Compression of Deep Convolutional Neural Networks

Anna Klyueva
Ilya Nachinkin
Matvey Morozov

Skoltech

Our team

Anna Klyueva

- presentation
- report

Ilya Nachinkin

- implementation
- report

Matvey Morozov

- implementation
- presentation

Problem Statement

Why? Deep convolutional neural networks tend to be overparameterized, therefore training and testing procedures might be time and energy consuming.

What? One way to reduce computation costs is to compress the whole convolutional neural network or just a single convolutional layer.

Hypothesis We want to implement Tucker decomposition on convolution kernel tensor to compress the whole CNN.

Application One of the possible applications is to allocate deep CNNs for complex tasks such as ImageNet classification on mobile devices.

How to measure quality We estimate number of model parameters and model accuracy after compression

Methods

In our work we present a scheme to compress the entire CNN:

Rank selection with variational Bayesian matrix factorization → Tucker decomposition on kernel tensor → Fine-tuning to recover accumulated loss of accuracy

We conducted our experiments on **CIFAR10** dataset.

Mathematical Description

Convolution Kernel Tensor

In CNNs the **Convolution kernel tensor** maps an input tensor $\mathcal{X}$ of size $H \times W \times S$ into an output tensor $\mathcal{Y}$ of size $H' \times W' \times T$ using the following linear mapping:

$$\mathcal{Y}_{h',w',t} = \sum_{i=1}^{D}\sum_{j=1}^{D}\sum_{s=1}^{S}\mathcal{K}_{i,j,s,t}\,\mathcal{X}_{h_i,w_j,s}$$

$$h_i = (h'-1)\Delta + i - P$$

$$w_j = (w'-1)\Delta + j - P$$

where $\mathcal{K}$ is a 4-way dimensional tensor of size $D \times D \times S \times T$, Δ is stride and P is zero-padding size.

Mathematical Description

Tucker Decomposition

The rank-(R_1, R_2, R_3, R_4) Tucker decomposition of 4-way kernel tensor has the form:

$$\mathcal{K}_{i,j,s,t} = \sum_{r_1=1}^{R_1} \sum_{r_2=1}^{R_2} \sum_{r_3=1}^{R_3} \sum_{r_4=1}^{R_4} \mathcal{C}'_{r_1,r_2,r_3,r_4} U^{(1)}_{i,r_1} U^{(2)}_{j,r_2} U^{(3)}_{s,r_3} U^{(4)}_{t,r_4}$$

where $\mathcal{C}'$ is a core tensor of size $R_1 \times R_2 \times R_3 \times R_4$ and $\boldsymbol{U}^{(1)}, \boldsymbol{U}^{(2)}, \boldsymbol{U}^{(3)}$ and $\boldsymbol{U}^{(4)}$ are factor matrices of size $D \times R_1, D \times R_2, S \times R_3$ and $T \times R_4$.

Under the variant called Tucker-2 decomposition the kernel tensor is decomposed to:

$$\mathcal{K}_{i,j,s,t} = \sum_{r_3=1}^{R_3} \sum_{r_4=1}^{R_4} \mathcal{C}_{i,j,r_3,r_4} \, U^{(3)}_{s,r_3} \, U^{(4)}_{t,r_4}$$

with the core vector of size $D \times D \times R_3 \times R_4$.

Mathematical Description

Rank selection with variational Bayesian matrix factorization

The rank-(R_3, R_4) are hyper-parameters which control both accuracy loss and performance improvement. They are determined by applying global analytic VBMF on mode-3 matricization and mode-4 matricization of kernel tensor.

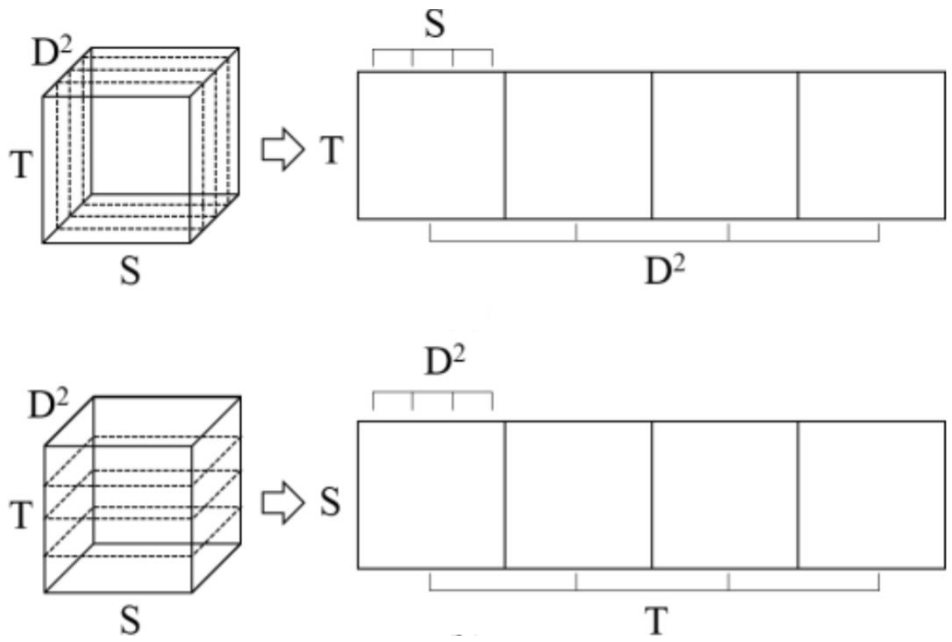

Figure 1. Tensor matricization

Complexity analysis

The convolution operation requires D^2ST parameters (kernel size). With Tucker decomposition compression ratio is given by:

$$\frac{D^2ST}{SR_3 + D^2R_3R_4 + TR_4}$$

Experiments

model	ratio	accuracy_before	accuracy_after	fine_tune
AlexNet	1.037	0.7469	0.286	0.7827
ResNet 18	1.18	0.8065	0.8126	0.8045
VGG 16	1.11	0.8663	0.1496	0.8034
ShuffleNet V2	**1.81**	0.8542	0.8684	0.90
DenseNet 161	1.08	0.8611	0.8828	0.91

Summary

As a result we obtained reductions in model size often without loosing in quality.

Potentially, we can conduct more experiments testing different models and datasets, for example, **ImageNet**.

Bibliography

Yong-Deok Kim, Eunhyeok Park, Sungjoo Yoo, Taelim Choi, Lu Yang & Dongjun Shin. (2016) **COMPRESSION OF DEEP CONVOLUTIONAL NEURAL NETWORKS FOR FAST AND LOW POWER MOBILE APPLICATIONS**, conference paper at ICLR.

Tucker, Ledyard R. (1966) **Some mathematical notes on three-mode factor analysis**, Psychometrika.

Nakajima, Shinichi, Tomioka, Ryota, Sugiyama, Masashi, and Babacan, S Derin.(2012) **Perfect dimensionality recovery by variational bayesian pca. In Advances in Neural Information Processing Systems**, pp. 971–979

Shashua, Amnon and Hazan, Tamir. **Non-negative tensor factorization with applications to statistics and computer vision. In Proceedings of the 22nd international conference on Machine learning**, pp. 792–799. ACM, 2005.

Ye, Jieping. **Generalized low rank approximations of matrices.** Machine Learning, 61(1-3):167–191, 2005.

thx.

Appendix

Tucker decomposition

The Tucker decomposition decomposes a tensor $\mathcal{Y} \in \mathbb{R}^{I_1 \times I_2 \times I_3}$ into a smaller tensor, called the *core tensor*, and *factor matrices*. The principle is illustrated in Figure 3. The core tensor $\mathcal{G} \in \mathbb{R}^{R_1 \times R_2 \times R_3}$ is multiplied by matrices $\mathbf{A} \in \mathbb{R}^{I_1 \times R_1}$,$\mathbf{B} \in \mathbb{R}^{I_2 \times R_2}$ and $\mathbf{C} \in \mathbb{R}^{I_3 \times R_3}$ along the first, second and third mode, respectively. Mode-1 and mode-2 multiplication are equivalent to left and right multiplication in case of matrices. In general, mode-n multiplication will be indicated by $\times_n$. Hence, the Tucker decomposition in Figure 3 can be written as:

$$\mathcal{Y} = \mathcal{G} \times_1 \mathbf{A} \times_2 \mathbf{B} \times_3 \mathbf{C}$$