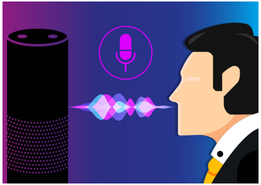
A Unified Approximation Framework for Compressing and Accelerating Deep Neural Networks

Yuzhe Ma[1], Ran Chen[1], Wei Li[1], Fanhua Shang[2],
Wenjian Yu[3], Minsik Cho[4], Bei Yu[1]

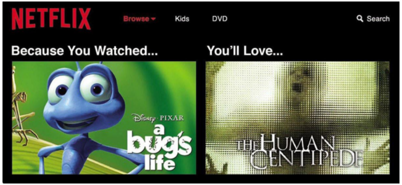
[1]CUHK, [2]Xidian Univ., [3]Tsinghua Univ. [4]IBM T. J. Watson

Introduction

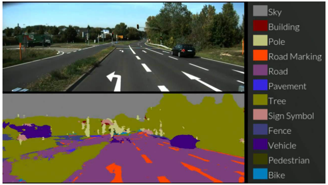
- Deep neural networks keep setting new records;
- More and more difficult tasks;
- The change on models?



Virtual Assistant



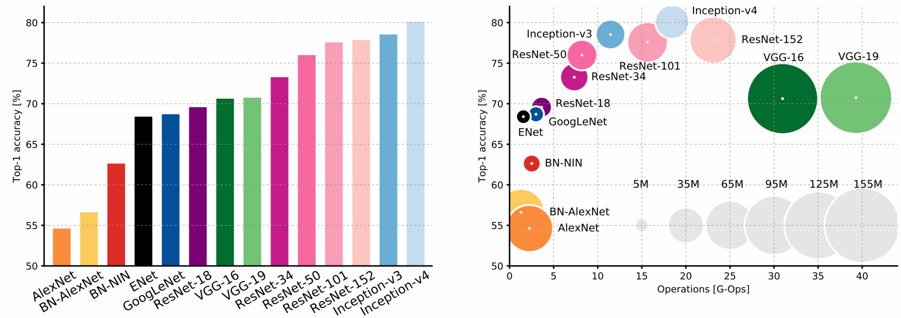
Recommendation System



Self-driving Cars

Trend on the Models

- Performance is getting better;
- Models are going deeper;
- Size is growing larger;
- Would this be a problem?

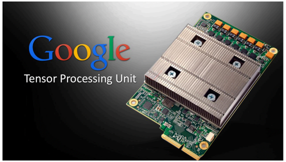


[1]

[1] Alfredo Canziani, Adam Paszke, and Eugenio Culurciello (2016). "An analysis of deep neural network models for practical applications". In: *arXiv preprint arXiv:1605.07678*.

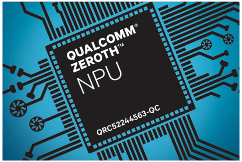
Challenges

- More applications need to be deployed on end-point devices.
- Smartphones
- Drones
- Cameras

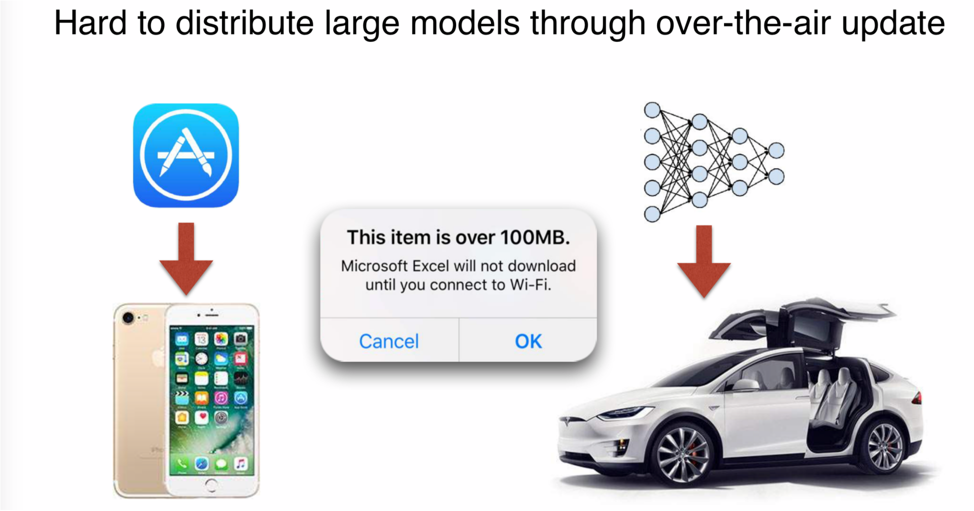


MOBILEYE®
An intel company

DEEPHI TECH



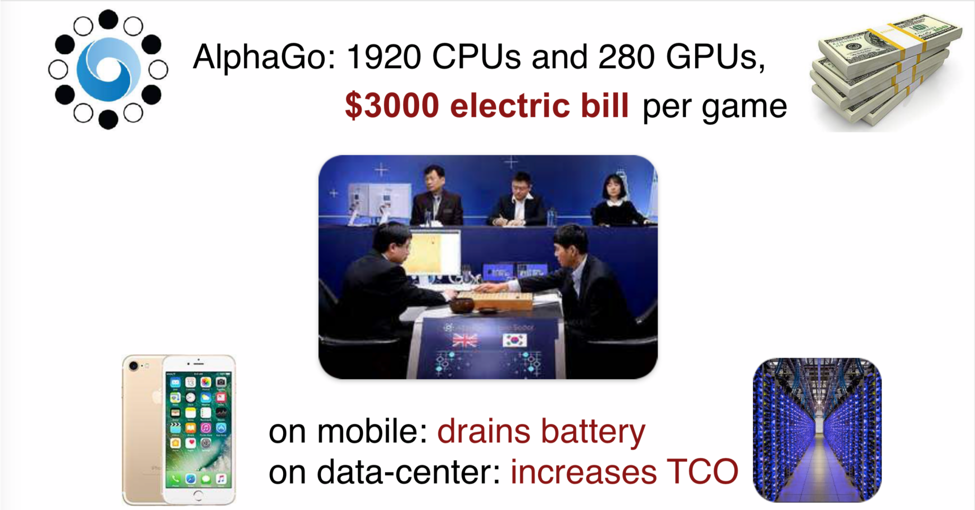
Model Size



[2]

[2] Song Han and William J Dally (2018). “Bandwidth-efficient deep learning”. In: *Proc. DAC*, pp. 1–6.

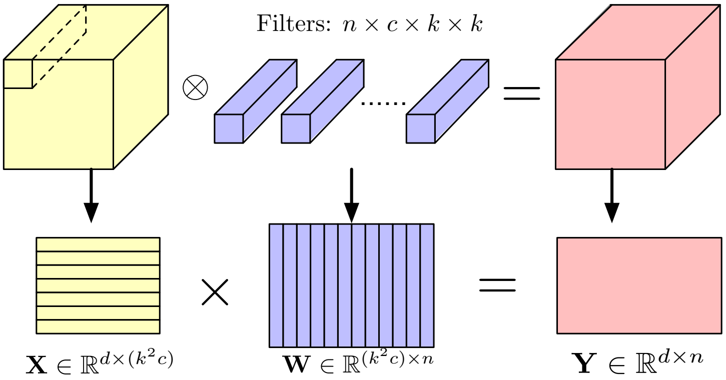
Energy Efficiency



[3]

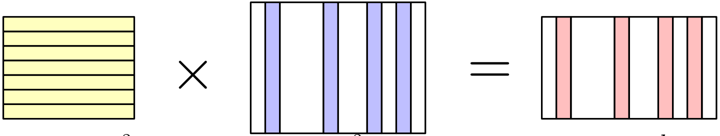
[3] Song Han and William J Dally (2018). “Bandwidth-efficient deep learning”. In: *Proc. DAC*, pp. 1–6.

Im2col (Image2Column) Convolution



- Transform convolution to matrix multiplication
- Unified calculation for both convolution and fully-connected layers

Property: Sparsity[4,5]



$\mathbf{X} \in \mathbb{R}^{d \times (k^2 c)}$ $\times$ $\mathbf{S} \in \mathbb{R}^{(k^2 c) \times n}$ $=$ $\mathbf{Y} \in \mathbb{R}^{d \times n}$

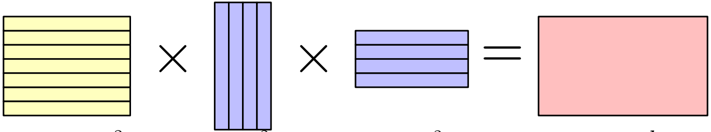
Sparse DNN

- *Sparsification*: weight pruning;
- *Compression*: compressed sparse format for storage;
- *Potential acceleration*: sparse matrix multiplication algorithm.

[4] Wei Wen et al. (2016). "Learning structured sparsity in deep neural networks". In: *Proc. NIPS*, pp. 2074–2082.

[5] Yihui He, Xiangyu Zhang, and Jian Sun (2017). "Channel Pruning for Accelerating Very Deep Neural Networks". In: *Proc. ICCV*.

Property: Low-Rank[6,7]



$\mathbf{X} \in \mathbb{R}^{d\times(k^2c)}$ $\mathbf{U} \in \mathbb{R}^{(k^2c)\times r}$ $\mathbf{V} \in \mathbb{R}^{1^2 r\times n}$ $\mathbf{Y} \in \mathbb{R}^{d\times n}$

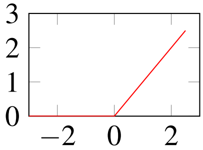
Low-rank DNN

- *Low-rank approximation*: matrix decomposition or tensor decomposition.
- *Compression and acceleration*: less storage required and less FLOP in computation.

[6] Xiangyu Zhang et al. (2015). “Efficient and accurate approximations of nonlinear convolutional networks”. In: *Proc. CVPR*, pp. 1984–1992.

[7] Xiyu Yu et al. (2017). “On compressing deep models by low rank and sparse decomposition”. In: *Proc. CVPR*, pp. 7370–7379.

Non-linearity Approximation[8]



ReLU

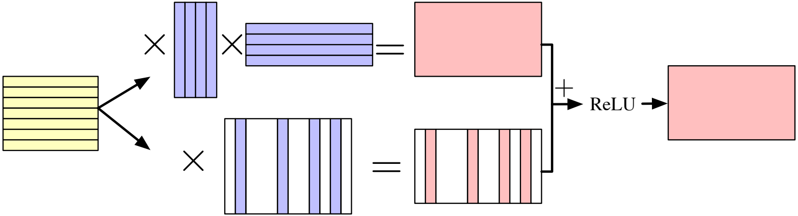
- Analyze the output error caused by approximation
- Activation unit: `ReLU`
- Error more sensitive to positive response;
- Enlarge the solution space.

$$\min_{\boldsymbol{W}} \sum_{i=1}^{N} \|\boldsymbol{W}\boldsymbol{X}_i - \boldsymbol{Y}_i\|_F \rightarrow \min_{\boldsymbol{W}} \sum_{i=1}^{N} \|r(\boldsymbol{W}\boldsymbol{X}_i) - \boldsymbol{Y}_i\|_F$$

- $\boldsymbol{X}$: input feature map
- $\boldsymbol{Y}$: output feature map

[8] Xiangyu Zhang et al. (2015). “Efficient and accurate approximations of nonlinear convolutional networks”. In: *Proc. CVPR*, pp. 1984–1992.

Our Idea: Unified Structure



- Simultaneous low-rank approximation and network sparsification;
- Non-linearity is taken into account;
- Acceleration is achieved with structured sparsity;
- Flexibility between two properties.

Formulation

Given a pre-trained network, the goal is to minimize the reconstruction error of the response in each layer after activation using sparse component and low-rank component.

$$\min_{\boldsymbol{A},\boldsymbol{B}} \sum_{i=1}^{N} \|\boldsymbol{Y}_i - r((\boldsymbol{A}+\boldsymbol{B})\boldsymbol{X}_i)\|_F \,,$$
$$\text{s.t.} \quad \|\boldsymbol{A}\|_0 \le S,$$
$$\text{rank}(\boldsymbol{B}) \le L.$$

- $\boldsymbol{X}$: input feature map
- $\boldsymbol{Y}$: output feature map

Not easy to solve: l_0 minimization and rank minimization are both NP-hard.

Relaxation

$$\min_{\boldsymbol{A},\boldsymbol{B}} \sum_{i=1}^{N} \|\boldsymbol{Y}_i - r((\boldsymbol{A}+\boldsymbol{B})\boldsymbol{X}_i)\|_F^2 + \lambda_1 \|\boldsymbol{A}\|_{2,1} + \lambda_2 \|\boldsymbol{B}\|_*$$

- The l_0 constraint is relaxed by $l_{2,1}$ norm such that the zero elements in $\boldsymbol{A}$ appear column-wise;
- The rank constraint on $\boldsymbol{B}$ is relaxed by nuclear norm of $\boldsymbol{B}$, which is the sum of the singular values;
- Apply alternating direction method of multipliers (ADMM) to solve it;

Alternating Direction Method of Multipliers (ADMM)

Reformulating the problem with an auxiliary variable $\boldsymbol{M}$,

$$\min_{\boldsymbol{A},\boldsymbol{B},\boldsymbol{M}} \sum_{i=1}^{N} \|\boldsymbol{Y}_i - r(\boldsymbol{M}\boldsymbol{X}_i)\|_F^2 + \lambda_1 \|\boldsymbol{A}\|_{2,1} + \lambda_2 \|\boldsymbol{B}\|_* ,$$
$$\text{s.t. } \boldsymbol{A} + \boldsymbol{B} = \boldsymbol{M}.$$

Then the augmented Lagrangian function is

$$L_t(\boldsymbol{A}, \boldsymbol{B}, \boldsymbol{M}, \boldsymbol{\Lambda})$$
$$= \sum_{i=1}^{N} \|\boldsymbol{Y}_i - r(\boldsymbol{M}\boldsymbol{X}_i)\|_F^2 + \lambda_1 \|\boldsymbol{A}\|_{2,1} + \lambda_2 \|\boldsymbol{B}\|_* + \langle \boldsymbol{\Lambda}, \boldsymbol{A} + \boldsymbol{B} - \boldsymbol{M} \rangle + \frac{t}{2} \|\boldsymbol{A} + \boldsymbol{B} - \boldsymbol{M}\|_F^2$$

Alternating Direction Method of Multipliers (ADMM)

Iteratively solve with following rules. All of them can be solved efficiently.

$$
\begin{cases}
\boldsymbol{A}_{k+1} = \underset{\boldsymbol{A}}{\operatorname{argmin}}\ \lambda_1 \|\boldsymbol{A}\|_{2,1} + \frac{t}{2} \left\| \boldsymbol{A} + \boldsymbol{B}_k - \boldsymbol{M}_k + \frac{\boldsymbol{\Lambda}_k}{t} \right\|_F^2, \\
\boldsymbol{B}_{k+1} = \underset{\boldsymbol{B}}{\operatorname{argmin}}\ \lambda_2 \|\boldsymbol{B}\|_* + \frac{t}{2} \left\| \boldsymbol{B} + \boldsymbol{A}_{k+1} - \boldsymbol{M}_k + \frac{\boldsymbol{\Lambda}_k}{t} \right\|_F^2, \\
\boldsymbol{M}_{k+1} = \underset{\boldsymbol{M}}{\operatorname{argmin}} \sum_{i=1}^{N} \|\boldsymbol{Y}_i - r(\boldsymbol{M}\boldsymbol{X}_i)\|_F^2 + \langle \boldsymbol{\Lambda}_k, \boldsymbol{A}_{k+1} + \boldsymbol{B}_{k+1} - \boldsymbol{M} \rangle + \frac{t}{2} \|\boldsymbol{A}_{k+1} + \boldsymbol{B}_{k+1} - \boldsymbol{M}\|_F^2, \\
\boldsymbol{\Lambda}_{k+1} = \boldsymbol{\Lambda}_k + t(\boldsymbol{A}_{k+1} + \boldsymbol{B}_{k+1} - \boldsymbol{M}_{k+1}).
\end{cases}
$$

Solving $l_{2,1}$-norm

$$\min_{\boldsymbol{A}} \lambda_1 \|\boldsymbol{A}\|_{2,1} + \frac{t}{2}\left\|\boldsymbol{A} + \boldsymbol{B}_k - \boldsymbol{M}_k + \frac{\boldsymbol{\Lambda}_k}{t}\right\|_F^2$$

Closed Form Update Rule[9]

$$\boldsymbol{A}_{k+1} = \text{prox}_{\frac{\lambda_1}{t}\|\cdot\|_{2,1}}(\boldsymbol{M}_k - \boldsymbol{B}_k - \frac{\boldsymbol{\Lambda}_k}{t}),$$

$$\boldsymbol{C} = \boldsymbol{M}_k - \boldsymbol{B}_k - \frac{\boldsymbol{\Lambda}_k}{t},$$

$$[\boldsymbol{A}_{k+1}]_{:,i} = \begin{cases} \dfrac{\|[\boldsymbol{C}]_{:,i}\|_2 - \frac{\lambda_1}{t}}{\|[\boldsymbol{C}]_{:,i}\|_2}[\boldsymbol{C}]_{:,i}, & \text{if } \|[\boldsymbol{C}]_{:,i}\|_2 > \dfrac{\lambda_1}{t}; \\ 0, & \text{otherwise.} \end{cases}$$

[9] Guangcan Liu et al. (2013). "Robust recovery of subspace structures by low-rank representation". In: *IEEE TPAMI* 35.1, pp. 171–184.

Solving Nuclear-norm

$$\min_{\boldsymbol{B}} \lambda_2 \|\boldsymbol{B}\|_* + \frac{t}{2} \left\| \boldsymbol{B} + \boldsymbol{A}_{k+1} - \boldsymbol{M}_k + \frac{\boldsymbol{\Lambda}_k}{t} \right\|_F^2$$

Closed Form Update Rule[10]

$$\boldsymbol{B}_{k+1} = \text{prox}_{\frac{\lambda_2}{t}\|\cdot\|_*}(\boldsymbol{M}_k - \boldsymbol{A}_{k+1} - \frac{\boldsymbol{\Lambda}_k}{t}),$$

$$\boldsymbol{D} = \boldsymbol{M}_k - \boldsymbol{A}_{k+1} - \frac{\boldsymbol{\Lambda}_k}{t},$$

$$\boldsymbol{B}_{k+1} = \boldsymbol{U}\mathcal{D}_{\frac{\lambda_2}{t}}(\boldsymbol{\Sigma})\boldsymbol{V}, \quad \text{where } \mathcal{D}_{\frac{\lambda_2}{t}}(\boldsymbol{\Sigma}) = \text{diag}(\{(\sigma_i - \frac{\lambda_2}{t})_+\}).$$

[10] Jian-Feng Cai, Emmanuel J Candès, and Zuowei Shen (2010). "A singular value thresholding algorithm for matrix completion". In: *SIAM Journal on Optimization (SIOPT)* 20.4, pp. 1956–1982.

Solving M

$$\min_{\boldsymbol{M}} \sum_{i=1}^{N} \|\boldsymbol{Y}_i - r(\boldsymbol{M}\boldsymbol{X}_i)\|_F^2 + \langle \boldsymbol{\Lambda}_k, \boldsymbol{A}_{k+1} + \boldsymbol{B}_{k+1} - \boldsymbol{M} \rangle + \frac{t}{2} \|\boldsymbol{A}_{k+1} + \boldsymbol{B}_{k+1} - \boldsymbol{M}\|_F^2$$

Gradient-based optimization

- Can be solved using first-order condition, but computing matrix inverse in each iteration is expensive.
- Convex problem. Use SGD to solve it efficiently.
- GPU can accelerate the process.

Comparison on *CIFAR-10* dataset

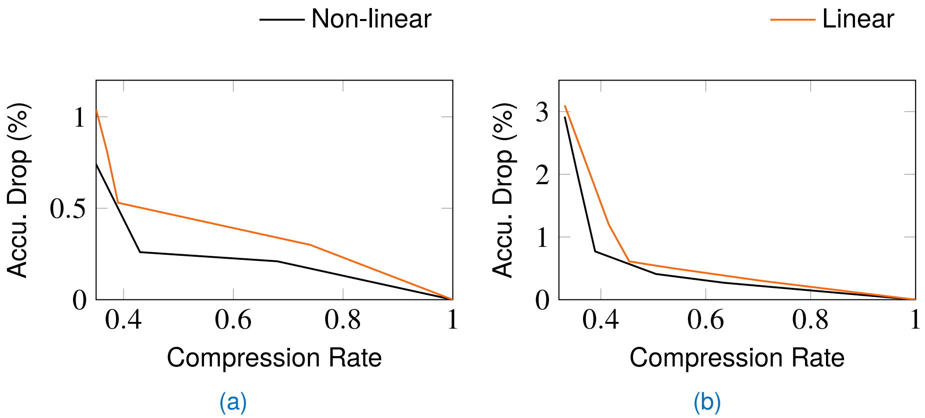
Model	Method	Accuracy ↓	CR	Speed-up
VGG-16	Original	0.00%	1.00	1.00
	ICLR'17[11]	**0.06%**	2.70	1.80
	Ours	0.40%	**4.44**	**2.20**
NIN	Original	0.00%	1.00	1.00
	ICLR'16[12]	1.43%	1.54	1.50
	IJCAI'18[13]	1.43%	1.45	-
	Ours	**0.41%**	**2.77**	**1.70**

[11] Hao Li et al. (2017). "Pruning filters for efficient convnets". In: *Proc. ICLR*.

[12] Cheng Tai et al. (2016). "Convolutional neural networks with low-rank regularization". In: *Proc. ICLR*.

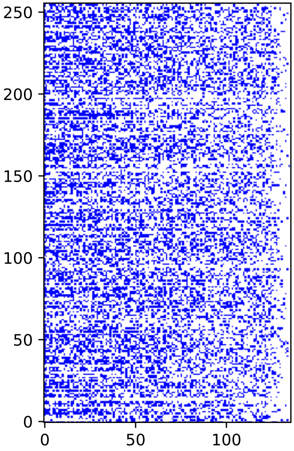
[13] Shiva Prasad Kasiviswanathan, Nina Narodytska, and Hongxia Jin (2018). "Network Approximation using Tensor Sketching". In: *Proc. IJCAI*, pp. 2319–2325.

Linear vs. Non-linear

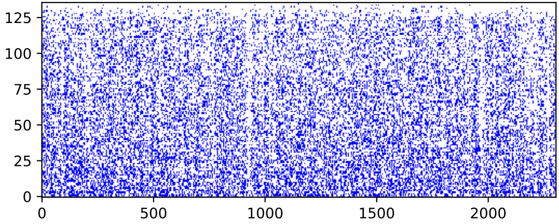


Comparison of reconstructing linear response and non-linear response: (a) layer `conv2-1`; (b) layer `conv3-1`.

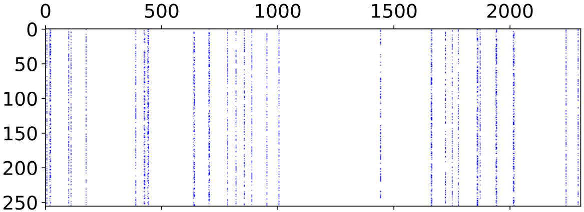
Approximation Example



(a)



(b)



(c)

Comparison on *ImageNet* dataset

Model	Method	Top-5 Accu.↓	CR	Speed-up
AlexNet	Original	0.00%	1.00	1.00
	ICLR'16[14]	**0.37%**	5.00	**1.82**
	ICLR'16[15]	1.70%	5.46	1.81
	CVPR'18[16]	1.43%	1.50	-
	Ours	1.27%	**5.56**	1.10
GoogleNet	Original	0.00%	1.00	1.00
	ICLR'16[11]	0.42%	2.84	1.20
	ICLR'16[12]	0.24%	1.28	1.23
	CVPR'18[23]	0.21%	1.50	-
	Ours	**0.00%**	**2.87**	**1.35**

[14] Cheng Tai et al. (2016). "Convolutional neural networks with low-rank regularization". In: *Proc. ICLR*.

[15] Yong-Deok Kim et al. (2016). "Compression of deep convolutional neural networks for fast and low power mobile applications". In: *Proc. ICLR*.

[16] Ruichi Yu et al. (2018). "NISP: Pruning networks using neuron importance score propagation". In: *Proc. CVPR*.

Conclusion

- A unified model for compressing the deep neural networks with low-rank approximation and network sparsification, while taking non-linearity into consideration.
- ADMM is applied to solve the problem, which can be proved to converge to the optimal solution of the relaxed problem.
- $5\times$ compression and more than $2\times$ speedup is achieved with less accuracy loss.
- Flexibility is provided to choose different network architectures by setting different penalty weights.