

Спарсификация и дистилляция

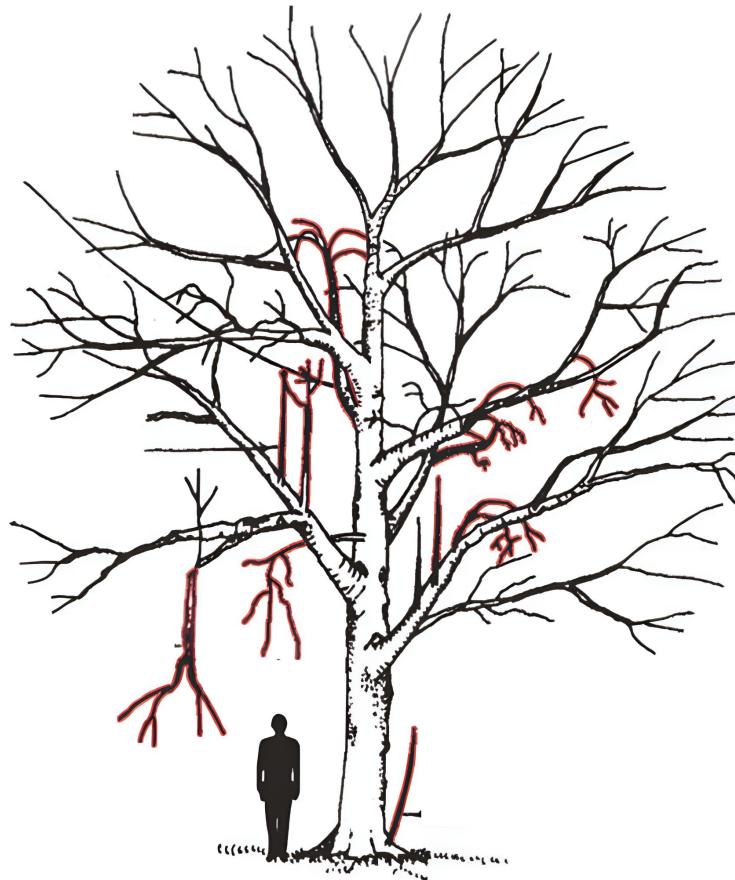
Дмитрий Осин @xaosina

Содержание:

1. Мотивация
2. Когда обрезать
3. Низкоуровневая точка зрения. Структурный и неструктурный прунинги
4. Как выбрать важность веса?
5. Методы с дообучением с использованием данных
6. OBD, OBS
7. Динамическая спарсификация
8. Дистилляция

>>>

Мотивация



На деревьях бывает много мертвых ветвей без листьев и плодов. Они сильно утяжеляют дерево. Поэтому мертвые ветки можно безопасно срезать, не затрагивая ветки, которым полезно дополнительное пространство

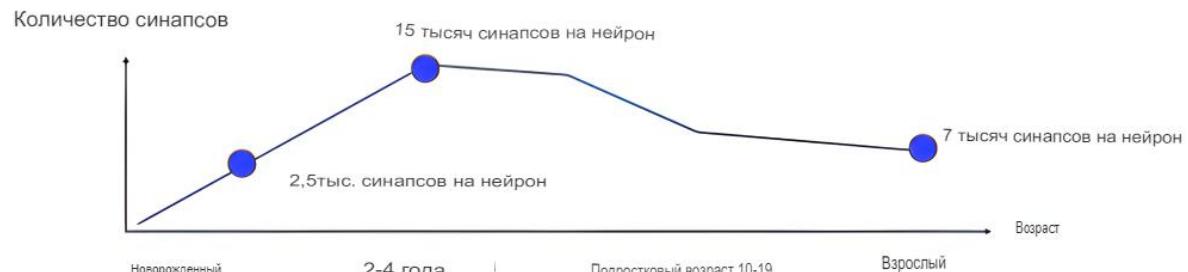


До



После

Обрезать или удалять нейронные связи, которые мы не используем — оптимизация из реальной жизни.



Pruning (прунинг) — процесс избавления от ненужных параметров сети для ее ускорения.

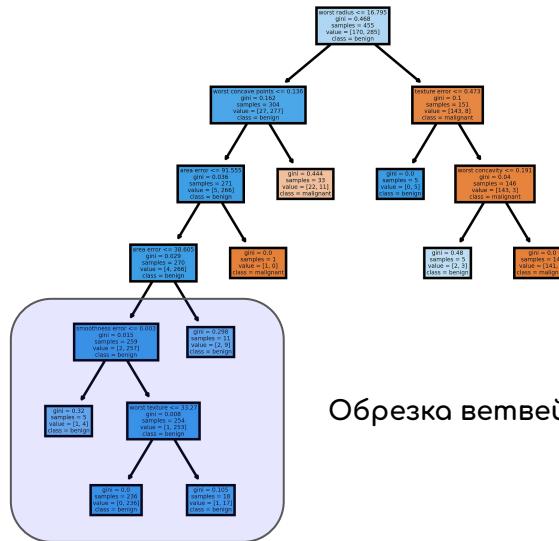
Sparsity (разреженность) — высокое содержание нулевых весов или нейронов в сети.

Прунинг приводит к разреженности.

Иногда вместо прунинга могут сказать «спарсификация».

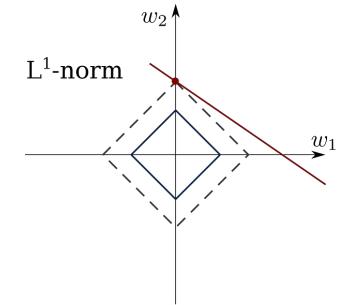
Идея не нова

Деревья решений



Лассо-регрессия

$$\sum_{i=1}^n (Y_i - \sum_{j=1}^p X_{ij}\beta_j)^2 + \lambda \sum_{j=1}^p |\beta_j|$$



Robust Principal Component Analysis — RPCA

$$W = S + L$$

S — разреженная;

L — низкоранговая.

Почему это работает и почему нельзя с самого
начала обучить меньшую модель?

Модели
оверпараметризованы



Случайная
инициализация



Динамика обучения
(оптимизации)
SGD — всего лишь первого
порядка

Размер данных и
их качество

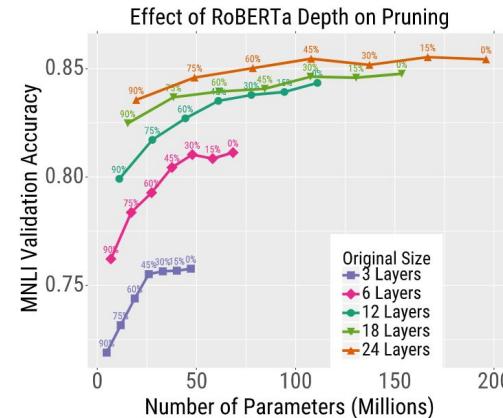
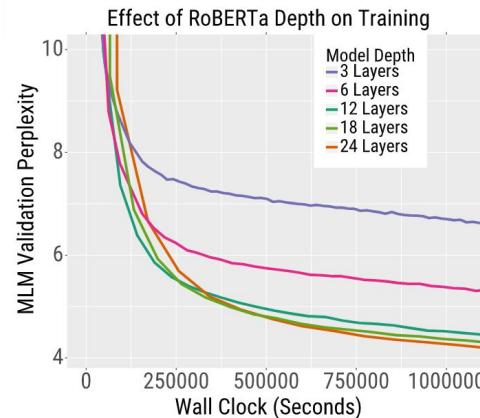
THE LOTTERY TICKET HYPOTHESIS: FINDING SPARSE, TRAINABLE NEURAL NETWORKS

Гипотеза статьи: внутри большой модели есть меньшая модель, которая, если ее обучать самостоятельно, будет соответствовать производительности более крупной модели



[Статья](#)

Train Large, Then Compress: Rethinking Model Size for Efficient Training and Inference of Transformers



Большие модели сходятся лучше и быстрее, а также лучше переносят прунинг. Даже при одинаковом числе параметров.

[Источник](#)

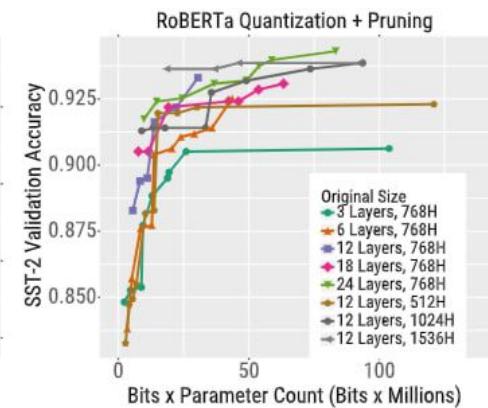
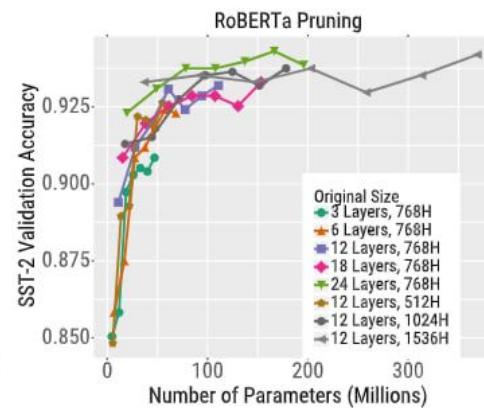
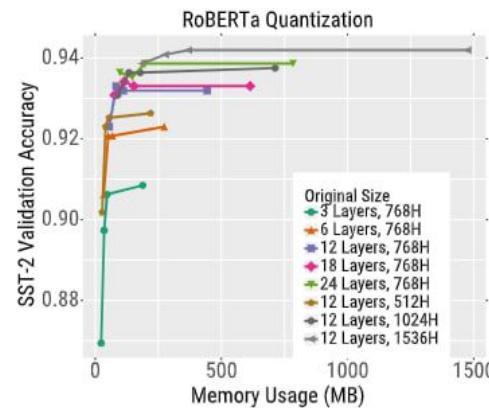
Что лучше?

Сжимать большие
модели

Обучать
маленькие

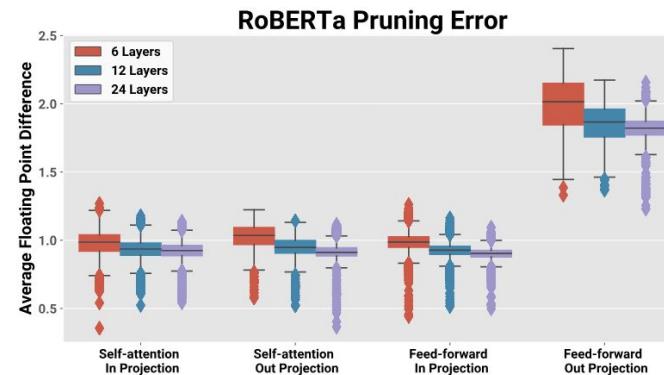
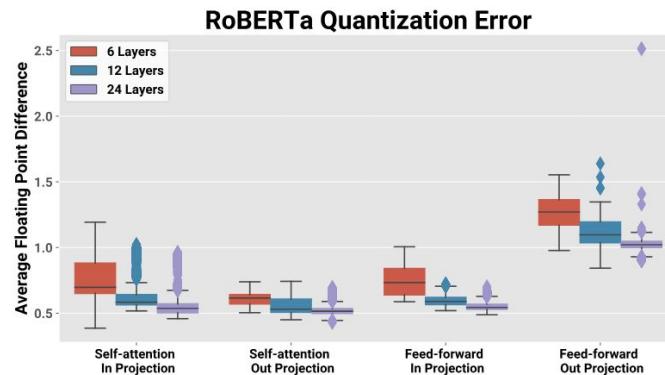


Train Large, Then Compress: Rethinking Model Size for Efficient Training and Inference of Transformers



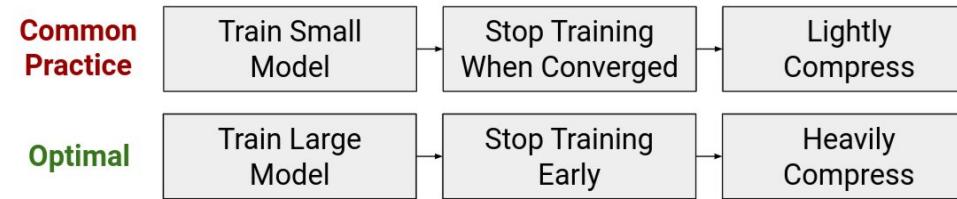
[Источник](#)

Train Large, Then Compress: Rethinking Model Size for Efficient Training and Inference of Transformers



[Источник](#)

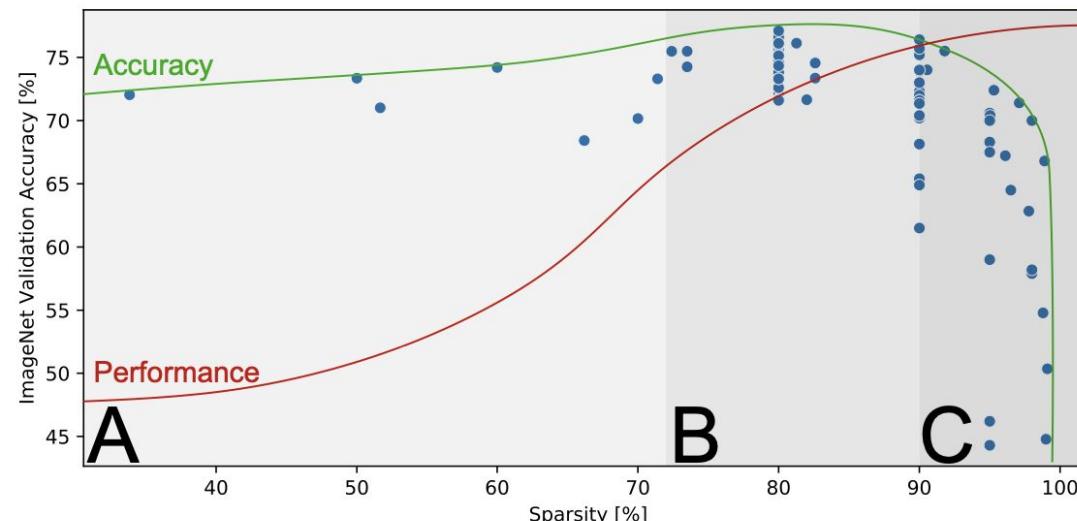
Train Large, Then Compress: Rethinking Model Size for Efficient Training and Inference of Transformers



[Источник](#)

Resnet 50

Иногда спарсификация улучшает качество



[Источник](#)

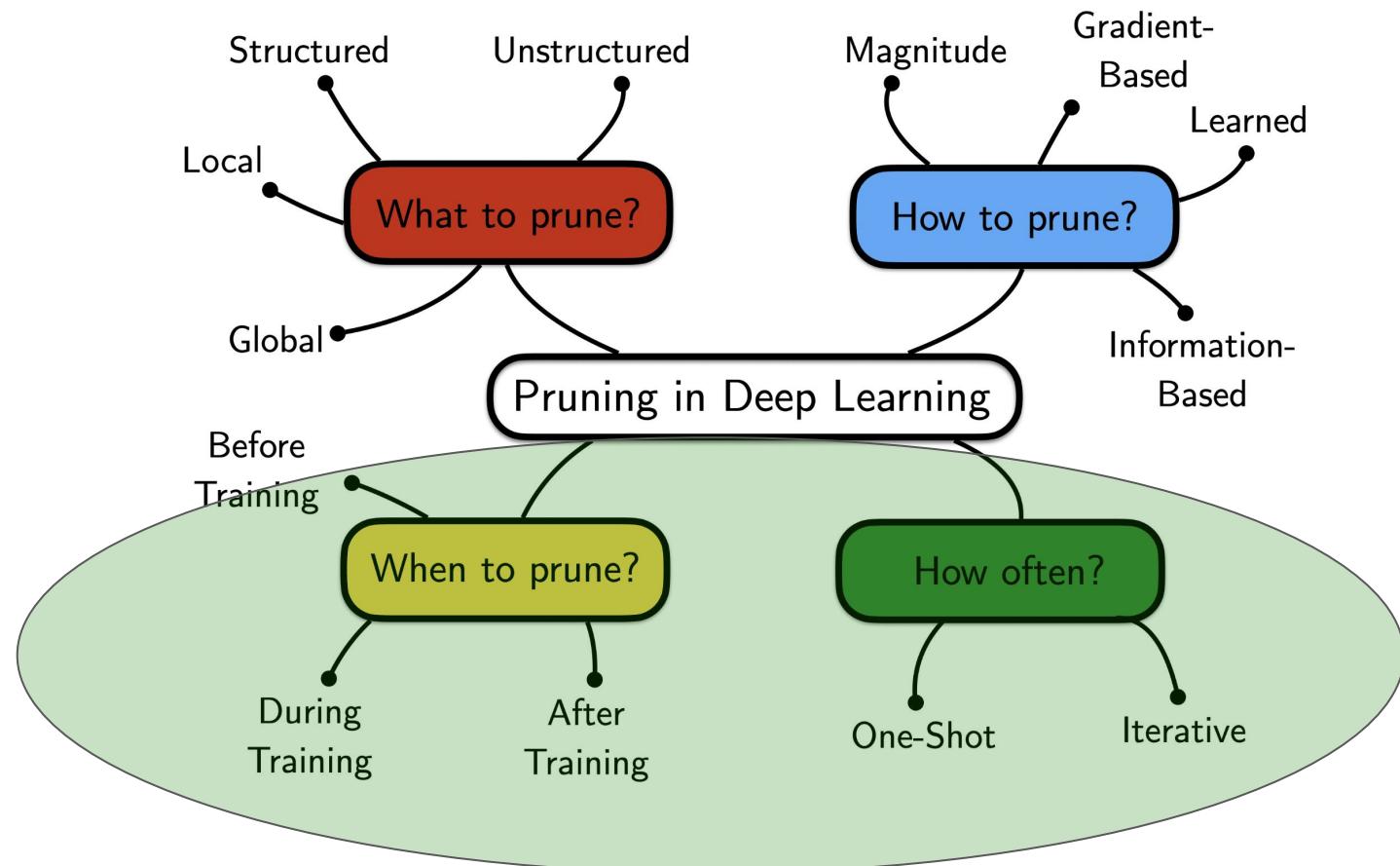
Заключение

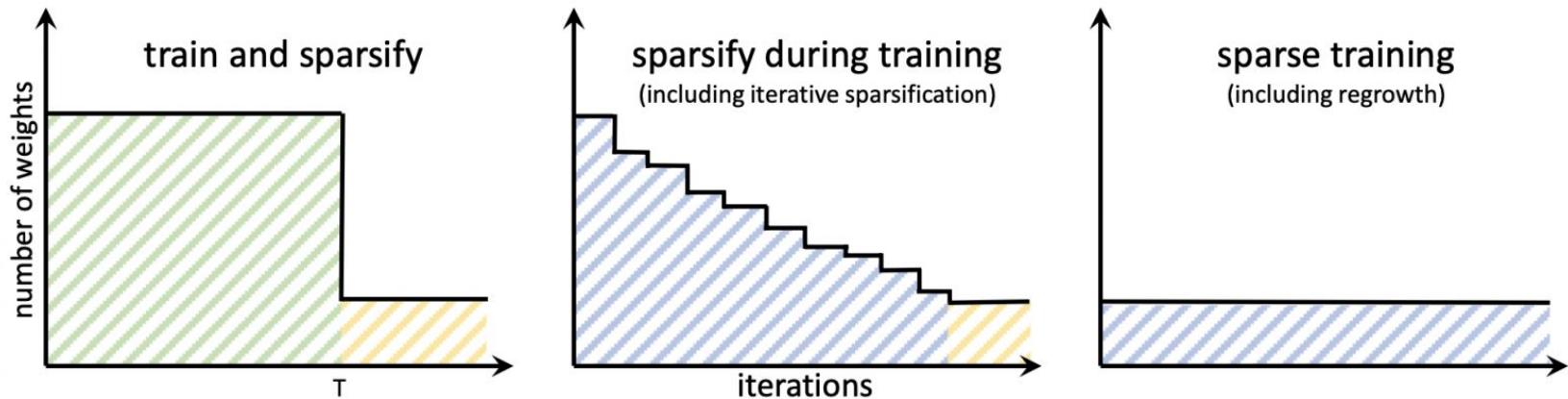
1. Современные нейронные сети избыточны в своих весах.
2. Благодаря этому можно значительно сжимать модели с небольшим падением качества.
3. Проще обрезать большие модели, чем маленькие.

Заключение

1. Современные нейронные сети избыточны в своих весах.
2. Благодаря этому можно значительно сжимать модели с небольшим падением качества.
3. **Проще обрезать большие модели, чем маленькие.**

>>> Когда обрезать





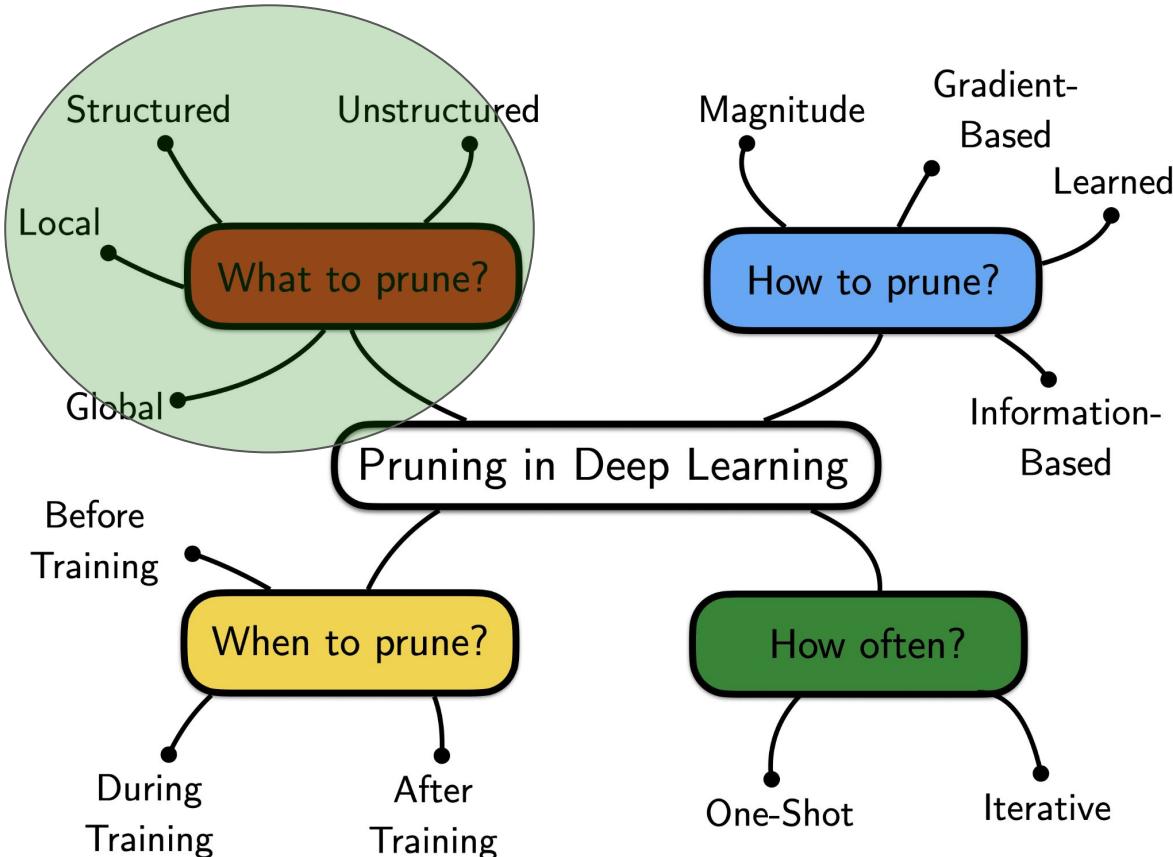
[Источник](#)

Примеры:

1. [SNIP: Single-shot Network Pruning based on Connection Sensitivity](#)
2. [Scalable training of artificial neural networks with adaptive sparse connectivity inspired by network science](#)
3. [Rigging the Lottery: Making All Tickets Winners](#)
4. [Picking Winning Tickets Before Training by Preserving Gradient Flow](#)

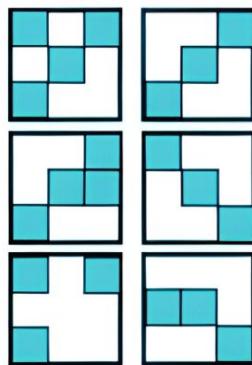
Заключение

Есть множество стратегий спарсификации в плане того, как сильно и когда обрезать модель.

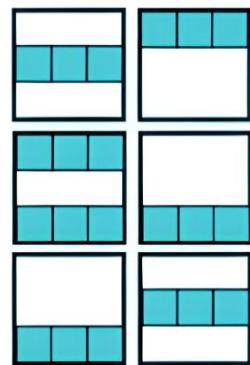


>>> Низкоуровневая точка зрения.
Структурный и
неструктурный прунинги

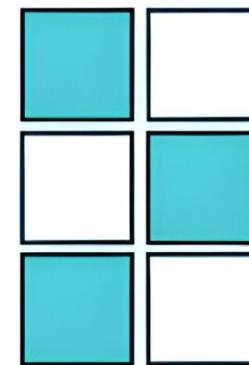
Виды пруニングа



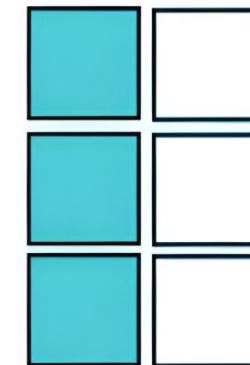
Неструктурный
прунинг



Vector level
1-D



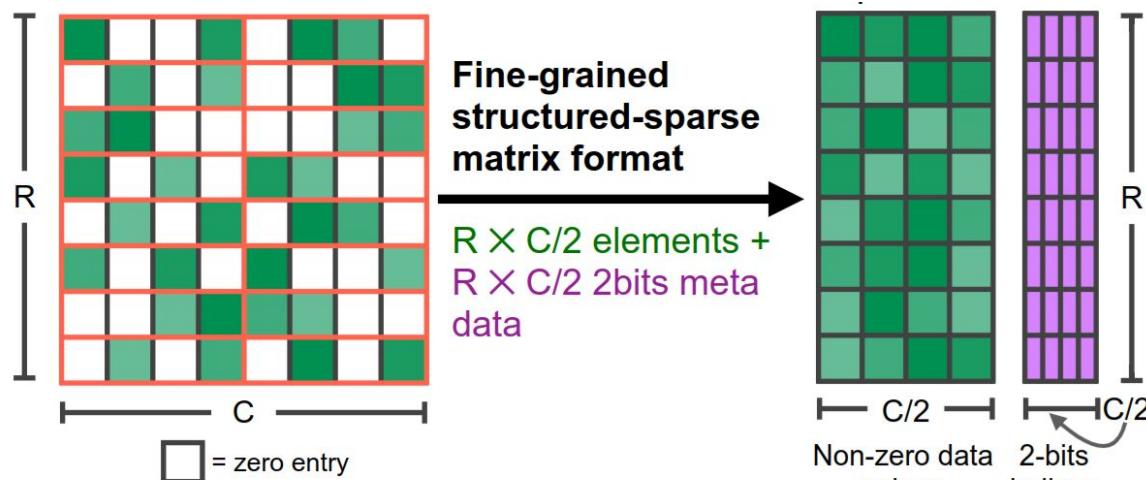
Kernel level
2-D



Filter level
2-D

N:M-разреженность — это структурированный шаблон разреженности, который хорошо работает с современной аппаратной оптимизацией графического процессора, в которой N из каждого M последовательных элементов являются нулями.

Эффективность сжатого формата: использование формата CSR для неструктурированной разреженности может привести к увеличению накладных расходов на хранение из-за метаданных до 200 %.

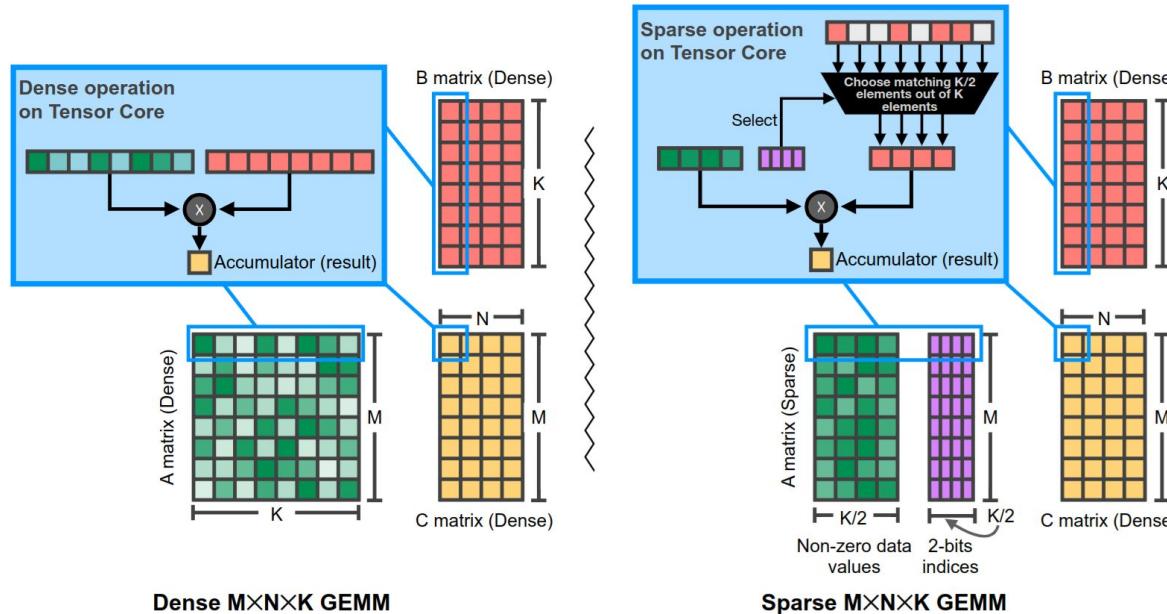


[Источник](#)

2:4 разреженность

~ 50% уменьшение весов

~ 2x ускорение математических вычислений



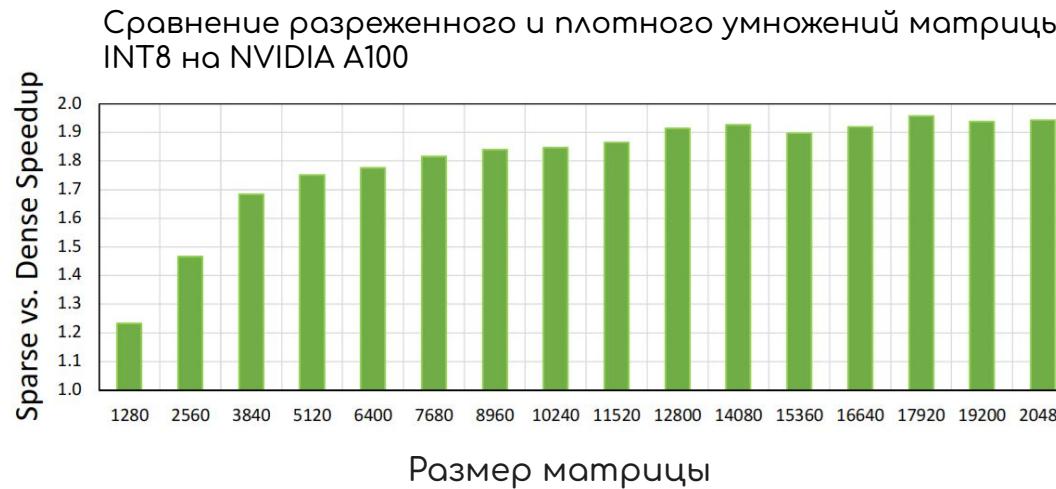
В настоящее время поддерживается только с:
FP32, FP16, int8.

[Источник](#)

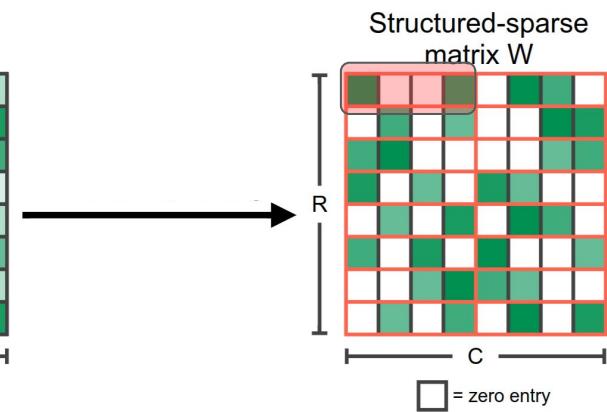
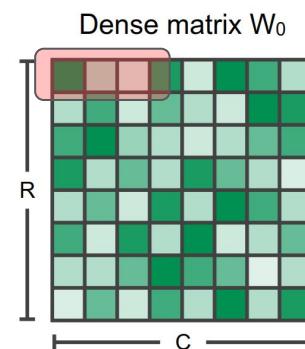
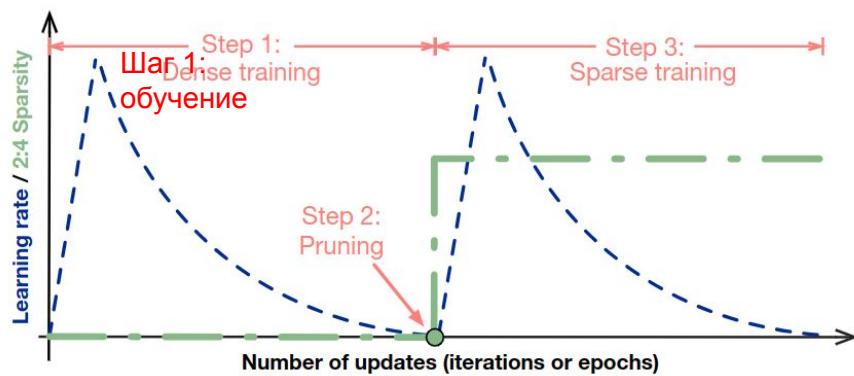
2:4 разреженность

~ 50% уменьшение весов

~ 2x ускорение математических вычислений

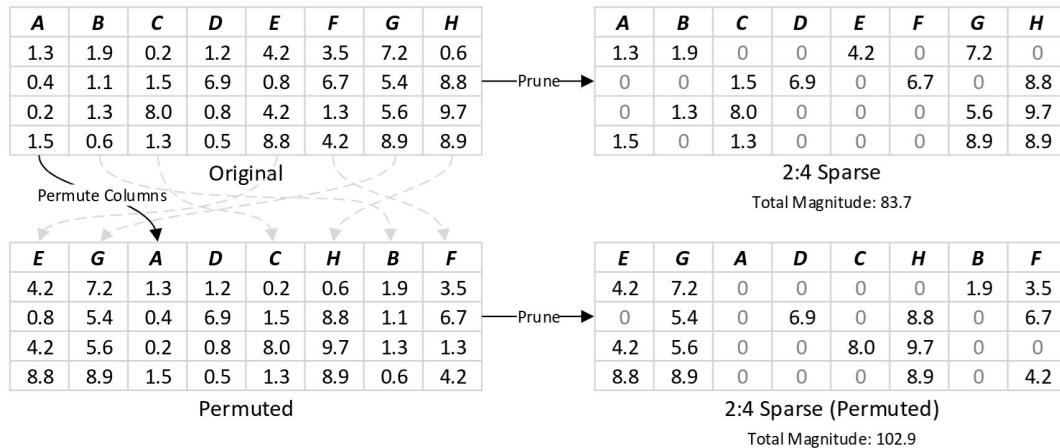


[Источник](#)



Удалить два значения с наименьшей
мagnитудой (значением)
в каждой ячейке.

[Источник](#)



Переставить столбцы, чтобы сохранить более важные веса с более высокой величиной.

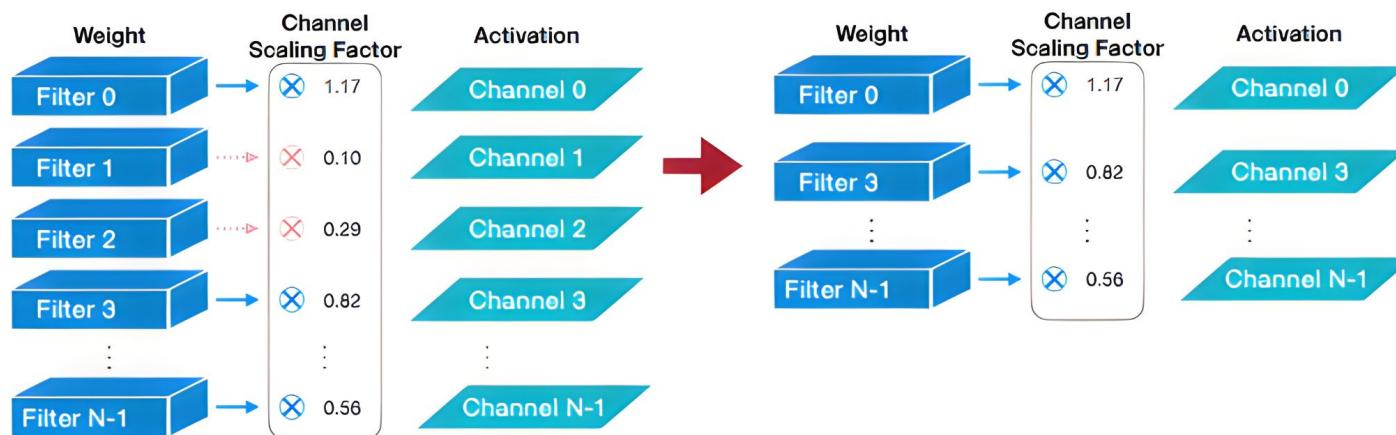
Оптимизация для увеличения общей магнитуды.

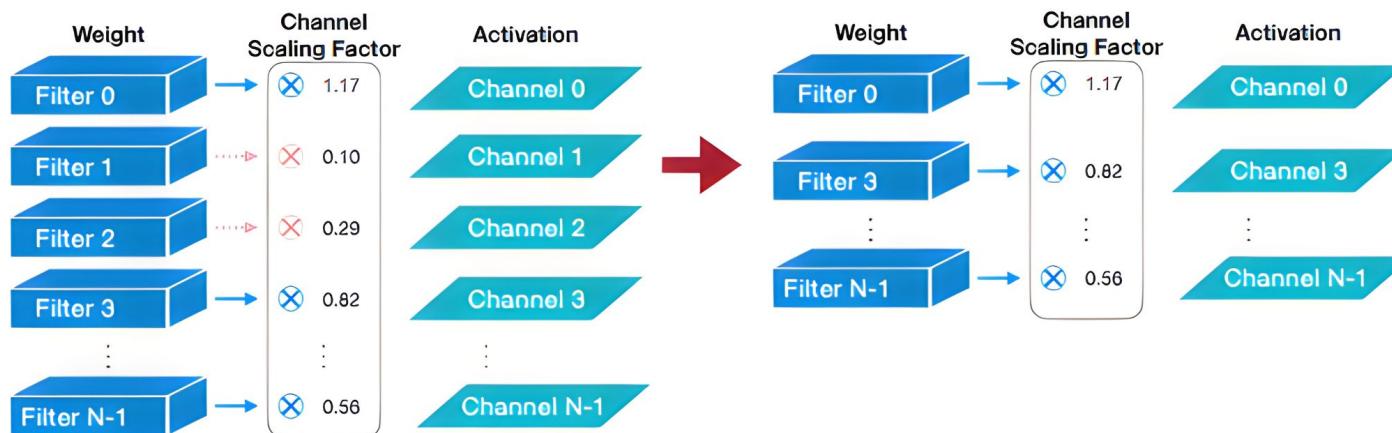
[Источник](#)

Network	Accuracy		
	Dense FP16	Sparse FP16	Sparse INT8
ResNet-34	73.7	73.9	73.7
ResNet-50	76.1	76.2	76.2
ResNet-50 (WSL)	81.1	80.9	80.9
ResNet-101	77.7	78.0	77.9
ResNeXt-50-32x4	77.6	77.7	77.7
ResNeXt-101-32x16	79.7	79.9	79.9
ResNeXt-101-32x16 (WSL)	84.2	84.0	84.2
DenseNet-121	75.5	75.3	75.3
DenseNet-161	78.8	78.8	78.9
Wide ResNet-50	78.5	78.6	78.5
Wide ResNet-101	78.9	79.2	79.1
Inception v3	77.1	77.1	77.1
Xception	79.2	79.2	79.2
VGG-11	70.9	70.9	70.8
VGG-16	74.0	74.1	74.1
VGG-19	75.0	75.0	75.0
SUNet-128	75.6	76.0	75.4
SUNet-7-128	76.4	76.5	76.3
DRN26	75.2	75.3	75.3
DRN-105	79.4	79.5	79.4

[Источник](#)

Поканальная спарсификация





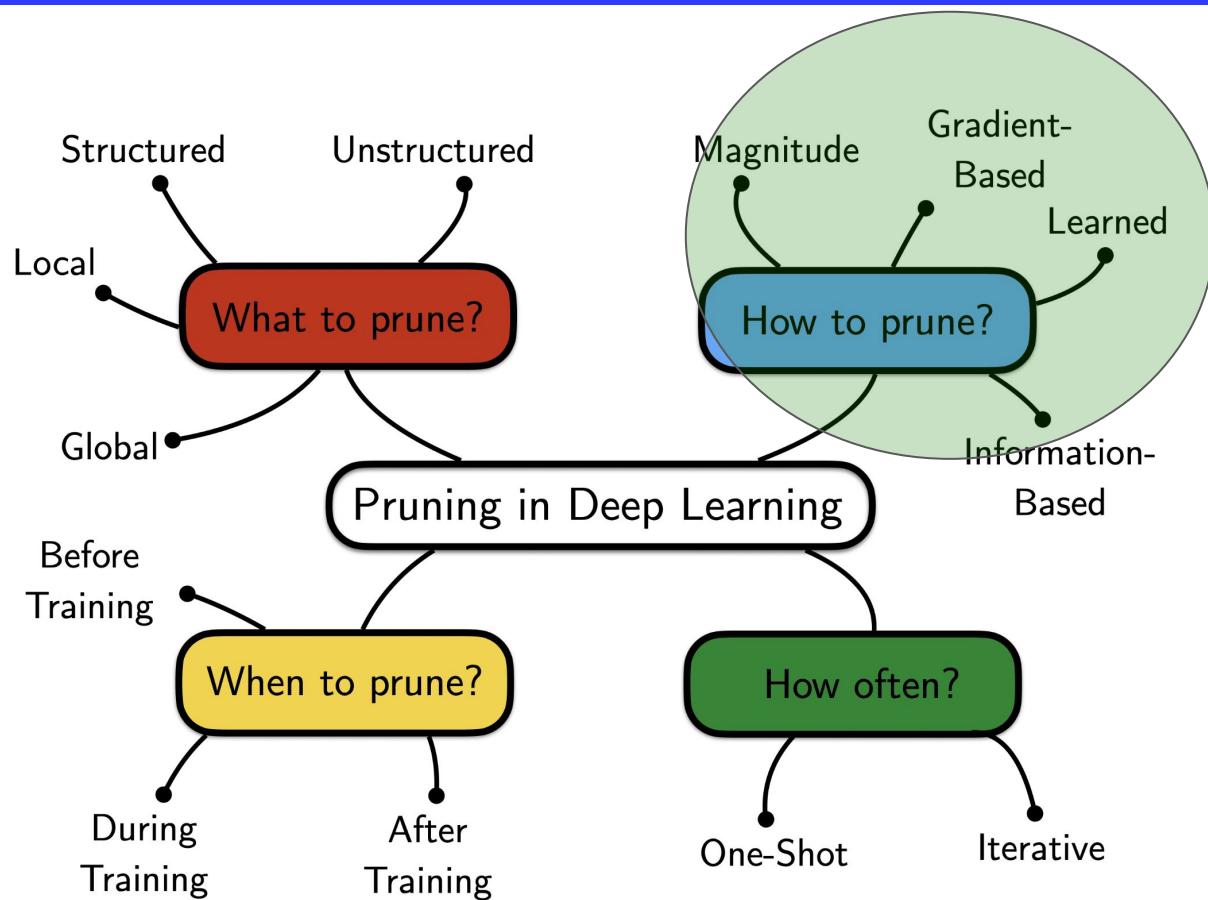
Использовать значения
BatchNorm для удаления каналов

$$\hat{x}_i = \frac{x_i - \mu_b^k}{\sqrt{(\sigma_b^k)^2 + \epsilon}}$$

$$y = \hat{x}_i \gamma + \beta$$

Заключение

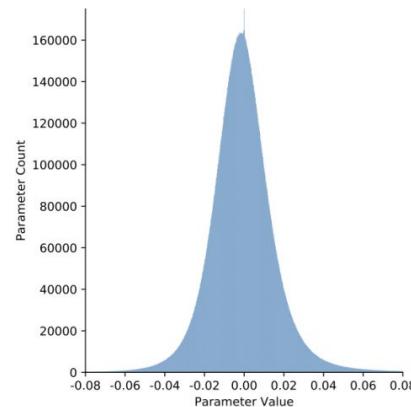
1. Для получения физических преимуществ спарсификации надо использовать виды структурного прунинга.
2. Структурный прунинг, как правило, дает качество хуже, чем неструктурный. Зато дает физическое ускорение.
3. Есть методы, которые предлагают компромисс между структурным и неструктурным прунингами.



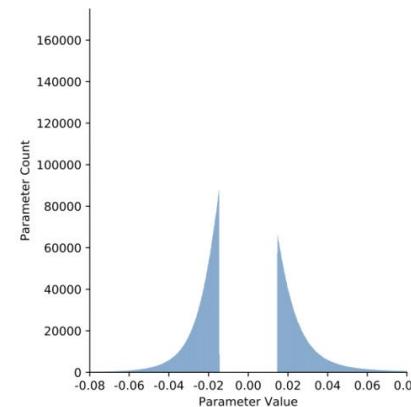
Как выбрать важность веса?

- Магнитуда весов
- На основе градиентов
- Разложение Тейлора
- ...

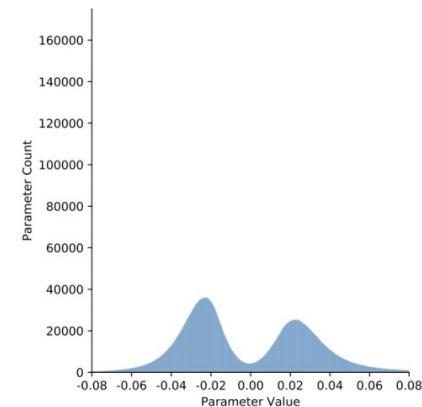
На основе магнитуды —
не требуем данных



(a) Dense Network (76.0%)



(b) 70% Pruned (36.1%)



(c) After 3-epoch Retraining (71.4%)

Если же использовать данные?

PRUNING CONVOLUTIONAL NEURAL NETWORKS FOR RESOURCE EFFICIENT INFERENCE

Лосс модели с разреженными весами

$$\mathcal{C}(\mathcal{D}|\mathcal{W}')$$

Лосс оригинальной модели

$$\mathcal{C}(\mathcal{D}|\mathcal{W})$$

Задача:

$$\min_{\mathcal{W}'} \left| \mathcal{C}(\mathcal{D}|\mathcal{W}') - \mathcal{C}(\mathcal{D}|\mathcal{W}) \right| \quad \text{s.t.} \quad \|\mathcal{W}'\|_0 \leq B,$$

[Источник](#)

PRUNING CONVOLUTIONAL NEURAL NETWORKS FOR RESOURCE EFFICIENT INFERENCE

$$\mathcal{C}(\mathcal{D}, h_i = 0) = \mathcal{C}(\mathcal{D}, h_i) - \frac{\delta \mathcal{C}}{\delta h_i} h_i + R_1(h_i = 0). \quad (5)$$

$$\Theta_{TE}(h_i) = |\Delta \mathcal{C}(h_i)| = \left| \mathcal{C}(\mathcal{D}, h_i) - \frac{\delta \mathcal{C}}{\delta h_i} h_i - \mathcal{C}(\mathcal{D}, h_i) \right| = \left| \frac{\delta \mathcal{C}}{\delta h_i} h_i \right|. \quad (7)$$

Для структурного прунинга:

$$\Theta_{TE}(z_l^{(k)}) = \left| \frac{1}{M} \sum_m \frac{\delta C}{\delta z_{l,m}^{(k)}} z_{l,m}^{(k)} \right|, \quad (8)$$

[Источник](#)

Заключение

1. В методе прунинга важную роль играет стратегия выбора весов для удаления.
2. Самая базовая стратегия — по магнитуде.
3. Более сложные методы требуют дополнительных данных и вычислений.

>>> Методы БЕЗ дообучения с
использованием данных:
OBS, OBD, EBD

OBD - Optimal Brain Damage 1989 (LeCun)

OBS - Optimal Brain Surgeon 1993

EBD - Early Bird Damage 1996

Wood - Fisher 2020 (оценка обратного гессиана)

M - FAC 2021 (оценка обратного гессиана)

Optimal Bert Surgeon 2022

Используем разложение Тейлора для оценки разницы лосса до и после спарсификации:

$$\mathcal{E}(W^*) = \mathcal{L}(W) - \mathcal{L}(W^*) = \frac{\mathcal{L}'(W^*)}{1!}(W - W^*) + \frac{\mathcal{L}''(W^*)}{1!}(W - W^*)^2 + \dots$$

$$\mathcal{E}(W^*) = (\frac{\partial L}{\partial W})^T \cdot \delta W + \frac{1}{2}\delta W^T \cdot \mathcal{H} \cdot \delta W + \mathcal{O}(\|\delta W\|^3)$$

>> [Детальный вывод и объяснение](#)

Допустим, что модель обучена до сходимости, поэтому градиент ноль.

$$\mathcal{E}(W^*) = \mathcal{L}(W) - \mathcal{L}(W^*) = \frac{\mathcal{L}'(W^*)}{1!}(W - W^*) + \frac{\mathcal{L}''(W^*)}{1!}(W - W^*)^2 + \dots$$

$$\mathcal{E}(W^*) = (\cancel{\frac{\partial L}{\partial W}})^T \cdot \delta W + \frac{1}{2} \delta W^T \cdot \mathcal{H} \cdot \delta W + \mathcal{O}(\|\delta W\|^3)$$

Формула работает как для прунинга, так и для квантизации. (δW — разница между оригиналными весами и измененными.)

Сколько элементов придется посчитать для вычисления гессиана?

Допустим, что изменение каждого веса меняет лосс независимо от других. Тогда гессиан диагональный:

$$\mathcal{E}(W^*) \approx \frac{1}{2} \delta W^T \cdot \mathcal{H} \cdot \delta W \approx \frac{1}{2} \sum_i h_{ii} \delta W_i^2 \quad (1)$$

$$h_{ij} = \frac{\partial L^2}{\partial W_i \partial W_j} \quad (2)$$

Таким образом можно выбрать веса для удаления, которые минимизируют изменение функции потерь.

Optimal Brain Damage - Algorithm:

- For each network weight W_i , determine its corresponding $h_{ii} = \frac{\partial L^2}{\partial W_i \partial W_i}$
- For each h_{ii} compute its saliency $\frac{1}{2}h_{ii}w_i^2$
- Sort all values by its saliency scores and delete or quantize the weights

Optimal Brain Surgeon

Допустим, e_i это one-hot-вектор, соответствующий изменению i -го элемента.

- Pruning: $e_i^T \times \delta W + W_i = 0$
- Quantization: $e_i^T \times \delta W + (W_i - Q(W_i)) = 0$

Проблема прунинга тогда переписывается в следующем виде:

$$\min_{i \in W} \left\{ \min_{\delta W} \left(\frac{1}{2} \delta W^T \cdot \mathcal{H} \cdot \delta W \right) | e_i^T \cdot \delta W + W_i = 0 \right\}$$

$$\min_{i \in W} \left\{ \min_{\delta W} \left(\frac{1}{2} \delta W^T \cdot \mathcal{H} \cdot \delta W \right) | e_i^T \cdot \delta W + W_i = 0 \right\}$$

For that, we use Lagrange multipliers:

$$L = \frac{1}{2} \delta W^T \cdot \mathcal{H} \cdot \delta W + \lambda (e_i^T \cdot \delta W + W_i)$$

$$\mathcal{H} \delta W + \lambda e_i^T = 0$$

$$e_i^T \cdot \delta W + W_i = 0$$

$$\lambda e_i^T \mathbf{H}^{-1} e_i^T = W_i$$

$$\lambda = \frac{W_i}{[\mathcal{H}^{-1}]_{ii}}$$

Как должны измениться другие веса, если удалить один вес

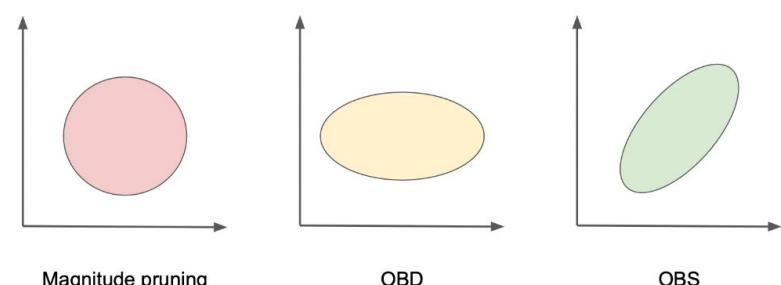
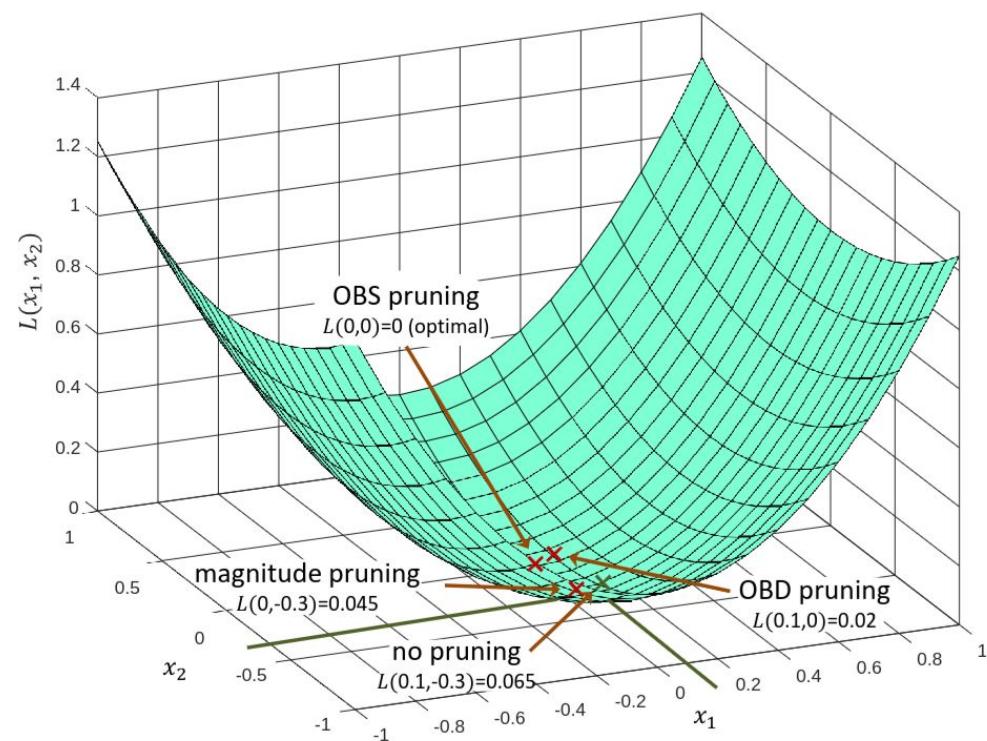
$$\delta W = - \frac{W_i}{[\mathcal{H}^{-1}]_{ii}} \mathcal{H}^{-1} \cdot e_i$$

$$\mathcal{E} = \frac{1}{2} \frac{W_i^2}{[\mathcal{H}^{-1}]_{ii}}$$

Optimal Brain Surgeon - Algorithm:

- Compute H^{-1}
- Find weights that has the smallest importance $\mathcal{E} = \frac{1}{2} \frac{W_i^2}{[\mathcal{H}^{-1}]_{ii}}$
- Remove or quantize these weights
- Update remaining weights $\delta W = -\frac{W_i}{[\mathcal{H}^{-1}]_{ii}} \mathcal{H}^{-1} \cdot e_i$
- Repeat

>>> OBS



>>> 50

OBS & GPTQ For LLM

Теперь надо адаптировать эти решения для LLM.

Проблемы:

- Вычислять H очень дорого.
- Слишком много параметров обновлять.
- После каждого изменения весов надо обновлять H .

OBS & GPTQ For LLM

Теперь надо адаптировать эти решения для LLM.

Решения:

Проблемы:

- Вычислять H очень дорого.
- Слишком много параметров обновлять.
- После каждого изменения весов надо обновлять H .

Algorithm 1 Prune $k \leq d_{\text{col}}$ weights from row \mathbf{w} with inverse Hessian $\mathbf{H}^{-1} = (2\mathbf{X}\mathbf{X}^\top)^{-1}$ according to OBS in $O(k \cdot d_{\text{col}}^2)$ time.

```

 $M = \{1, \dots, d_{\text{col}}\}$ 
for  $i = 1, \dots, k$  do
     $p \leftarrow \operatorname{argmin}_{p \in M} \frac{1}{[\mathbf{H}^{-1}]_{pp}} \cdot w_p^2$ 
     $\mathbf{w} \leftarrow \mathbf{w} - \mathbf{H}_{:,p}^{-1} \frac{1}{[\mathbf{H}^{-1}]_{pp}} \cdot w_p$ 
     $\mathbf{H}^{-1} \leftarrow \mathbf{H}^{-1} - \frac{1}{[\mathbf{H}^{-1}]_{pp}} \mathbf{H}_{:,p}^{-1} \mathbf{H}_{p,:}$ 
     $M \leftarrow M - \{p\}$ 
end for
```

- Рассмотрим проблему по слойно и разложим MSE между выходом обычного и обрезанного слоев.
- Обновляем веса построчно.

$$\mathcal{L} = \|(W\mathbf{X}^T - \hat{W}\mathbf{X}^T)\|^2$$

$$\mathcal{H} = \frac{\partial^2 \mathcal{L}}{\partial^2 W} = \frac{\partial^2}{\partial^2 W} \|(W\mathbf{X}^T - \hat{W}\mathbf{X}^T)\|^2 = \mathbf{X}\mathbf{X}^T$$

[Optimal Brain Compression: A Framework for Accurate Post-Training Quantization and Pruning](#)

OBS & GPTQ For LLM

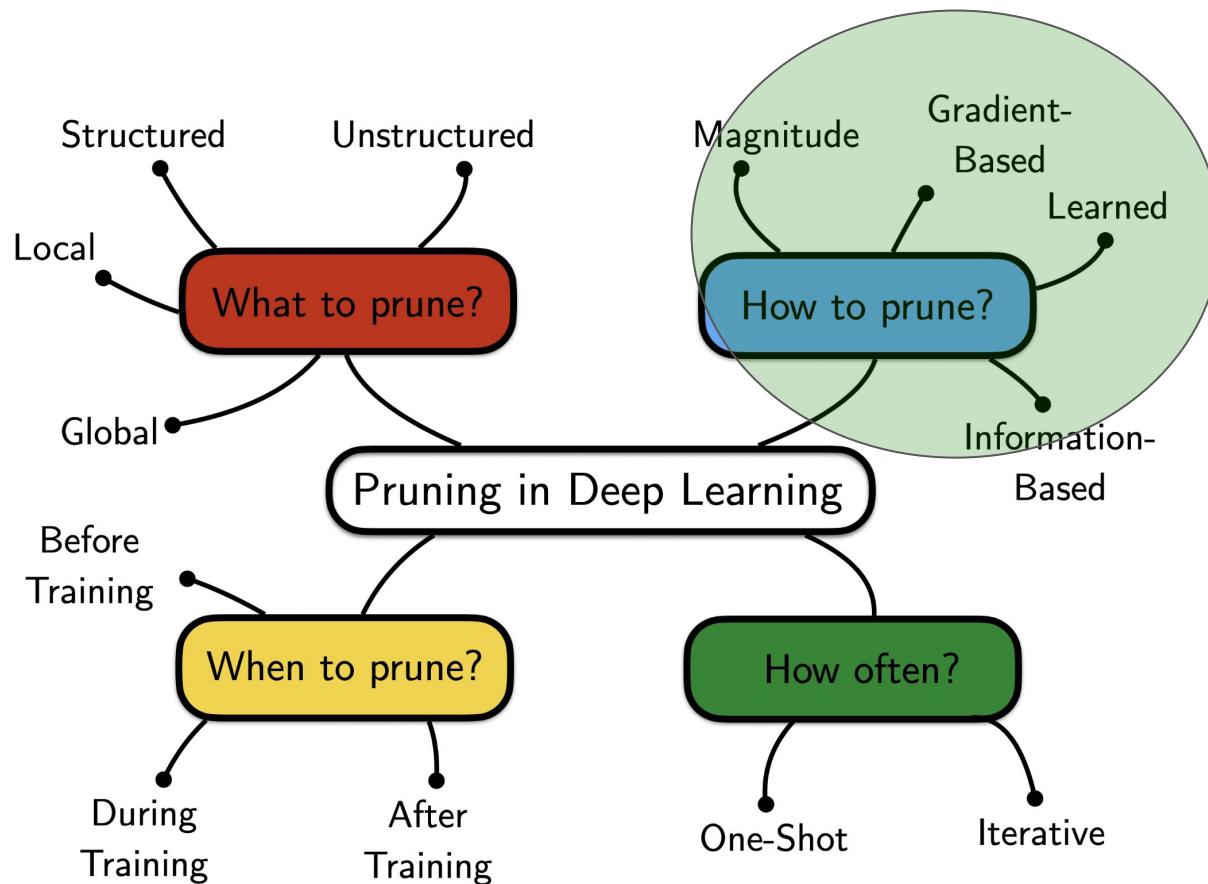
SOTA результаты

Method	ResNet50 – 76.13			YOLOv5l – 66.97			BERT – 88.53		
	2×	3×	4×	2×	3×	4×	2×	3×	4×
GMP	74.86	71.44	64.84	65.83	62.30	55.09	65.64	12.52	09.23
L-OBS	75.48	73.73	71.24	66.21	64.47	61.15	77.67	3.62	6.63
AdaPrune	75.53	74.47	72.39	66.00	64.88	62.71	87.12	70.32	18.75
ExactOBS	75.64	75.01	74.05	66.14	65.35	64.05	87.81	85.87	82.10

[Optimal Brain Compression: A Framework for Accurate Post-Training Quantization and Pruning](#)

Заключение

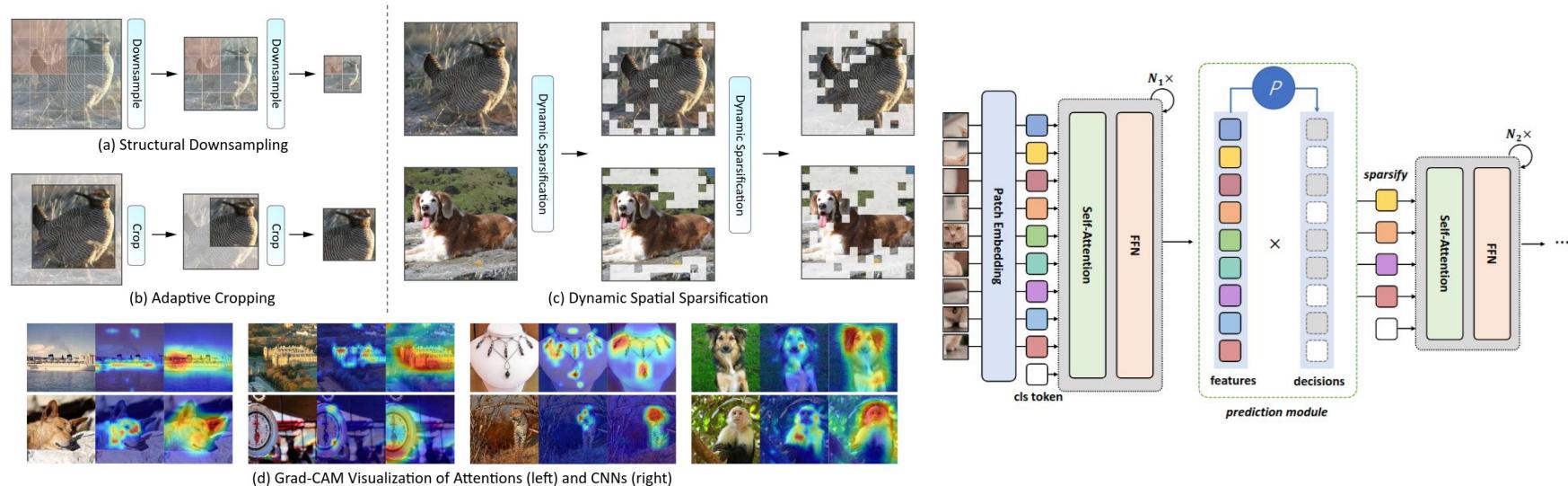
1. Не всегда есть возможность проводить прунинг по магнитуде и итеративный прунинг — это может быть дорого.
2. Техники, разработанные много лет назад, находят свое применение сейчас в условиях больших языковых моделей.
3. Их оценки возможны благодаря множеству допущений, которые ухудшают оценку, но значительно ускоряют вычисления.



>>>

Динамическая
спарсификация

Динамическая спарсификация



[Dynamic Spatial Sparsification for Efficient Vision Transformers and Convolutional Neural Networks](#)

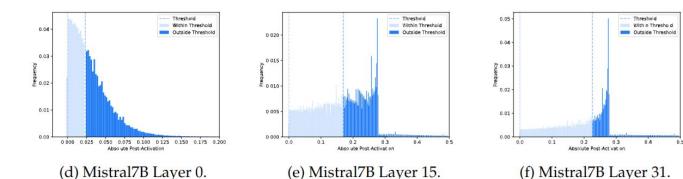
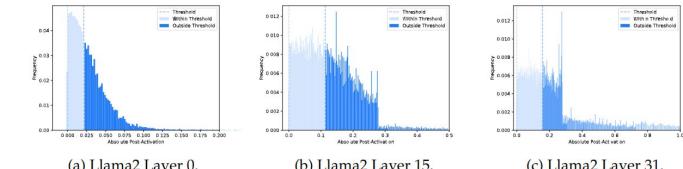
Динамическая спарсификация

$$\text{Gated-MLP}(x) := (\text{SiLU}(xW_{\text{gate}}) * (xW_{\text{up}}))W_{\text{down}}$$

$$\text{CATS}_t(\text{SiLU}(xW_{\text{gate}})) = \begin{cases} \text{SiLU}(xW_{\text{gate}}) & |\text{SiLU}(xW_{\text{gate}})| \geq t \\ 0 & |\text{SiLU}(xW_{\text{gate}})| < t \end{cases}$$

Custom GPU Kernel 1 MLP using CATS

- 1: **Input:** threshold $t > 0$, hidden layer x , weights W_{gate} , W_{down} , and W_{up}
- 2: $v \leftarrow \text{SiLU}(xW_{\text{gate}})$
- 3: $\text{Mask} \leftarrow 1$ if $|v| \geq t$ else 0
- 4: $x_1 \leftarrow (xW_{\text{up}}[\text{Mask}] * v[\text{Mask}])$
- 5: $y \leftarrow x_1 W_{\text{down}}[\text{Mask}]$



[CATS: Contextually-Aware Thresholding for Sparsity in Large Language Models](#)

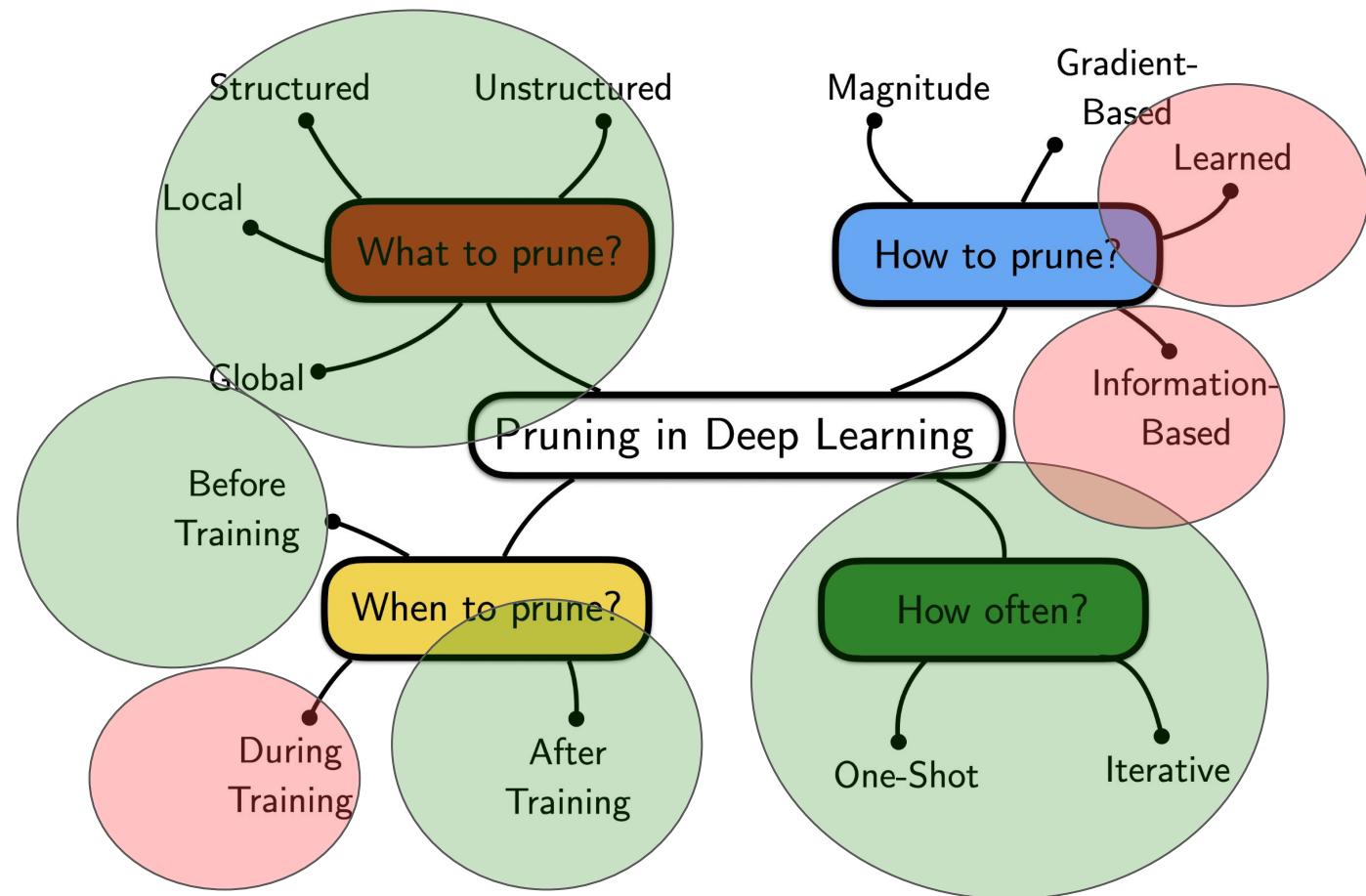
Динамическая спарсификация

Model \ Dataset	WG acc↑	PIQA acc↑	SciQ acc↑	QA acc↑	HS acc↑	BoolQ acc↑	Arc-E acc↑	Arc-C acc↑	Avg acc↑
Mistral-7B	0.7419	0.8069	0.959	0.3260	0.6128	0.8370	0.8085	0.5034	0.6994
CATS 50%	0.7245	0.8009	0.948	0.3200	0.6097	0.8193	0.7849	0.5043	0.6890
CATS 70%	0.7190	0.8003	0.929	0.292	0.6057	0.8028	0.7492	0.4693	0.6709
CATS 90%	0.5627	0.6001	0.422	0.212	0.3359	0.7086	0.3754	0.2773	0.4368
ReLUification	0.5043	0.5092	0.236	0.142	0.2580	0.4208	0.2723	0.2415	0.3230
Llama2-7B	0.6906	0.7807	0.94	0.314	0.5715	0.7774	0.7630	0.4343	0.6589
CATS 50%	0.6748	0.7693	0.927	0.322	0.5711	0.7263	0.7441	0.4121	0.6433
CATS 70%	0.6693	0.7584	0.902	0.294	0.5500	0.6590	0.7008	0.3805	0.6143
CATS 90%	0.5738	0.6627	0.611	0.212	0.3848	0.6284	0.4566	0.2816	0.4764
ReLUification	0.4893	0.5408	0.2570	0.154	0.2586	0.6003	0.2795	0.2406	0.3525

[CATS: Contextually-Aware Thresholding for Sparsity in Large Language Models](#)

Заключение

1. Динамическая спарсификация не дает уменьшения весов и памяти.
2. Вместо этого уменьшается размер активаций — происходит чистое ускорение.

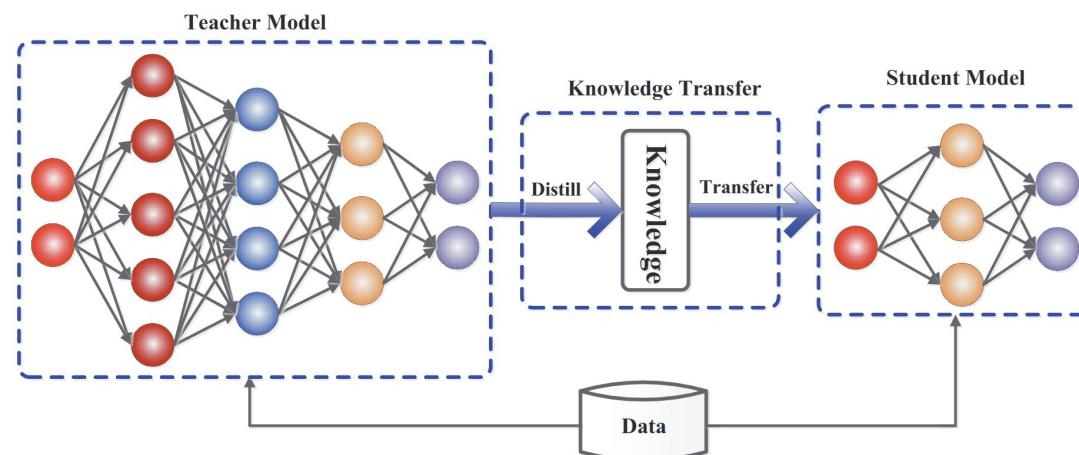


>>>

Дистилляция

Маленькие модели плохо оптимизируются.
Как помочь им уже известными способами?

Knowledge distillation



[Источник](#)

Knowledge distillation

Вместо того, чтобы просто обучать маленькую/разреженную/квантизованную модель градиентным спуском на таргеты, мы будем учить ее копировать поведение большой/сложной модели-учителя.

[Источник](#)

Knowledge distillation

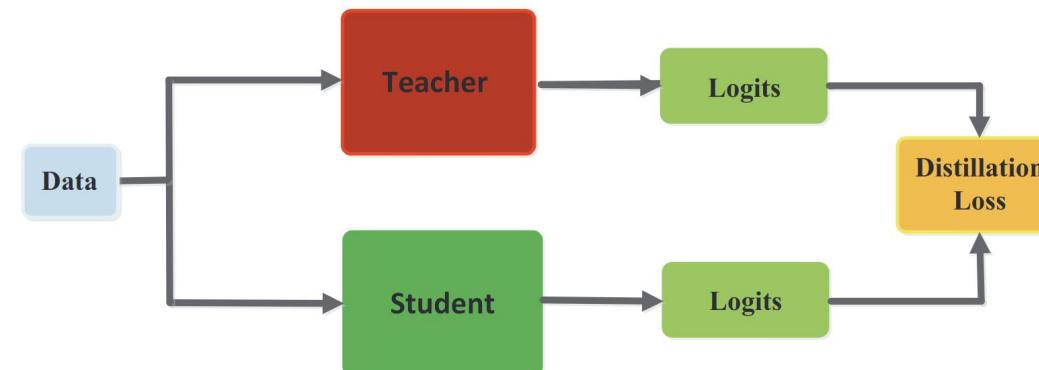
Используется там, где нужна эффективность:

1. Приложения на телефон
2. Автопилот для машин
3. Квантизированные модели
4. Разреженные модели

[Источник](#)

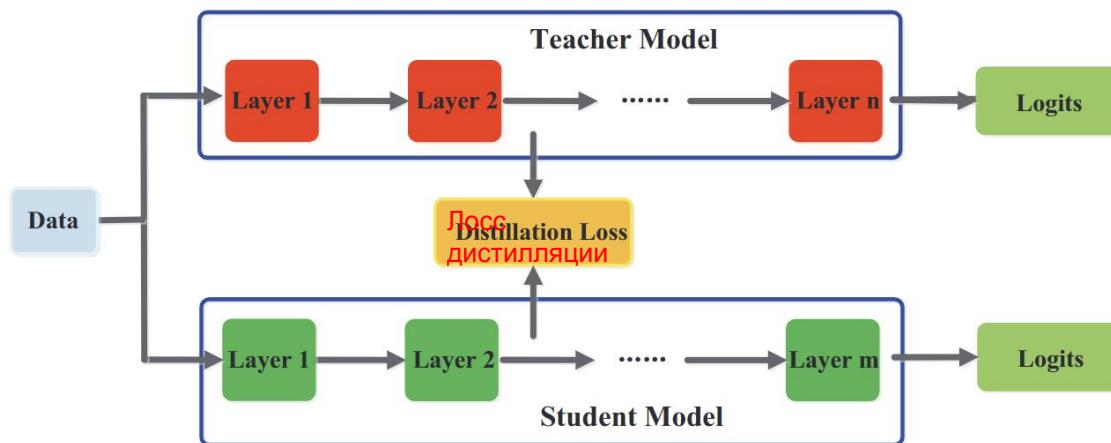
Knowledge distillation

Дистилляция выходов

[Источник](#)

Knowledge distillation

Послойная дистилляция

[Источник](#)

Заключение

1. Дистилляция знаний позволяет передавать опыт большой предобученной модели другим моделям-ученикам.
2. Эту технику можно использовать поверх других техник компрессии:
 - a. Учить маленькую модель с помощью большой.
 - b. Учить спарсифицированную модель с помощью оригинальной.
 - c. Учить квантизированную модель с помощью оригинальной.