Отчет по заданию LLM Foundations в Tinkoff Research

Муравецкий Даниил, МФТИ, ex-Sber AI Lab

Telegram: @dvachewski

1 Введение

В данном отчете приведены результаты экспериментов с кодом из моего репозитория Looped Transformer. Все эксперименты были выполнены на GPU Kaggle Notebooks и Google Colab, результаты логировались на Weights&Biases. Для всех экспериментов был использован синтетический датасет Linear Regression Task из исходной статьи.

2 Имплементация исходной статьи

Мною была поставлена серия экспериментов по запуску бейзлайна из Looped Transformer и Universal Transformer (Vanilla GPT 2),и были получены следующие результаты Рис. 1. Однако не смотря на то что модель обучалась(лосс уменьшался) и MSE на валидационной и тестовой выборке уменьшились, на тесте график не совпал с тем, что было представлено в оригинальной статье. Попытки увеличить количество эпох до 500k дали также отличающиеся результаты. Рис. 2

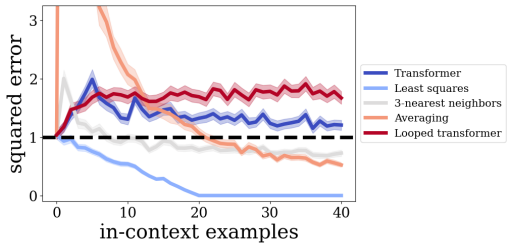

Рис. 1: Looped Transformer и Universal Transformer, 150k эпох, дефолтные параметры конфига у Looped Transformer: b=350, T=20

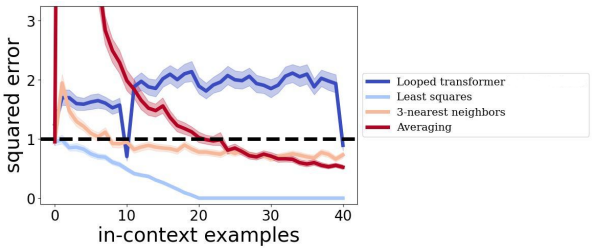

Рис. 2: Looped Transformer 500k эпох, дефолтные параметры конфига у Looped Transformer: b=350, T=20

После эксперимента с теми параметрами, которые предлагались в самой статье как оптимальные $b = 20, T = 15$ получились результаты хуже чем на Рис. 1. У меня возникло предположение, что так получается из-за того, что даже 500k, а уж тем более 150k эпох недостаточно для эффективного обучения моделей с $dim = 20$ у регрессии, поэтому было принято решение уменьшить размерность до $dim = 4$ и сравнивать последующие эксперименты с ним. И после уменьшения размерности графики на тесте стали наконец напоминать графики из исходной статьи Рис. 5.

Так как из-за лимита бесплатных GPU-часов на Kaggle, я не могу себе позволить тренировать модели более 150k эпох и было установлено, что после 5-10k эпох наблюдается медленная сходимостьРис. 3, было принято решение тренировать модели на 150k эпохах.

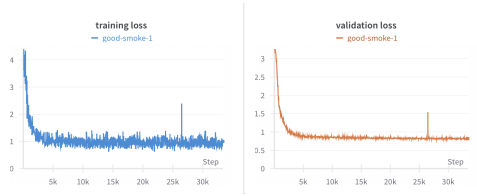

Рис. 3: Train и Validation loss Universal Transformer, обучение на ргегрессии с dim=4

3 Добавление N последних токенов

Была проверена гипотеза добавления только N последних токенов output с предыдущего шага у Looped Transformer. При тренировке на регрессии с $dim = 20$ получился следующий результат Рис. 4. Где трансформер с 22 токенами - это Universal Transfomer, так как размерность output была равна 22 токенам.

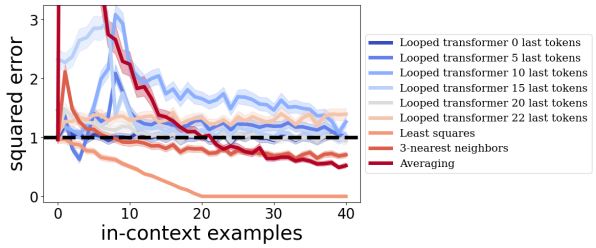

Рис. 4: Looped Transformer, 150k, эпох, с разным количеством токенов в output, дефолтные параметры конфига у Looped Transformer: b=350, T=20. У регрессии dim=20

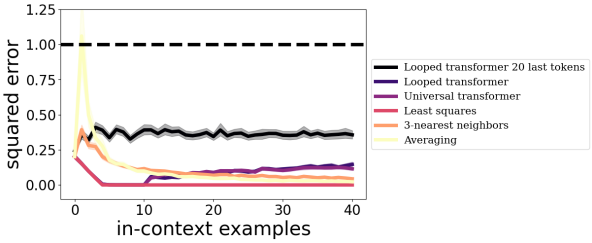

Рис. 5: Looped Transformer 20 last tokens дефолтные параметры конфига у Looped Transformer: b=350, T=20. У регрессии dim=4

Исходя из данных можно понять, что при недолгом обучении на небольшом количестве эпох или при сложных данных конкатенация только части output может быть лучше наших бенчмарков Рис. 4, между тем, при полном обучении использование только N последних токенов даёт результаты хуже, чем исходный трансформер Рис. 5

Так же можно заметить, что при размерности 4 модели сразу выдают показатели лучше при размерности 20 из-за меньшей сложности данных.

4 A→B Слой

Была предложена идея законнектить два looped-слоя трансформера между собой.

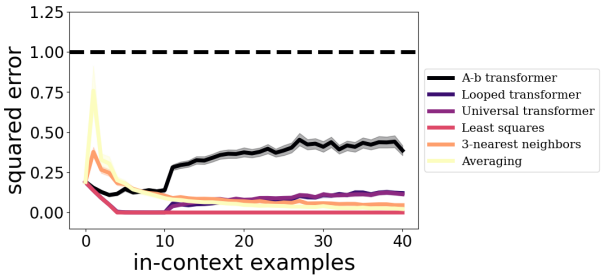

Рис. 6: Сравнение моделей с N Last Tokens и Vanilla Looped Transformer(22 tokens), У регрессии dim=4

Как мы можем видеть, при небольшой длине контекста такой трансформер показывает неплохой результат, обгоняющий некоторые бенчмарки, однако при большой длине контекста, такого не происходит Рис. 6.

5 SSM-слой вместо Attention

Также ещё одной идеей было вставить SSM-слой вместо классического Attention, так как иногда они показывают результаты лучше классического трансформера.

Получилось так, что при стандартных 150k эпохах модель с SSM-слоём обучалась сильно более 12 часов(при 12 часах обучилась только на 62209 эпох), что намного дольше других моделей, и пришлось уменьшить кол-во эпох до 40k. И на инфернсе после 5 часов работы почему-то выдала везде Nan. Из-за этого ещё и был сдан отчет с небольшим опозданием. Следует детальнее изучить этот вопрос

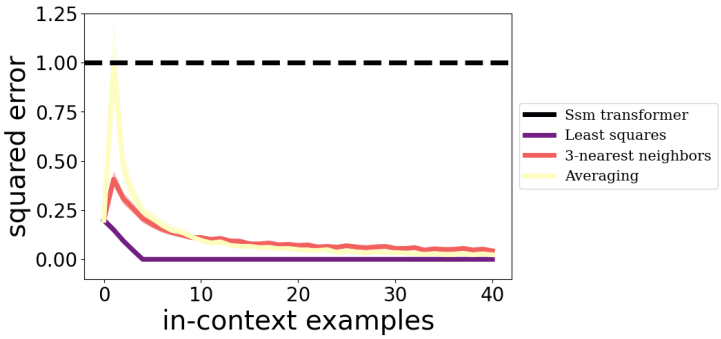

Рис. 7: Transformer с SSM-layer, дефолтные параметры, У регрессии dim=4

6 Bonus: Попытки улучшить результаты статьи

Одним из возможных улучшений Looped Transformer может служить замена простого суммирования input и output суммированием методом скользящего среднего:

$$out_t = out_{t-1} \cdot \alpha + inp \cdot (1 - \alpha), \forall \alpha \in [0, 1] \tag{1}$$

Так же, исходя из того что данные имеют не нормальное распределение, а распределены как $f(x) \sim \text{Uniform}(\mathbb{R}), \quad \forall x \in \mathbb{R}$, логичнее было бы использовать MAE в качестве лосса вместо MSE как в оргинальной работе. После дальнейших экспериментов с различными функциями потерь оказалось, что лучшим является HuberLoss - сглаженная модификация MAE Рис. 9

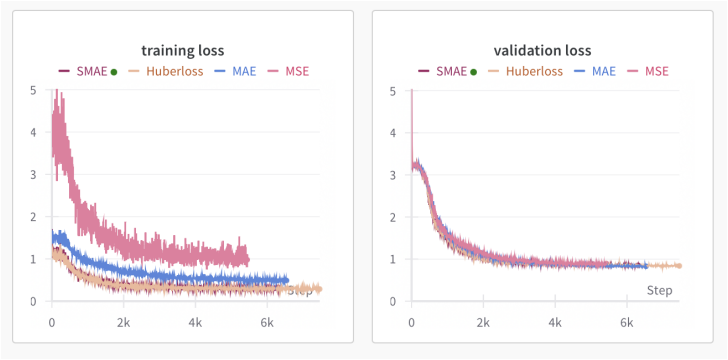

Рис. 8: Сравнение различных лоссов, модель - Universal Trasformer, reg. dim=4,validation loss унифицирован для всех моделей как среднеквадратическое отклонение(MSE)

Кроме того были эксперименты с различными видами Gradient Clipping, однако они не дали удовлетворительных результатов при обучении. Также можно было бы попробовать в формуле (1) обернуть правую часть в нелинейную функцию типа tanh или sigmoid, попробовать туда прикрутить LSTM/GRU или сделать аугментацию данных с помощью гауссовского шума $X = X + \tilde{X} | \tilde{X} \sim \mathcal{N}(\mu, \sigma^2)$, однако у меня больше нет GPU-часов :(

7 Несколько слов для уважаемого проверяющего

Уважаемые исследователи Tinkoff Research(T-Lab), надеюсь, что Вам понравился мой отчет и мои эксперименты, ведь я их долго делал и очень старался :3 Много чего из неудачных ресерческих идей я не включил в отчет, однако, мне кажется, по итогу вышло довольно неплохо. Очень надеюсь на то, что Вы по достоинству оцените мой труд и мой бэкграунд, и пригласите меня к себе =) Я верю, что мои знания и Ваш опыт положат начало нашему дальнейшему плодотворному сотрудничеству!
Спасибо за внимание! ☺
P.S.
Мои контакты есть в шапке отчета и в CV