Pixelated Butterfly

Бобков Денис 192

Чем вообще занимаемся?

- Работаем с GEMM нейросетями
- Хотим ускорить умножение матриц за счёт sparse структуры
- В начале работы для каждой матрицы весов находим свой паттерн и обучаем только ненулевые веса

Паттерн - правило, по которому выбираем позиции для ненулевых элементов

Другие методы

Есть проблемы:

1) Статические методы - плохое качество, динамические - долго
2) Найденная маска не всегда параллелится
3) Хотим чтобы метод работал с разными типами операций (attention, MLP)

Предлагается метод, решающий всё это

Общая схема

1) Распределяем бюджет по слоям модели
2) Для каждой матрицы весов предсказываем sparse маску (паттерн)
3) Инициализируем также обучаемую малоранговую добавку в виде U и V

Получаем представление матрицы весов вида $W = \gamma B + (1 - \gamma)\dot{U}\dot{V}^\top$

B - sparse матрица

$\dot{U}\dot{V}^\top$ - малоранговая добавка

γ - обучаемый параметр

Распределение бюджета

- Считаем, что доступ к блоку размера b равносилен доступу 1 элемента
- Будем распределять бюджет прямо пропорционально стоимости слоёв

Как вычислить стоимость?

Распределение бюджета

- Считаем, что доступ к блоку размера b равносилен доступу 1 элемента
- Будем распределять бюджет прямо пропорционально стоимости слоёв

Модель бюджета для 1 слоя: $\mathrm{Totalcost} = \mathrm{Cost}_{\mathrm{mem}} \cdot N_{\mathrm{blockmem}} + \mathrm{Cost}_{\mathrm{flop}} \cdot N_{\mathrm{flop}}.$

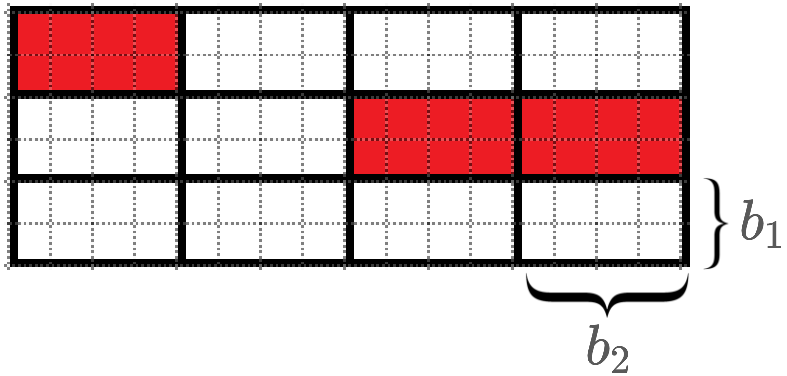
Cost - стоимость операции

N - количество операций

Замечание

Важно использовать (b, b) block-aligned патерны в масках

Пример (b_1, b_2) - aligned маски:



1) Разбиваем матрицу на блоки размера (b_1, b_2)
2) Каждый блок должен состоять только из 1, либо только из 0

Выбор паттерна: теория

Ранее был изучен класс butterfly матриц, но они:

1) Не block-align -> неэффективные
2) Сложно параллелятся, т.к. состоят из произведения нескольких матриц
3) Имеют большой ранг -> плохо работаю с матрицами низкого ранга

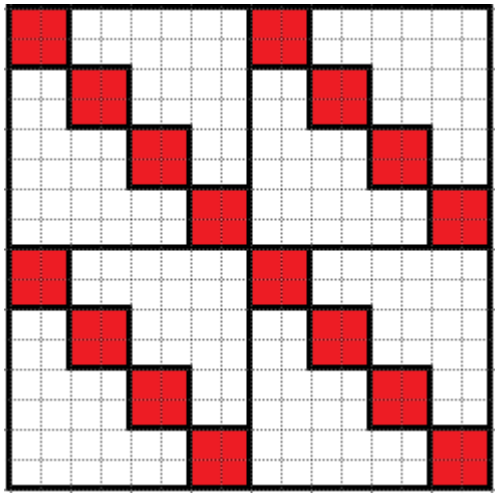
Выбор паттерна: определения

Block butterfly factor $B_{k,b}$ размера kb с размером блока b это матрица вида $\mathbf{B}_{k,b} = \begin{bmatrix} \mathbf{D}_1 & \mathbf{D}_2 \\ \mathbf{D}_3 & \mathbf{D}_4 \end{bmatrix}$, где $\mathbf{D}_i$ это блочно-диагональная матрица размера $\frac{k}{2} \times \frac{k}{2}$

Пример:

b = 2

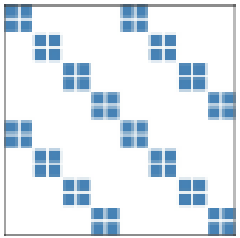
k = 8



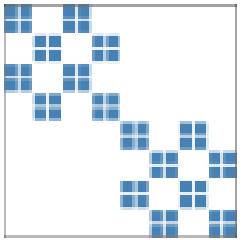
Выбор паттерна: определения

Block butterfly factor matrix $\mathbf{B}_k^{(n,b)}$ размера nb с размером блока b и страйдом k это блочно-диагональная матрица вида $\mathbf{B}_k^{(n,b)} = \operatorname{diag}\left([\mathbf{B}_{k,b}]_1, [\mathbf{B}_{k,b}]_2, \ldots, [\mathbf{B}_{k,b}]_{\frac{n}{k}}\right)$

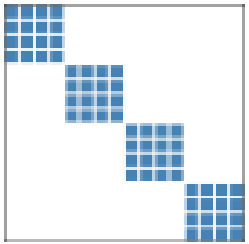
с n/k блоками.



$B_8^{(8,2)}$



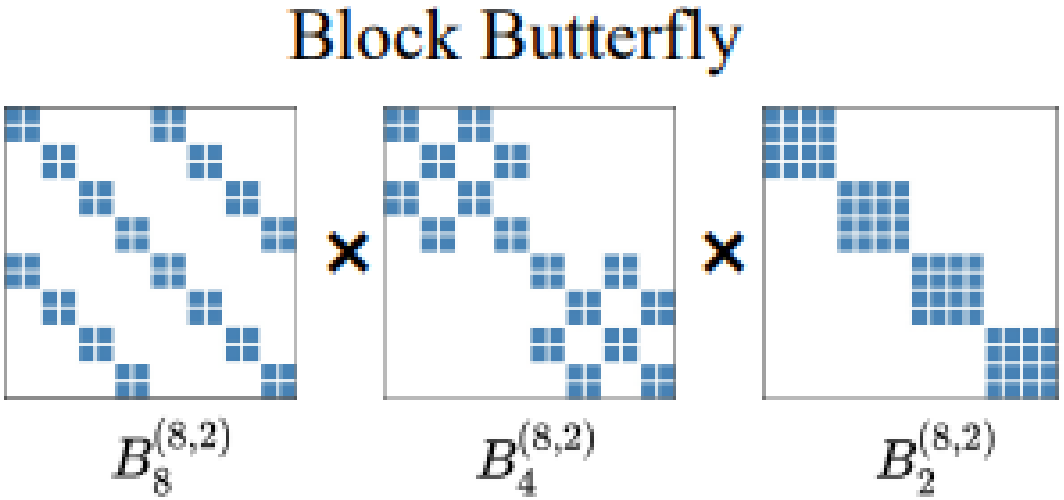
$B_4^{(8,2)}$



$B_2^{(8,2)}$

Выбор паттерна: определения

Block butterfly matrix $\mathbf{B}^{(n,b)}$ размера nb с размером блока b это матрица представленная в виде произведения b.b. factor matrix: $\mathbf{B}^{(n,b)} = \mathbf{B}_n^{(n,b)} \mathbf{B}_{\frac{n}{2}}^{(n,b)} \dots \mathbf{B}_2^{(n,b)}$



Выбор паттерна: Flat

Хотим избавиться от произведения

Показали, что если M - b.b. matrix, то для некоторого $\lambda \in \mathbb{R}$:

$$M = (I + \lambda \mathbf{B}_n^{(n)})(I + \lambda \mathbf{B}_{n/2}^{(n)}) \dots (I + \lambda \mathbf{B}_2^{(n)})$$

Т.е. если λ маленькая, то можем представить:

$$M = I + \lambda(\mathbf{B}_2^{(n)} + \mathbf{B}_4^{(n)} + \dots + \mathbf{B}_n^{(n)}) + \widetilde{O}(\lambda^2).$$

На практике просто берётся сумма из k b.b. matrix, где k зависит от бюджета

Общая схема (опять)

1) Распределяем бюджет по слоям модели
2) Для каждой матрицы весов инициализируем k flat b.b. matrices, будем брать B как их сумма.

Проблема:

B будет большого ранга, поэтому сделаем малоранговую добавку

Получаем представление матрицы весов вида $W = \gamma B + (1 - \gamma)U\dot{V}^\top$

Полная схема

1) Распределяем бюджет по слоям модели прямо пропорционально стоимости слоёв
2) От ⅓ до ¼ бюджета отводится для малорангового разложения, на основании этого вычисляется ранг
3) Для каждой матрицы весов инициализируем k flat b.b. matrices, будем брать B как их сумма, k высчитывается из оставшегося бюджета

   Получаем представление матрицы весов вида $W = \gamma B + (1 - \gamma) U V^\top$

4) Обучаем

Результаты CV

Model	CIFAR10	CIFAR100	ImageNet	Speedup
Mixer-S/16	86.4	58.7	72.4	-
Pixelfly-Mixer-S/16	89.8	62.9	72.6	1.7×
Mixer-B/16	87.6	59.5	75.6	-
Pixelfly-Mixer-B/16	90.6	65.4	76.3	2.3×
ViT-S/16	89.5	65.1	77.7	-
Pixelfly-ViT-S/16	91.3	66.8	77.5	1.9×
ViT-B/16	89.9	61.9	78.5	-
Pixelfly-ViT-B/16	92.2	65.1	78.6	2.0×

Измеряют: accuracy

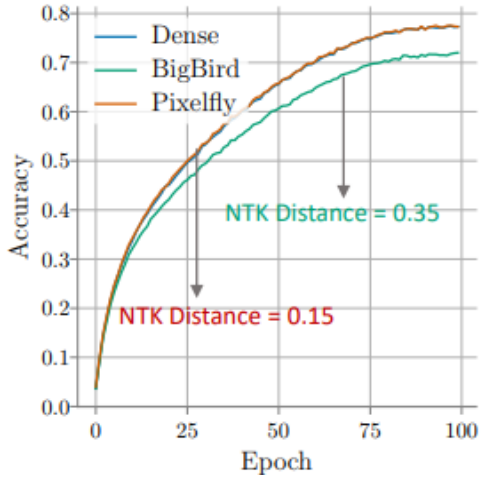
Speedup считают на ImageNet

Результаты CV

Model	ImageNet top-1 acc.	Speedup	Params	FLOPs
Mixer-B/16	75.6	-	59.9M	12.6G
Butterfly-Mixer-B/16	76.1	0.8×	17.4M	4.3G
Pixelfly-Mixer-B/16	76.3	2.3×	17.4M	4.3G

Результаты CV

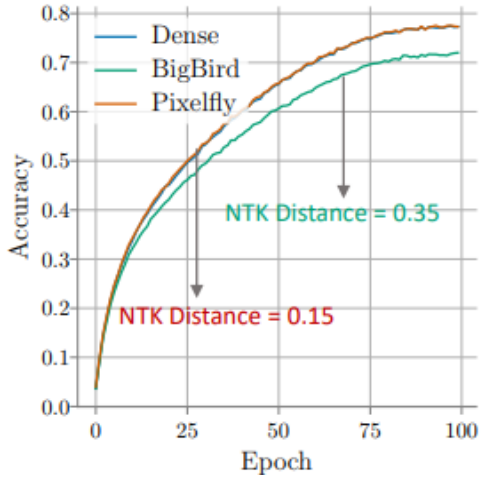
Model	ImageNet top-1 acc.	Speedup	Params	FLOPs
Mixer-B/16	75.6	-	59.9M	12.6G
Butterfly-Mixer-B/16	76.1	0.8×	17.4M	4.3G
Pixelfly-Mixer-B/16	76.3	2.3×	17.4M	4.3G



- Измеряют NTK
- Кривая обучения Pixelfly сильно ближе к Dense
- Обучают VIT на CIFAR 100

Результаты CV

Model	ImageNet top-1 acc.	Speedup	Params	FLOPs
Mixer-B/16	75.6	-	59.9M	12.6G
Butterfly-Mixer-B/16	76.1	0.8×	17.4M	4.3G
Pixelfly-Mixer-B/16	76.3	2.3×	17.4M	4.3G



Model	ImageNet (Acc)	Speedup
T2T-ViT	81.7	-
BigBird	81.5	0.9×
Sparse Transformer	81.4	1.3×
Pixelfly	81.7	1.4×

Результаты NLP

Model	WikiText-103 (ppl)	Speedup	Params	FLOPS
GPT-2-Small	22.2	-	117M	48.4G
BigBird	23.3	0.96×	117M	40.2G
Pixelfly	22.5	2.1×	68M	18.5G
GPT-2-Medium	20.9	-	345 M	168G
BigBird	21.5	1.1×	345 M	134G
Pixelfly	21.0	2.5×	203M	27G

Задача: Language modeling

Метрика: perplexity (меньше-лучше)

Результаты NLP

Model	ListOps	Text	Retrieval	Image	Pathfinder	Avg	Speedup
Transformer	36.54	63.12	80.33	41.56	**73.49**	59.01	-
Reformer	36.85	58.12	78.36	28.30	67.95	53.90	0.8×
Pixelfly	**37.65**	**66.78**	**80.55**	**42.35**	72.01	**59.86**	5.2×

Использовали LRA - набор бенчмарков для трансформеров

Метрика: accuracy

Спасибо за внимание

Статья: https://arxiv.org/pdf/2112.00029.pdf