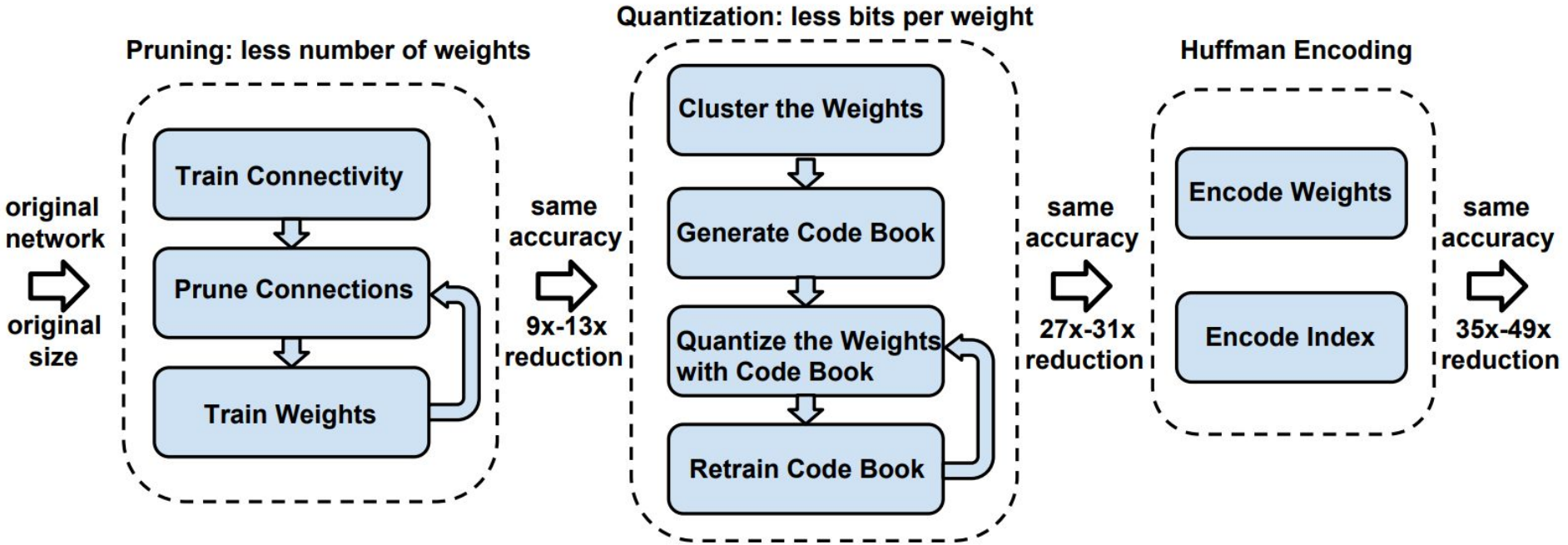
Networks compression. Pruning, trained quantization, matrix factorization.

Дроздова Анастасия, 181

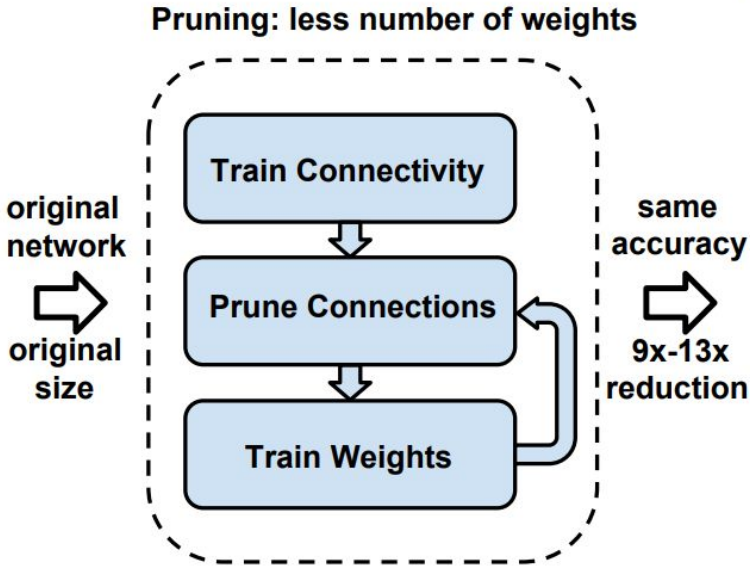
Зачем сжимать нейросети?

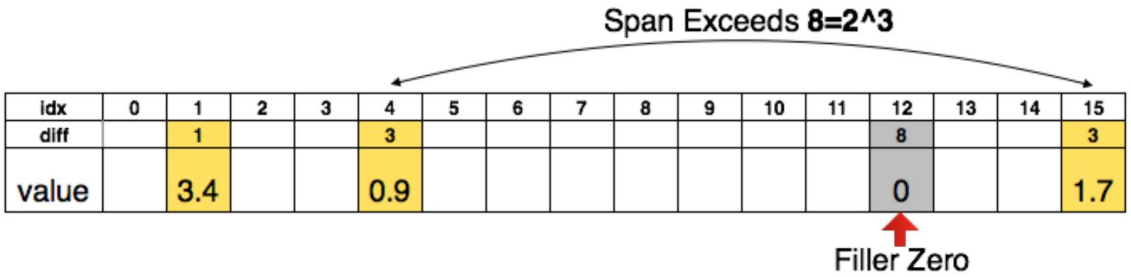
- нужно много вычислений и памяти
- глубокие нейронные сети на мобильных устройствах

Pipeline


Pruning: less number of weights
Train Connectivity
Prune Connections
Train Weights
original network
original size
same accuracy
9x-13x reduction
Quantization: less bits per weight
Cluster the Weights
Generate Code Book
Quantize the Weights with Code Book
Retrain Code Book
same accuracy
27x-31x reduction
Huffman Encoding
Encode Weights
Encode Index
same accuracy
35x-49x reduction

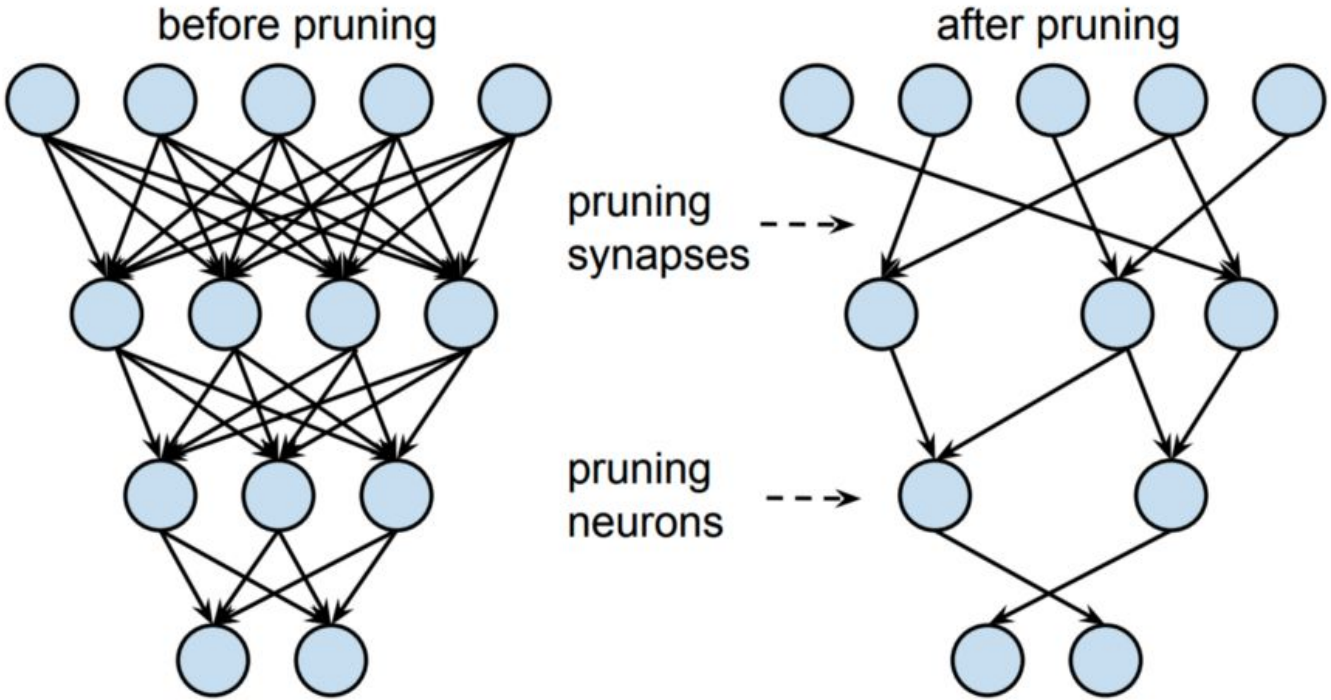
Pruning


Pruning: less number of weights
Train Connectivity
Prune Connections
Train Weights
original network
original size
same accuracy
9x-13x reduction

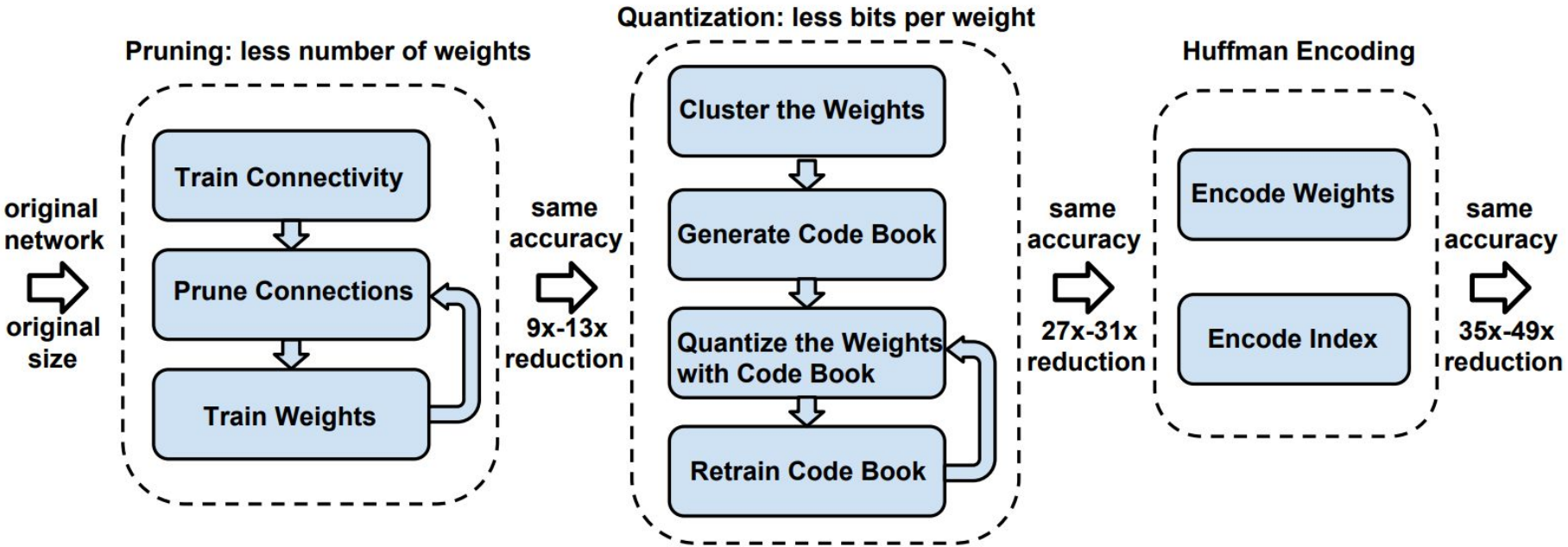

Span Exceeds 8=2^3
idx 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15
diff 1 3 8 3
value 3.4 0.9 0 1.7
Filler Zero

Pruning

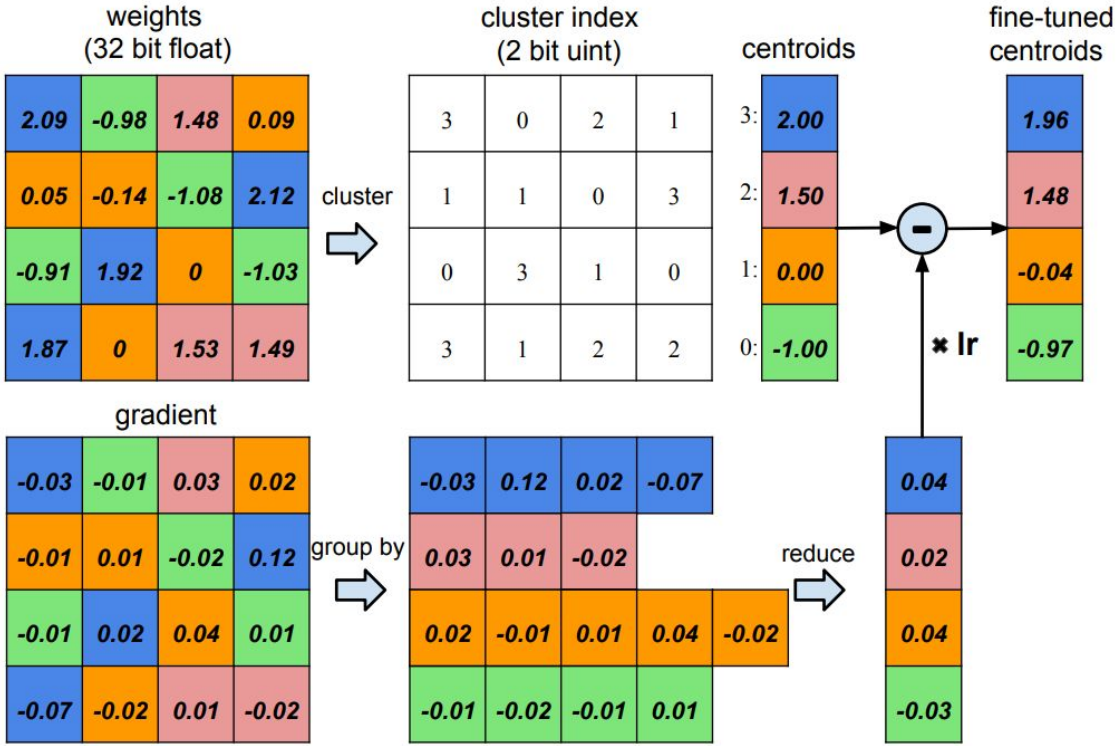
- удаление весов
- удаление нейронов
  - по весу
  - по выходу
  - похожие



Trained Quantization and Weight Sharing



Trained Quantization and Weight Sharing



k кластеров - $\log_2$ k бит

n связей, b бит на каждую,
k shared weights

compression rate =

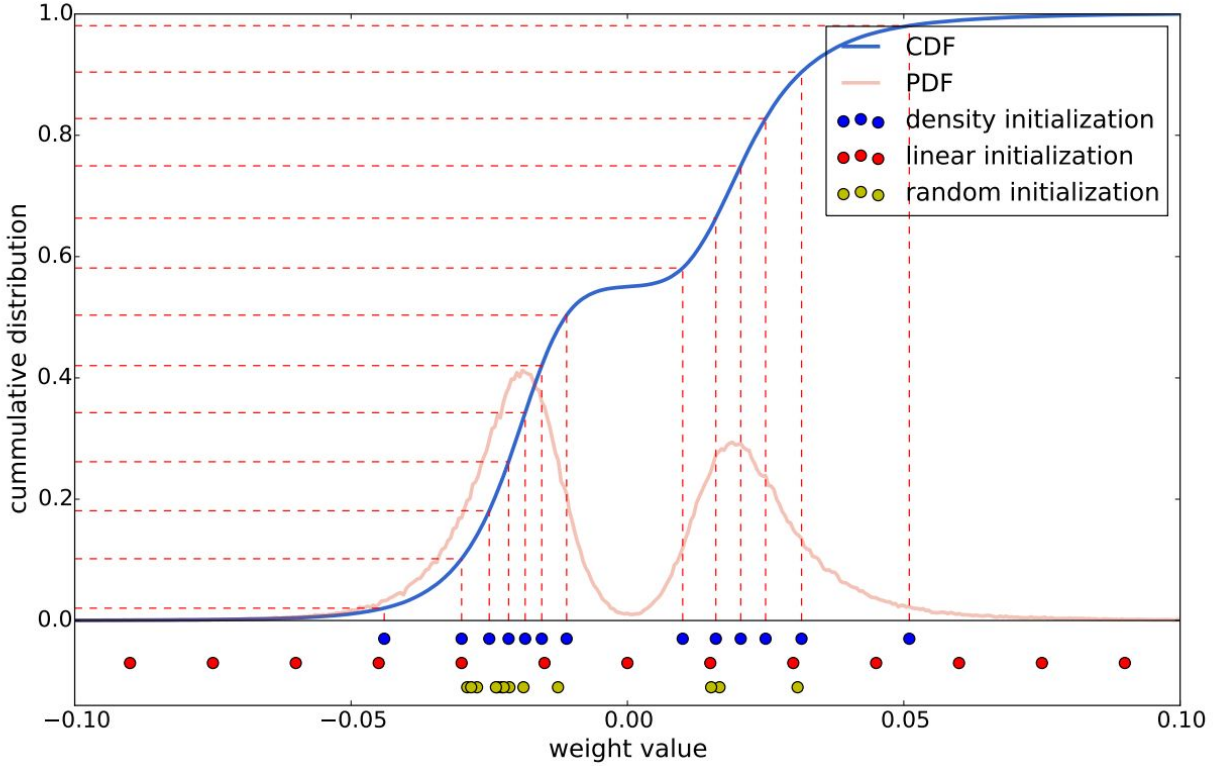
$$\frac{nb}{nlog_2(k)+kb}$$

Clustering. Centroid initialization

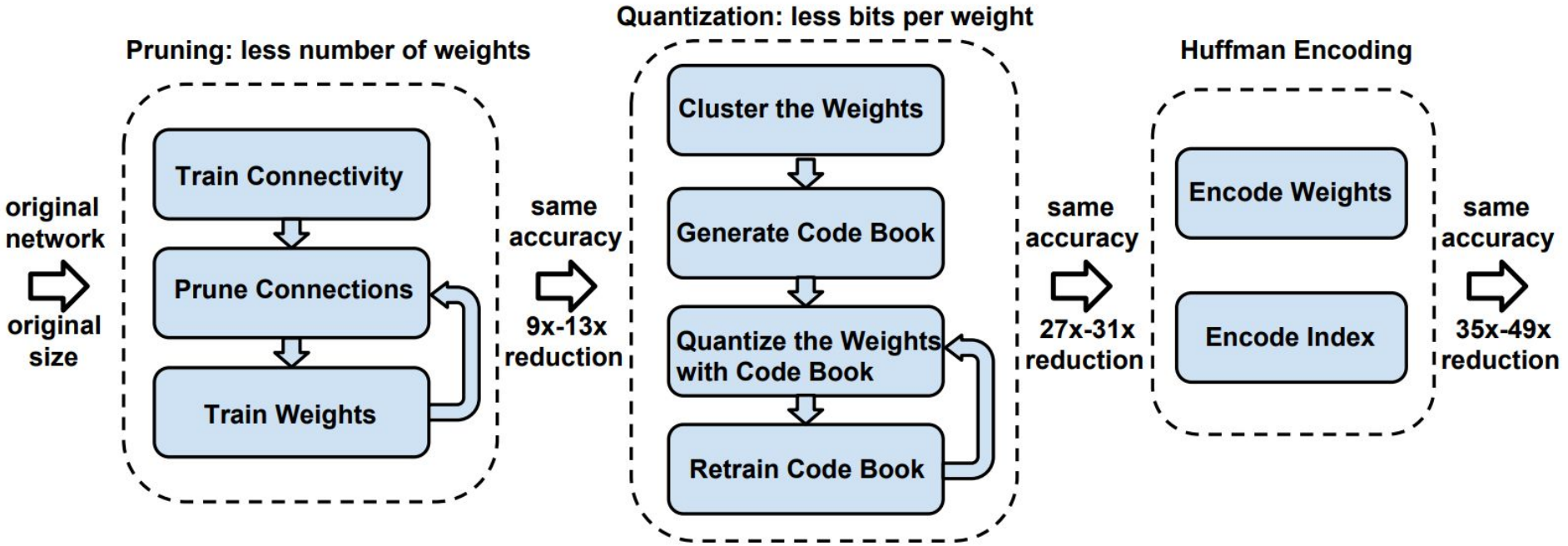
- k-means

$$\arg\min_{C} \sum_{i=1}^{k} \sum_{w \in c_i} |w - c_i|^2$$

- Forgy
- Density-based
- Linear



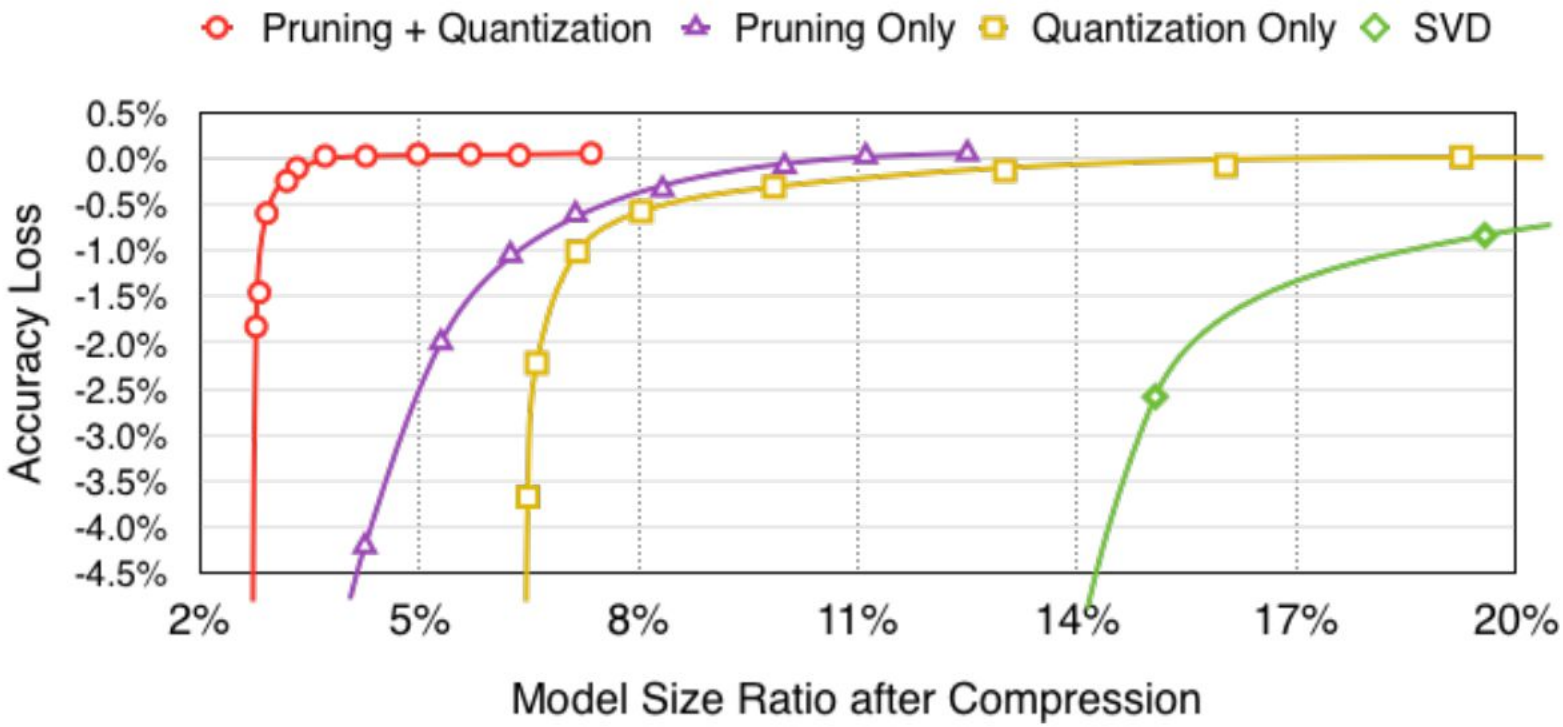
Huffman coding



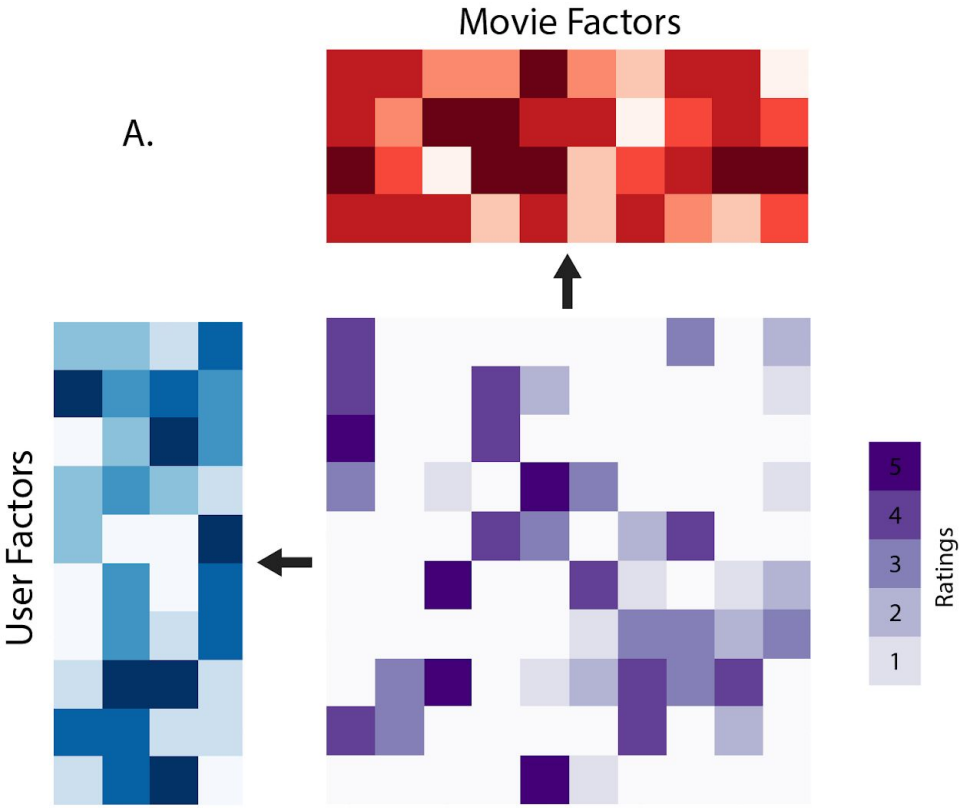
Experiments

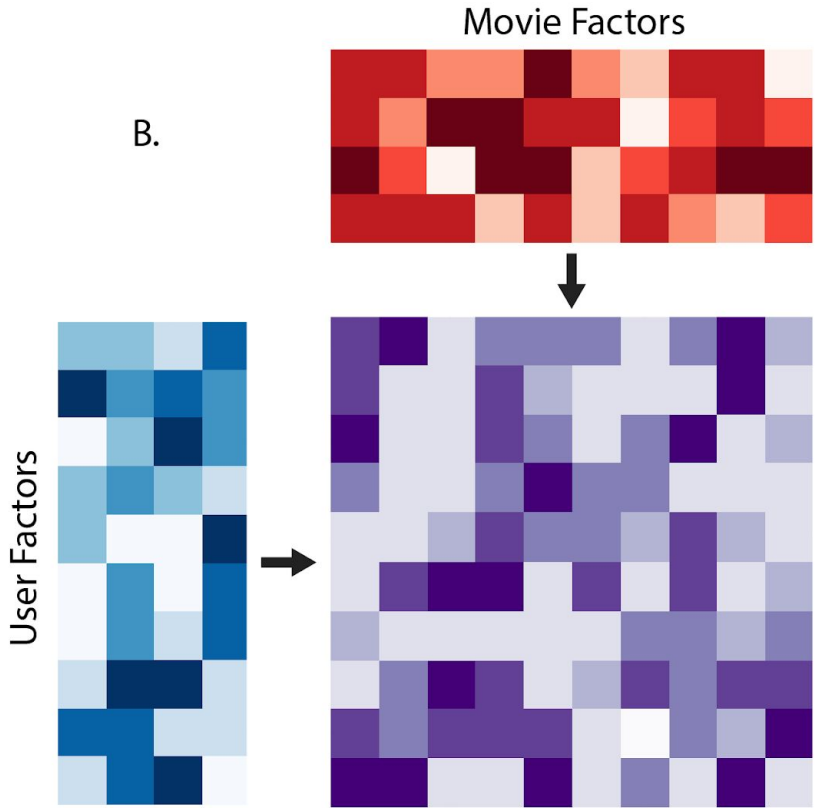
Network	Top-1 Error	Top-5 Error	Parameters	Compress Rate
LeNet-300-100 Ref	1.64%	-	1070 KB	
LeNet-300-100 Compressed	1.58%	-	**27 KB**	**40**×
LeNet-5 Ref	0.80%	-	1720 KB	
LeNet-5 Compressed	0.74%	-	**44 KB**	**39**×
AlexNet Ref	42.78%	19.73%	240 MB	
AlexNet Compressed	42.78%	19.70%	**6.9 MB**	**35**×
VGG-16 Ref	31.50%	11.32%	552 MB	
VGG-16 Compressed	31.17%	10.91%	**11.3 MB**	**49**×

Experiments



Matrix factorization





Neural Network Matrix factorization(NNMF)

- N x M data array $X_{n,m}$
- latent features $U_n \in \mathbb{R}^D \quad U'_n \in \mathbb{R}^{D' \times K}$

$$\hat{X}_{n,m} := \hat{X}(U_n, V_m, U'_n, V'_m) := f_\theta\big(U_n, V_m, U'_{n,1} \circ V'_{m,1}, \ldots, U'_{n,D'} \circ V'_{m,D'}\big)$$

$$\sum_{(n,m)\in J} (X_{n,m} - \hat{X}_{n,m})^2 + \lambda\Big[\sum_n ||U'_n||_F^2 + \sum_n ||U_n||_2^2 + \sum_m ||V'_m||_F^2 + \sum_m ||V_m||_2^2\Big]$$

Experiments

	NIPS	Protein	ML100k	ML1m
Vertices X	234	230	943	6040
Vertices Y	-	-	1682	3900
Edges	27144	52900	100000	1000209

Table 1: Data sets and their dimensions. The mark "-" highlights that the array is square.

Experiments

	NIPS	Protein	ML-100K
RFM (3)	0.110	0.136	-
PMF (3)	0.130	0.139	-
PMF (60)	0.062	0.104	0.952
BiasedMF (60)	0.065	0.111	0.911
NTN (60)	0.048	0.071	0.910
NNMF (3HL)	0.040	0.065	0.907
NNMF (4HL)	-	-	0.903

	ML-1M
PMF (60)	0.883
LLORMA-GLOBAL	0.865
I-RBM	0.854
BiasedMF (60)	0.852
NTN (60)	0.852
LLORMA-LOCAL	0.833
I-AutoRec	0.831
NNMF (3HL)	0.846
NNMF (4HL)	0.843

Table 2: Results across the four data sets for a variety of techniques. The token (D) specifies that a rank-D factorization was used. The token $(n\text{HL})$ specifies that n hidden layers were used.

Источники

- https://arxiv.org/pdf/1510.00149.pdf
- https://arxiv.org/pdf/1511.06443.pdf
- https://www.oreilly.com/content/deep-matrix-factorization-using-apache-mxnet/
- https://towardsdatascience.com/pruning-neural-networks-1bb3ab5791f9