



НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ
УНИВЕРСИТЕТ

Bayesian Compression for Deep Learning[1]

Подготовила:
Мадуар Дарин

НИУ ВШЭ

5 марта, 2020

Цена вопроса

- 1 кВт·ч стоит ~ 21.1 евроцентов;

Цена вопроса

- 1 кВт·ч стоит ~ 21.1 евроцентов;
- стоимость NVIDIA TITAN X ~ 4.77 евроцента в час;

Цена вопроса

- 1 кВт·ч стоит ~ 21.1 евроцентов;
- стоимость NVIDIA TITAN X ~ 4.77 евроцента в час;
- у компании Facebook порядка 2.5 млрд активных пользователей;

Цена вопроса

- 1 кВт·ч стоит ~ 21.1 евроцентов;
- стоимость NVIDIA TITAN X ~ 4.77 евроцента в час;
- у компании Facebook порядка 2.5 млрд активных пользователей;
- VGG тратит ~ 147 мс / 16 предсказаний;

Цена вопроса

- 1 кВт·ч стоит ~ 21.1 евроцентов;
- стоимость NVIDIA TITAN X ~ 4.77 евроцента в час;
- у компании Facebook порядка 2.5 млрд активных пользователей;
- VGG тратит ~ 147 мс / 16 предсказаний;
- одно предсказание обходится ~ 20 тыс. евро.

Summary

- мобильные устройства имеют очень **ограниченную процессорную мощность**;

Summary

- мобильные устройства имеют очень **ограниченную процессорную мощность**;
- **стоимость энергопотребления** для получения предсказаний;

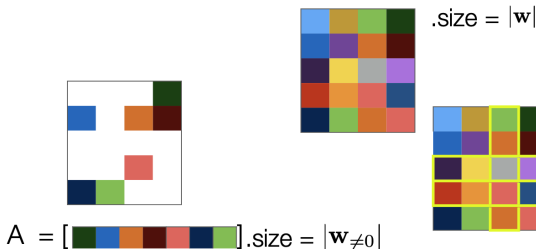
Summary

- мобильные устройства имеют очень **ограниченную процессорную мощность**;
- **стоимость энергопотребления** для получения предсказаний;
- **ускорение обработки** в режиме реального времени;

Summary

- **мобильные устройства имеют очень ограниченную процессорную мощность;**
- **стоимость энергопотребления для получения предсказаний;**
- **ускорение обработки в режиме реального времени;**
- **лучшая защита личных данных.**

Sparsity learning

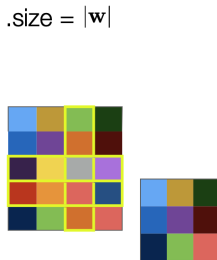


IC = [0 1 3 0 1 2]

IR = [3, 0, 2, 3, ..., 1]

Unstructured Pruning

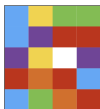
$$[2]CR \approx \frac{|\mathbf{w}|}{2|\mathbf{w}_0|}$$



Structured Pruning

$$CR = \frac{|\mathbf{w}|}{|\mathbf{w}_0|}$$

Quantisation



Precision Quantisation

$$CR = \frac{32}{10} \approx 3$$

- + быстрый в реализации
- сжатие незначительно



?

1	-3.5667
2	0.4545
3	0.8735
4	0.33546

Set Quantisation by clustering

$$CR = \frac{32}{4} = 8$$

- + очень хорошо сжимает
- как реализовывать?

Принцип минимальной длины описания (*minimum description length, MDL*)

[Принцип MDL] основыван на следующем осознании: любая закономерность в заданном наборе данных может быть использована для сжатия данных, то есть описания данных с использованием меньшего набора символов, чем нужно для описания данных буквально. (Грюнвальд, 1998)[3]

Вариационный байесовский вывод

- Пусть $\mathcal{D} = (\mathbf{X}, \mathbf{y})$ – набор данных. Причем,
 $p(\mathcal{D} \mid \mathbf{w}) = \prod_{(x,y) \in \mathcal{D}} p(y \mid x, \mathbf{w})$ и мы предполагаем
априорное распределение на веса $p(\mathbf{w})$;

$$\mathcal{L}(\phi) = \underbrace{\mathbb{E}_{q_{\phi}(\mathbf{w})}[\log p(\mathcal{D} \mid \mathbf{w})]}_{\mathcal{L}^E} + \underbrace{\mathbb{E}_{q_{\phi}(\mathbf{w})}[\log p(\mathbf{w})] + \mathcal{H}(q_{\phi}(\mathbf{w}))}_{\mathcal{L}^C},$$

Вариационный байесовский вывод

- Пусть $\mathcal{D} = (\mathbf{X}, \mathbf{y})$ – набор данных. Причем,
 $p(\mathcal{D} | \mathbf{w}) = \prod_{(x,y) \in \mathcal{D}} p(y | x, \mathbf{w})$ и мы предполагаем
априорное распределение на веса $p(\mathbf{w})$;
- Хотим приблизить $p(\mathbf{w} | \mathcal{D})$ некоторым параметрическим
распределением $q_\phi(\mathbf{w})$;

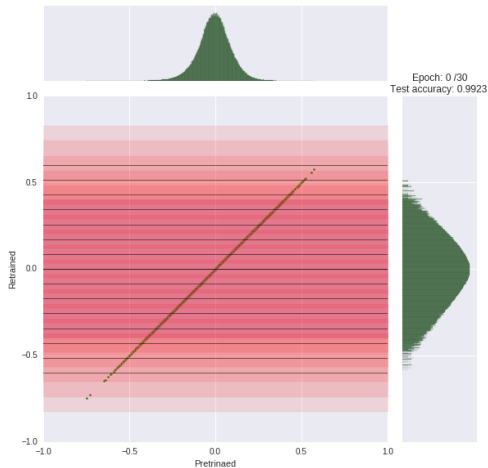
$$\mathcal{L}(\phi) = \underbrace{\mathbb{E}_{q_\phi(\mathbf{w})}[\log p(\mathcal{D} | \mathbf{w})]}_{\mathcal{L}^E} + \underbrace{\mathbb{E}_{q_\phi(\mathbf{w})}[\log p(\mathbf{w})] + \mathcal{H}(q_\phi(\mathbf{w}))}_{\mathcal{L}^C},$$

Вариационный байесовский вывод

- Пусть $\mathcal{D} = (\mathbf{X}, \mathbf{y})$ – набор данных. Причем,
 $p(\mathcal{D} | \mathbf{w}) = \prod_{(x,y) \in \mathcal{D}} p(y | x, \mathbf{w})$ и мы предполагаем
априорное распределение на веса $p(\mathbf{w})$;
- Хотим приблизить $p(\mathbf{w} | \mathcal{D})$ некоторым параметрическим
распределением $q_{\phi}(\mathbf{w})$;
- ϕ – оптимизируемый параметр

$$\mathcal{L}(\phi) = \underbrace{\mathbb{E}_{q_{\phi}(\mathbf{w})}[\log p(\mathcal{D} | \mathbf{w})]}_{\mathcal{L}^E} + \underbrace{\mathbb{E}_{q_{\phi}(\mathbf{w})}[\log p(\mathbf{w})] + \mathcal{H}(q_{\phi}(\mathbf{w}))}_{\mathcal{L}^C},$$

Soft weight-sharing for NN compression[4]



Soft weight-sharing for NN compression

$$q(w) = \prod_i q(w_i) = \delta(w_i \mid \mu_i)$$

$$p(w) = \prod_i \sum_j \pi_j \mathcal{N}(w_i \mid \mu_j, \sigma_j^2)$$

Model	Method	Top-1 Error[%]	Δ [%]	$ \mathbf{W} [10^6]$	$\frac{ \mathbf{W}_{\neq 0} }{ \mathbf{W} }$ [%]	CR
LeNet-300-100	Han et al. (2015a)	1.64 \rightarrow 1.58	0.06	0.2	8.0	40
	Guo et al. (2016)	2.28 \rightarrow 1.99	-0.29		1.8	56
	Ours	1.89 \rightarrow 1.94	-0.05		4.3	64
LeNet-5-Caffe	Han et al. (2015a)	0.80 \rightarrow 0.74	-0.06	0.4	8.0	39
	Guo et al. (2016)	0.91 \rightarrow 0.91	0.00		0.9	108
	Ours	0.88 \rightarrow 0.97	0.09		0.5	162
ResNet (light)	Ours	6.48 \rightarrow 8.50	2.02	2.7	6.6	45

Bayesian compression with scale mixtures of normals

Основная идея: сделать **вариационный дропаут**!

- Обучаем dropout rate для каждого распределения весов;

Bayesian compression with scale mixtures of normals

Основная идея: сделать **вариационный дропаут**!

- Обучаем dropout rate для каждого распределения весов;
- Веса с высоким dropout rate игнорируем;

Bayesian compression with scale mixtures of normals

Основная идея: сделать **вариационный дропаут**!

- Обучаем dropout rate для каждого распределения весов;
- Веса с высоким dropout rate игнорируем;
- Используем дисперсию распределения весов, чтобы понимать какой bit precision валидно использовать.

Bayesian compression with scale mixtures of normals

$$q(z) = \prod q(z_i) = \mathcal{N}(z_i | \mu_i^z, \alpha_i)$$
$$q(\mathbf{w}|z) = \prod q(w_i|z_i) = \mathcal{N}(w_i | z_i \mu_i, z_i^2 \sigma_i^2)$$

$$p(w) = \int p(z) p(w|z) dz$$
$$p(w) \propto \int \frac{1}{|z|} \mathcal{N}(w|0, z^2) dz = \frac{1}{|w|}$$

Bayesian compression with scale mixtures of normals

Примеры априорных распределений на z :

- **log-uniform prior** $p(z) \sim |z|^{-1}$
- **half-Cauchy scale prior**
 $p(z) \sim \mathcal{C}^+(0, s) = 2(s\pi(1 + (z/s)^2))^{-1}$

Эксперименты

Table 1: Learned architectures with Sparse VD [51], Generalized Dropout (GD) [66] and Group Lasso (GL) [73]. Bayesian Compression (BC) with group normal-Jeffreys (BC-GNJ) and group horseshoe (BC-GHS) priors correspond to the proposed models. We show the amount of neurons left after pruning along with the average bit precisions for the weights at each layer.

Network & size	Method	Pruned architecture	Bit-precision
LeNet-300-100	Sparse VD	512-114-72	8-11-14
784-300-100	BC-GNJ	278-98-13	8-9-14
	BC-GHS	311-86-14	13-11-10
LeNet-5-Caffe	Sparse VD	14-19-242-131	13-10-8-12
20-50-800-500	GD	7-13-208-16	-
	GL	3-12-192-500	-
	BC-GNJ	8-13-88-13	18-10-7-9
	BC-GHS	5-10-76-16	10-10-14-13
VGG	BC-GNJ	63-64-128-128-245-155-63- -26-24-20-14-12-11-11-15	10-10-10-10-8-8-8- -5-5-5-5-5-6-7-11
(2×64)-(2×128)- (3×256)-(8×512)	BC-GHS	51-62-125-128-228-129-38- -13-9-6-5-6-6-6-20	11-12-9-14-10-8-5- -5-6-6-6-8-11-17-10

Эксперименты

Table 2: Compression results for our methods. “DC” corresponds to Deep Compression method introduced at [25], “DNS” to the method of [21] and “SWS” to the Soft-Weight Sharing of [70]. Numbers marked with * are best case guesses.

Model		Method	$\frac{ w \neq 0 }{ w } \%$	Compression Rates (Error %)		
				Pruning	Fast Prediction	Maximum Compression
LeNet-300-100	1.6	DC	8.0	6 (1.6)	-	40 (1.6)
		DNS	1.8	28* (2.0)	-	-
		SWS	4.3	12* (1.9)	-	64(1.9)
		Sparse VD	2.2	21(1.8)	84(1.8)	113 (1.8)
		BC-GNJ	10.8	9(1.8)	36(1.8)	58(1.8)
		BC-GHS	10.6	9(1.8)	23(1.9)	59(2.0)
LeNet-5-Caffe	0.9	DC	8.0	6*(0.7)	-	39(0.7)
		DNS	file input	55*(0.9)	-	108(0.9)
		SWS	0.5	100*(1.0)	-	162(1.0)
		Sparse VD	0.7	63(1.0)	228(1.0)	365(1.0)
		BC-GNJ	0.9	108(1.0)	361(1.0)	573(1.0)
		BC-GHS	0.6	156(1.0)	419(1.0)	771(1.0)
VGG	8.4	BC-GNJ	6.7	14(8.6)	56(8.8)	95(8.6)
		BC-GHS	5.5	18(9.0)	59(9.0)	116(9.2)

Эксперименты

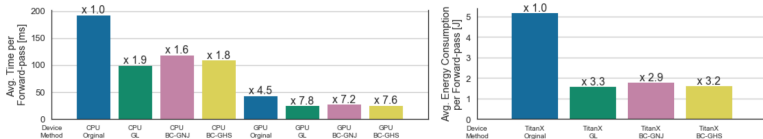






Figure 1: **Left:** Avg. Time a batch of 8192 samples takes to pass through LeNet-5-Caffe. Numbers on top of the bars represent speed-up factor relative to the CPU implementation of the original network. **Right:** Energy consumption of the GPU of the same process (when run on GPU).

-  [Christos Louizos, Karen Ullrich, and Max Welling.](#)
Bayesian compression for deep learning, 2017.
-  [Song Han, Huizi Mao, and William J. Dally.](#)
Deep compression: Compressing deep neural networks with pruning, trained quantization and huffman coding, 2015.
-  [Jorma Rissanen.](#)
Information and Complexity in Statistical Modeling.
Springer Publishing Company, Incorporated, 1 edition, 2007.
-  [Karen Ullrich, Edward Meeds, and Max Welling.](#)
Soft weight-sharing for neural network compression, 2017.

1. Дана выборка $\mathcal{D} = (\mathbf{X}, \mathbf{y})$. Пусть $p(\mathcal{D} \mid \mathbf{w}) = \prod_{(x,y) \in \mathcal{D}} p(y \mid x, \mathbf{w})$ и нам известно априорное распределение на веса $p(\mathbf{w})$. Опишите схему вариационного байесовского вывода для нахождения $q_{\phi}(\mathbf{w})$. Запишите формулу для оптимизируемого функционала $\mathcal{L}(\phi)$.
2. Опишите общую идею принципа MDL (Minimum Description Length Principle). Какую задачу он решает?
3. Опишите общую схему Bayesian compression:
 - какая задача решается?
 - как именно? (опишите основные этапы, выпишите формулы распределений на скрытые параметры и веса, покажите связь с вариационным байесовским выводом)
 - какие распределения можно выбрать в качестве априрных на скрытые параметры?