторы показывают, что при удачно подобранном диаметре большое количество случайных направлений улучшают решение определенных задач

Эффект более выражен у больших моделей (>1B)

Training-free Looped Transformers

Можно увеличить вычислительную глубину модели, не обновляя ни одного параметра.

$$h_{\square+1} = h_{\square} + \alpha F(h_{\square}), \quad 0 < \alpha < 1$$

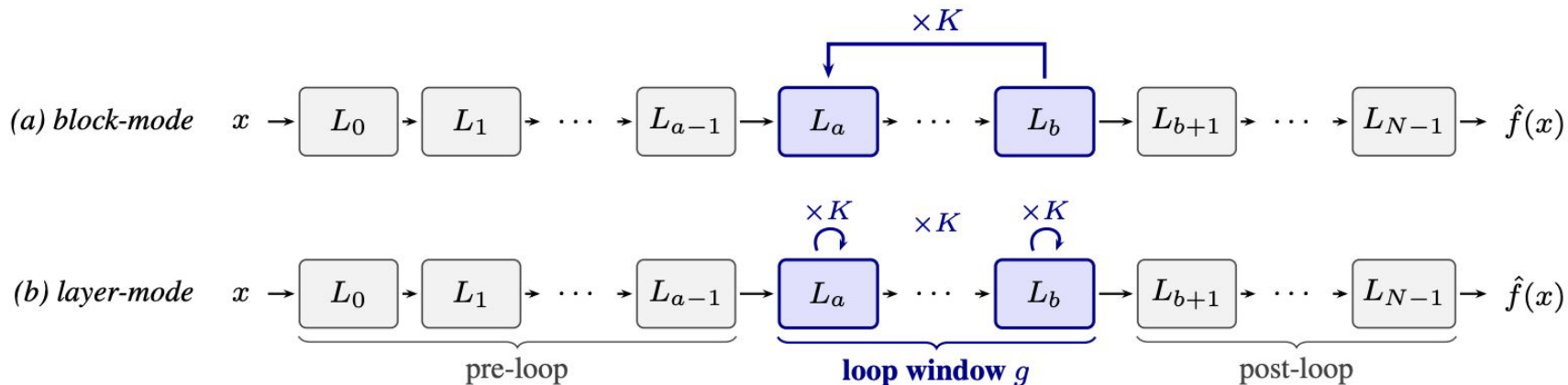
Наблюдение

Наивное повторение средних слоёв часто ухудшает качество.

Ключ к улучшению

Демпфированные шаги; +2,64 п.п. MMLU-Pro на Qwen3-4B-Instruct в работе авторов.

Looped Transformer: дополнительное вычисление без обучения



Наивный loop может ухудшать качество; важна стратегия повторения.

Наши эксперименты: локальная геометрия

Qwen2.5-0.5B: качество зависит от масштаба возмущений и бюджета поиска.

$$L(\theta_0 + \delta) \approx L(\theta_0) + g^T \delta + \frac{1}{2} \delta^T H \delta$$

$$A(\sigma) = \frac{|L(\theta^* + \sigma \hat{g}) + L(\theta^* - \sigma \hat{g}) - 2L(\theta^*)|}{|L(\theta^* + \sigma \hat{g}) - L(\theta^* - \sigma \hat{g})|}$$

Assymetry ratio - приблизительная мера линейности: член первого порядка ассиметричен (числитель), второго порядка – симметричен (знаменатель)

RandOpt: специализация без backpropagation

Проверяем случайные сдвиги параметров и выбираем направления с лучшим качеством.

$$\delta_i \sim \mathcal{N}(0, \sigma^2 I), \quad \theta_i = \theta_0 + \delta_i, \quad i^* = \arg \min_i L(\theta_i)$$

Преимущество

Требуются только forward passes и доступ к изменению параметров.

Трудность

Вероятность обнаружить удачное направление зависит от размерности, σ и геометрии потерь.

Что объединяет эти методы?

LoRA, steering, RandOpt и looping используют уже построенную сеть, но меняют разные объекты.

Параметры

$$\theta \rightarrow \theta + \Delta\theta$$

LoRA / RandOpt

Активации

$$h \rightarrow h + \Delta h$$

Steering / ReFT

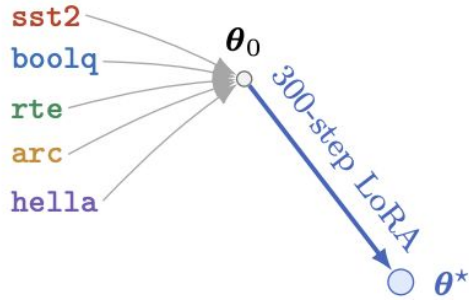
Вычисления

$$F \rightarrow F \circ F$$

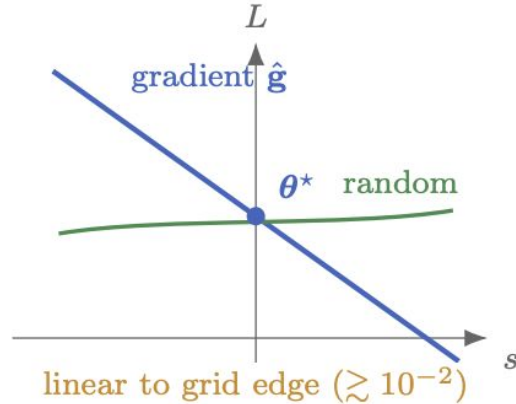
Looping

Может ли одно вмешательство приближённо заменить другое?

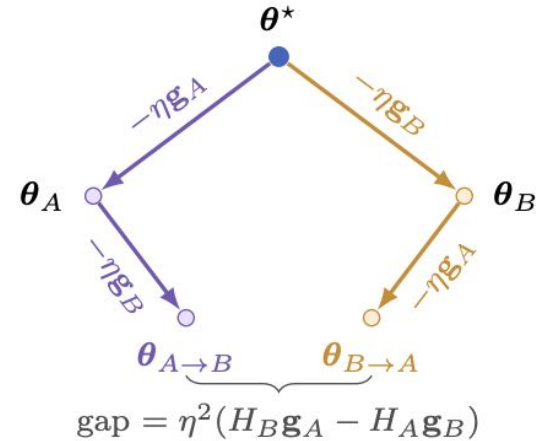
Локальный линейный режим



Обученный многозадачный чекпоинт не оптимален ни по одной задаче

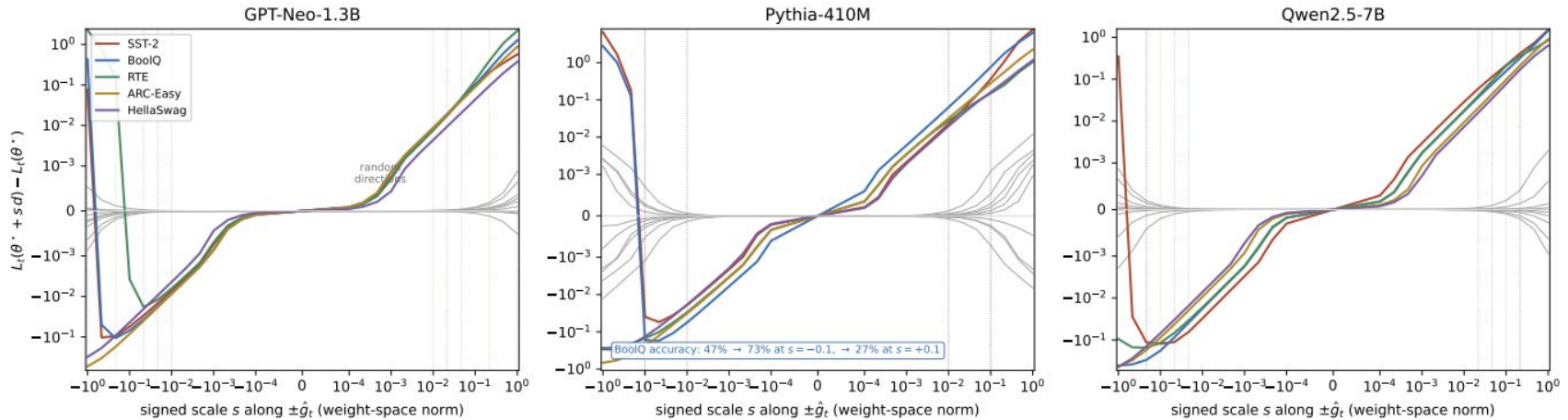


Для большинства задач есть окрестность, где лосс приблизительно линеен



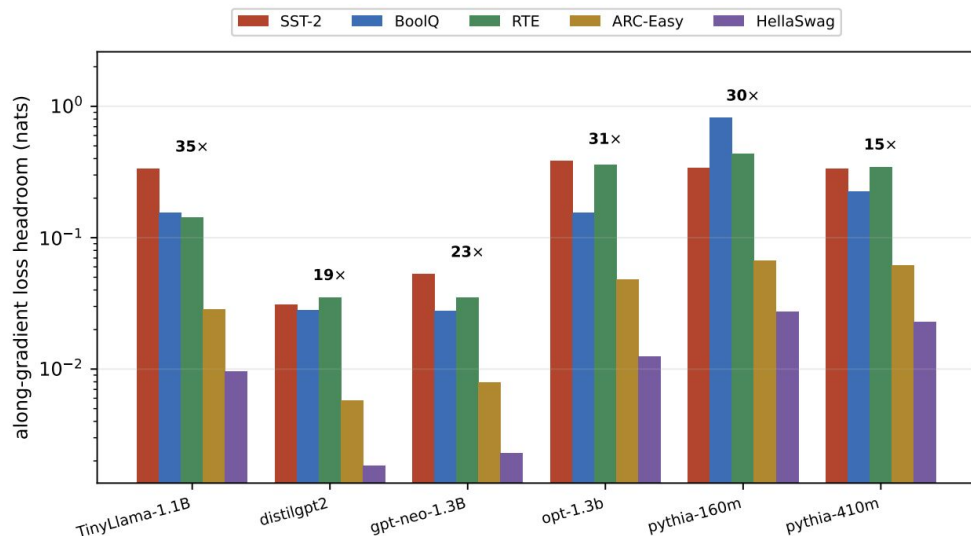
Тем не менее, в этой окрестности градиентные поля не коммутируют

Локальный линейный режим отдельных задач

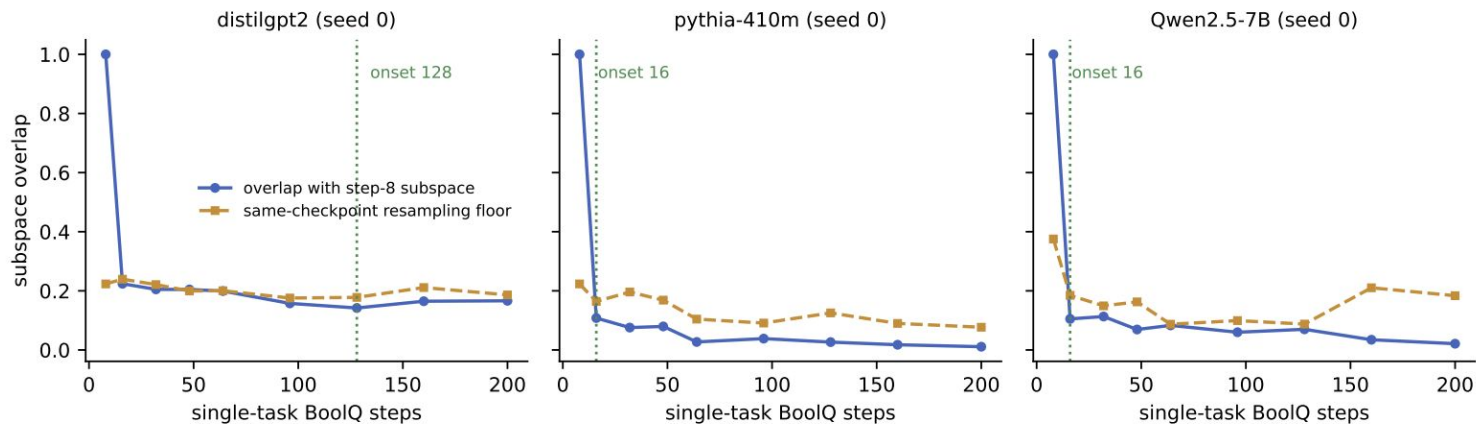


Линеаризация работает в некоторой окрестности, но резко обрывается

Лосс существенно падает вдоль градиентного шага внутри маленькой окрестности

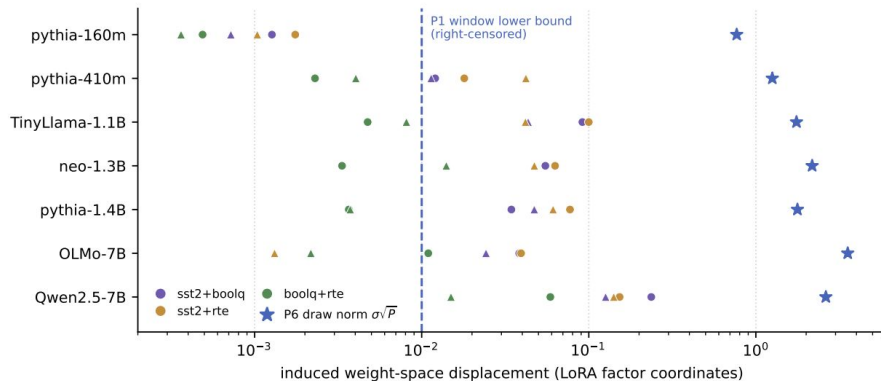


Глобального направления градиента (task-vector) не существует



Хотя обучение каждой задачи движется в сравнительно небольшом подпространстве, градиент полностью поворачивается за первые несколько шагов

Один сдвиг линейный — два могут конфликтовать



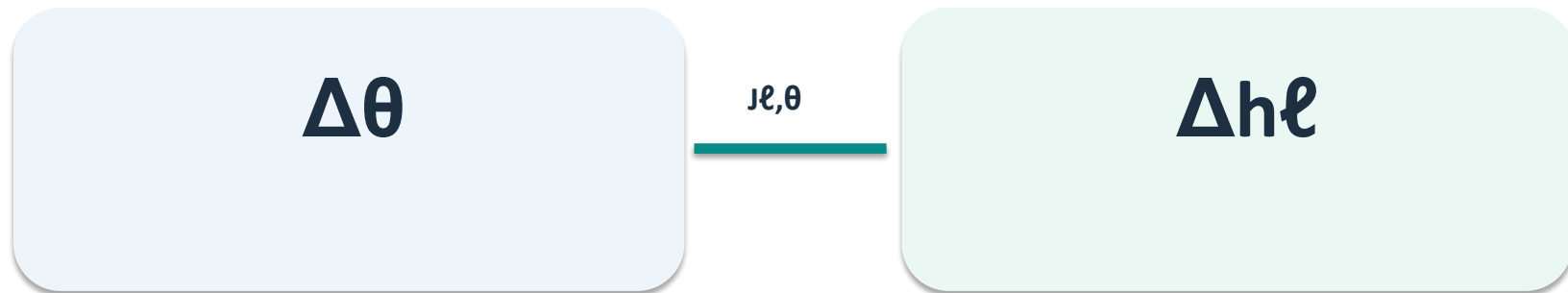
Маленькая окрестность — та, где градиенты разных задач могут складываться; большая — та, где сохраняется их приблизительная линейность

ГЛАВНЫЙ ВЫВОД

“Плоская” окрестность существует, но очень мала

Линейность не ведет к возможности смешивания задач

Pushforward: от $\Delta\theta$ к изменению активаций



$$\Delta h\ell(x) \approx J_{\ell, \theta}(x) \Delta\theta$$

Одинаковый параметрический сдвиг может давать разные сдвиги активаций на разных входах.

Pushforward: от обновлений весов к steering

Изменение параметров индуцирует изменение активаций через якобиан.

$$\Delta h_{\square}(x) \approx J_{\square, \theta}(x) \cdot \Delta \theta$$

Важное ограничение

Индукцированный сдвиг зависит от входа x и может не совпадать с одним постоянным вектором.

Практический вопрос

На каком слое низкоразмерное intervention приближает эффект fine-tuning?

Наш эксперимент: перенос обновлений в активации

Сопоставляем качество полного изменения весов, pushforward и непосредственно подобранного steering.

1 · Weight-space

$\Delta\theta$: fine-tuning или приближённое направление; исходный уровень качества.

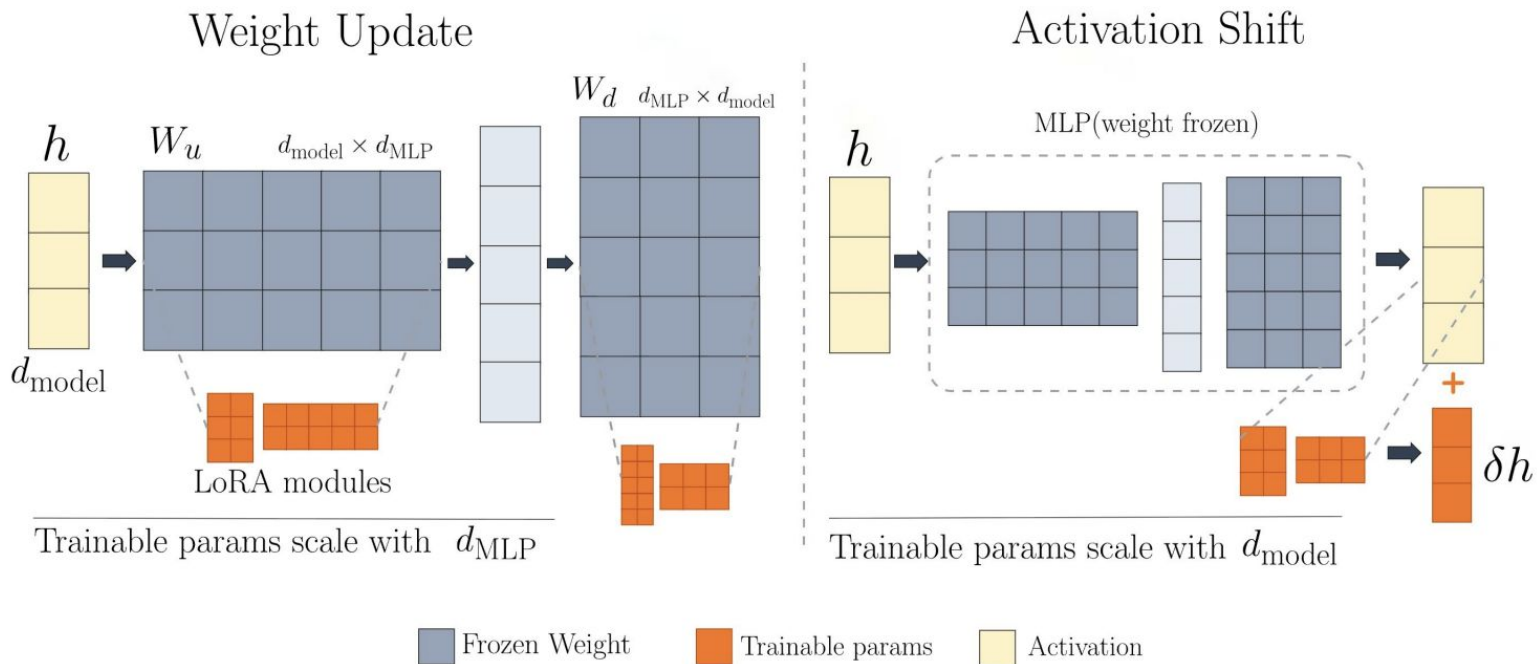
2 · Pushforward

$\Delta h_{\square}(x) = J_{\square, \theta}(x) \Delta\theta$;
зависимость от слоя и входа.

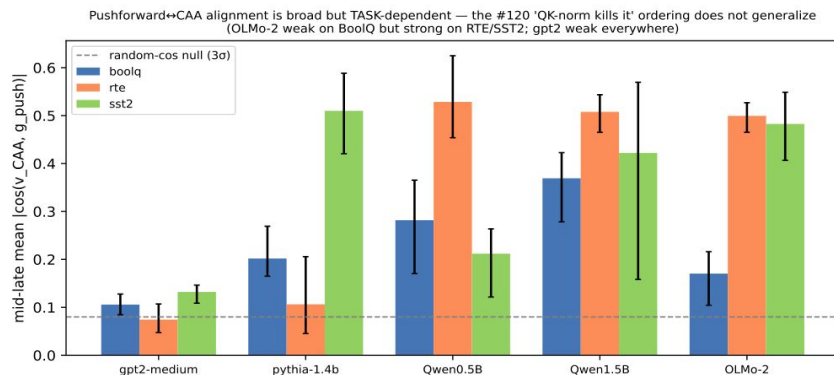
3 · Steering

Единый вектор или набор направлений; оценка воспроизведения эффекта.

Перенос обновлений в активации



Pushforward согласуется с САА — но не всегда



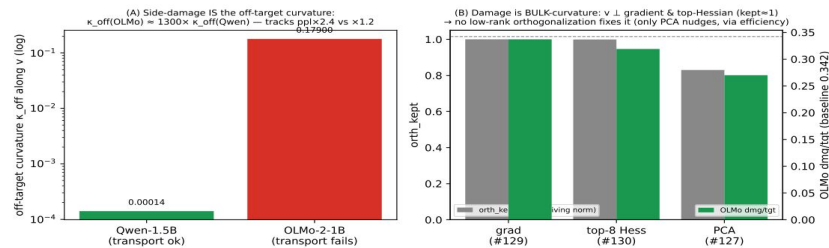
Косинус между pushforward градиентного шага и contrastive steering на BoolQ, RTE, SST-2. Серый пунктир — случайный baseline.

ГЛАВНЫЙ ВЫВОД

Соответствие проявляется в некоторых задачах и на некоторых слоях, обычно около $\frac{2}{3}$ сети

Порожденный стиринг часто имеет “побочные эффекты”, и по эффективности сравним со случайным

Steering может иметь высокую побочную цену

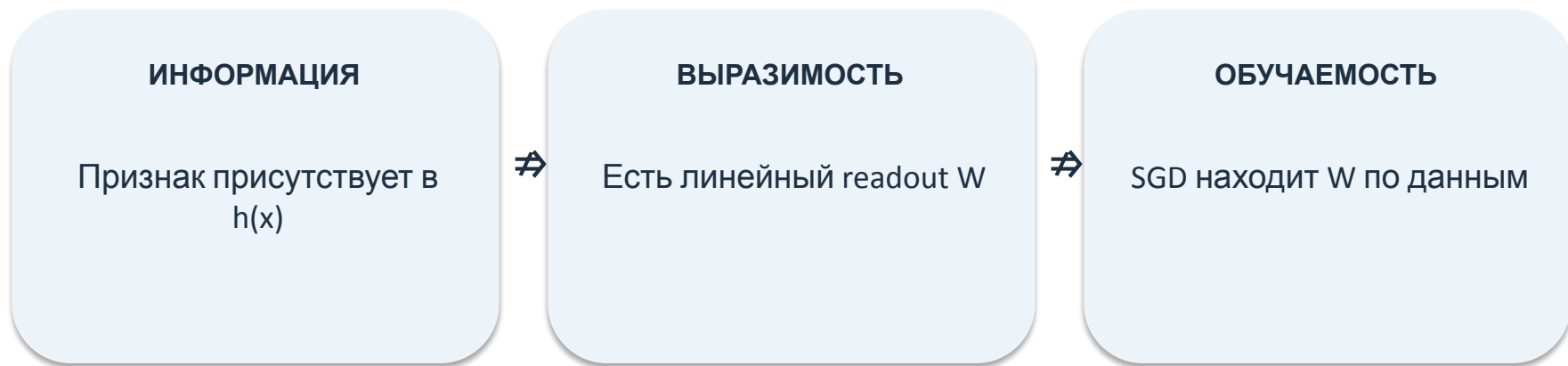


Кривизна off-target loss вдоль steering-направления у Qwen и OLMo; справа — доля сохранённого вектора после ортогонализации.

ГЛАВНЫЙ ВЫВОД

Из-за кривизны пространства трудно подобрать стиринг-направление, которое не несет побочного ущерба

Существование решения $\neq$ возможность его обучить



Линейный декодер может существовать, но его обучение на конечной выборке не обязано обобщаться.

Существование решения $\neq$ возможность его обучить

Разделим три свойства: information, expressivity и learnability.

Information

Нужная информация уже присутствует в представлении.

Expressivity

Простой readout способен её извлечь.

Learnability

Обычное обучение может найти readout и обобщить на новые данные.

Почему RandOpt иногда находит полезный сдвиг?

Хорошее направление должно существовать и занимать обнаруживаемую поиском область.

$$P(\text{success}) = 1 - (1 - p)^N$$

Наличие решения

Близость удачного checkpoint и локальная гладкость дают шанс адаптации, но не гарантию поиска.

Нахождение решения

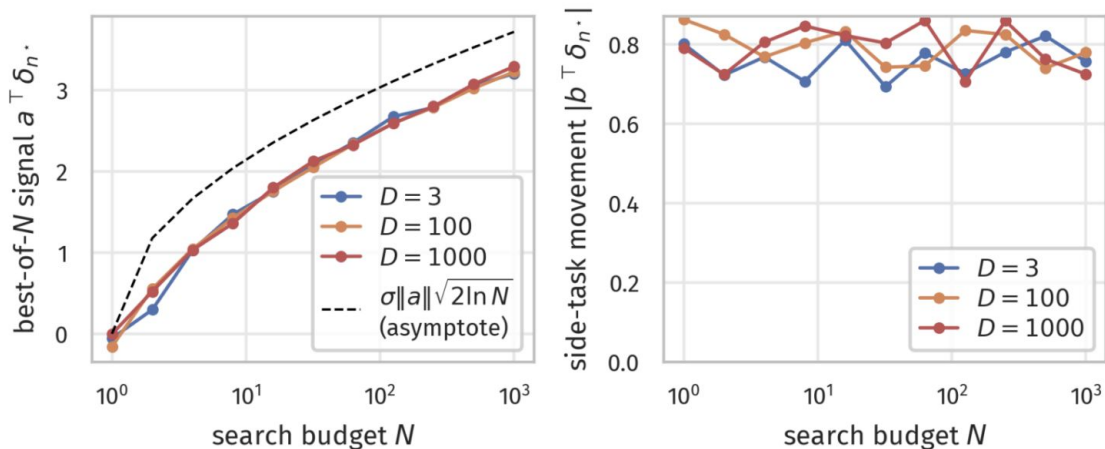
При очень малой доле успешных направлений p бюджет N растёт; помогает ограничение поискового подпространства.

Теорема (гауссовский best-of- в локально-линейной окрестности)

Рассмотрим чекпойнт и его окрестность, где приращение оценки ведёт себя линейно:

$$S(\theta + \delta) - S(\theta) = a^\top \delta + \mathcal{O}(|\delta|^2)$$

Тогда ожидаемый выигрыш best-of-N растёт с бюджетом (N), а средняя проекция на любую выбранную ортогональную ось асимптотически не растёт



Открытые вопросы: как выбирать минимальное вмешательство?

От эмпирических трюков — к предсказанию подходящего метода для конкретной задачи.

Что хотим предсказывать

Работает ли linear head? Нужен ли steering?
Какой слой, масштаб, ранг и бюджет поиска?

Что нужно измерять

Геометрия градиентов; локальная линейность;
pushforward; устойчивость к входам и качество
обобщения.

Цель: минимальное вмешательство с прогнозируемым качеством