Some slides referred from a lecture note of
George Mason University (Shuochao Yao, Yiwen Xu, Daniel Calzada)

ML Coding Practice
Lecture 02-2

Network Pruning & Quantization

Prof. Jongwon Choi
Chung-Ang University
Fall 2022

Today's Lecture

- Matrix Factorization (SVD)

- Network Pruning

- Network Quantization

- Network Pruning + Quantization

Eigenvalue & Eigenvector

- **Definition**
  - $\mathbf{A} \in \mathbb{R}^{n \times n}$ is a square matrix
  - Then, $\lambda \in \mathbb{R}$ is an eigenvalue of **A** and $\mathbf{x} \in \mathbb{R}^{n} \backslash \{0\}$ is the corresponding eigenvector of **A** if
  - $\mathbf{A}\mathbf{x} = \lambda\mathbf{x}$
    - This is called "Eigenvalue equation"

- Conventionally,
  - eigenvalues are sorted in descending order

Eigenvalue & Eigenvector

- **Equivalent Statements**
  - λ is an eigenvalue of $\mathbf{A} \in \mathbb{R}^{n \times n}$
  - There exists an $\mathbf{x} \in \mathbb{R}^n \backslash \{0\}$ with $\mathbf{Ax} = \lambda\mathbf{x}$
  - $(\mathbf{A} - \lambda\mathbf{I}_n)\mathbf{x} = 0$ can be solved non-trivially (i.e. $\mathbf{x} \neq 0$)
  - $rk(\mathbf{A} - \lambda\mathbf{I}_n) < n$
  - $\det(\mathbf{A} - \lambda\mathbf{I}_n) = 0$

Properties of e-val. & e-vec.

- **Properties**
  - $\mathbf{A}$ and $\mathbf{A}^T$ have the same e-val, but not necessarily the same e-vec
  - The eigenspace E_λ is the null space of $\mathbf{A} - \lambda\mathbf{I}_n$
    - pf. $\mathbf{Ax} = \lambda\mathbf{x}$ => $\mathbf{Ax} - \lambda\mathbf{x} = 0$ => $(\mathbf{A} - \lambda\mathbf{I})\mathbf{x} = 0$ => $\mathbf{x} \in \ker(\mathbf{A} - \lambda\mathbf{I})$
  - Similar matrices have the same e-val
    - Thus, e-val is independent of the choice of basis of the transformation matrices
    - e-val, determinant, trace are the key characteristic parameters of a linear mapping, which are invariant under basis changes!
  - Symmetric, positive definite matrices always have positive, real e-vals.

Linear independence & e-val

- The eigenvectors $x_1, \dots, x_n$ of a matrix $A \in \mathbb{R}^{n \times n}$ with n distinct eigenvalues $\lambda_1, \dots, \lambda_n$ are linearly independent

- **Defective**
  - A square matrix $A \in \mathbb{R}^{n \times n}$ is defective if it possesses fewer than n linearly independent e-vecs.
  - Remark!
    - A non-defective matrix does not necessarily require n distinct e-vals.

Determinant & Trace & e-vals

- The determinant of a matrix $\mathbf{A} \in \mathbb{R}^{n \times n}$ is the product of its e-vals:
  - $\det(\mathbf{A}) = \prod_{i=1}^{n} \lambda_i$
  - where $\lambda \in \mathbb{C}$ are (possibly repeated) e-vals of **A**
- The trace of $\mathbf{A} \in \mathbb{R}^{n \times n}$ is the sum of its e-vals:
  - $\mathrm{tr}(\mathbf{A}) = \sum_{i=1}^{n} \lambda_i$
  - where $\lambda \in \mathbb{C}$ are (possibly repeated) e-vals of **A**

Cholesky Decomposition

- **Necessity**
  - The square-root operation in matrix
  - In the case of single number, the square-root operation is only for the positive number
- **Definition**
  - A square-root equivalent operation
  - Only possible for symmetric, positive definite matrices

Cholesky Decomposition

- **Definition**
  - A square-root equivalent operation
  - Only possible for symmetric, positive definite matrices

  - For a symmetric, positive definite matrix A,
  - **A** can be factorized into a product $\mathbf{A} = \mathbf{L}\mathbf{L}^T$
  - , where **L** is a lower-triangular matrix with positive diagonal elements
  - $\begin{bmatrix} a_{11} & \dots & a_{1n} \\ \dots & \dots & \dots \\ a_{n1} & \dots & a_{nn} \end{bmatrix} = \begin{bmatrix} l_{11} & \dots & 0 \\ \dots & \dots & \dots \\ l_{n1} & \dots & l_{nn} \end{bmatrix} \begin{bmatrix} l_{11} & \dots & l_{n1} \\ \dots & \dots & \dots \\ 0 & \dots & l_{nn} \end{bmatrix}$

Cholesky Decomposition

- **Interesting Property**
  - We can compute determinants very efficiently!
  - $\det(\mathbf{A}) = \det(\mathbf{L})\det(\mathbf{L}^T) = \det(\mathbf{L})^2$
  - Here, since L is a triangular matrix,
  - $\det(\mathbf{L}) = \prod_{i=1}^{n} l_{ii}$
  - Thus, $\det(\mathbf{A}) = \det(\mathbf{L})^2 = \prod_{i=1}^{n} l_{ii}^2$

Diagonal Matrix

- **Definition**
  - A matrix has value 0 on all off-diagonal elements
  - $D = \begin{bmatrix} c_1 & \dots & 0 \\ \dots & \dots & \dots \\ 0 & \dots & c_n \end{bmatrix}$

- **Notation**
  - $D = \begin{bmatrix} c_1 & \dots & 0 \\ \dots & \dots & \dots \\ 0 & \dots & c_n \end{bmatrix} = diag([c_1, \dots, c_n])$

Diagonal Matrix

- **Properties**
  - Fast computation of determinants, powers, and inverses!

  - Determinant
    - $\det(\mathbf{D}) = \prod_{i=1}^{n} c_i$

  - Power
    - $\mathbf{D}^{\mathbf{k}} = diag\left(\left[c_1^k, \dots, c_n^k\right]\right)$

  - Inverse (Only if only the non-zero values are diagonal)
    - $\mathbf{D}^{-\mathbf{1}} = diag\left(\left[\frac{1}{c_1}, \dots, \frac{1}{c_n}\right]\right)$

Diagonalizable

- **e-val & e-vec & Diagonalizable**
  - $\mathbf{A} \in \mathbb{R}^{n\times n}$
  - $\lambda_1, \dots, \lambda_n \in \mathbb{R}^1$
  - $\mathbf{p}_1, \dots, \mathbf{p}_n \in \mathbb{R}^n$

  - Then, $\mathbf{AP} = \mathbf{PD}$
  - iff λ_i are e-val of **A** and $\mathbf{p}_i$ are the corresponding e-vec

Eigendecomposition

- A square matrix $\mathbf{A} \in \mathbb{R}^{n \times n}$ can be factored into
  - $\mathbf{A} = \mathbf{PDP}^{-1}$
  - , where $\mathbf{P} \in \mathbb{R}^{n \times n}$, $\mathbf{D} = diag([\lambda_1, \dots, \lambda_n])$, and λ_i are the e-val of **A**
- iff the e-val of A form a basis of $\mathbb{R}^n$

Computation Efficiency

- **Power of A**
  - $\mathbf{A}^k = (\mathbf{PDP}^{-1})^k = \mathbf{PD}^k\mathbf{P}^{-1}$

- **Determinant of A**
  - $\det(\mathbf{A}) = \det(\mathbf{PDP}^{-1}) = \det(\mathbf{P})\det(\mathbf{D})\det(\mathbf{P}^{-1}) = \det(\mathbf{D}) = \prod_{i=1}^{n} d_{ii}$

Limitation of e-val decomposition

Only works for the square matrix!

- **Cholesky Decomposition**
  - For a symmetric, positive definite matrix A,
  - **A** can be factorized into a product $\mathbf{A} = \mathbf{L}\mathbf{L}^T$
  - , where **L** is a lower-triangular matrix with positive diagonal elements
  - $\begin{bmatrix} a_{11} & \cdots & a_{1n} \\ \cdots & \cdots & \cdots \\ a_{n1} & \cdots & a_{nn} \end{bmatrix} = \begin{bmatrix} l_{11} & \cdots & 0 \\ \cdots & \cdots & \cdots \\ l_{n1} & \cdots & l_{nn} \end{bmatrix} \begin{bmatrix} l_{11} & \cdots & l_{n1} \\ \cdots & \cdots & \cdots \\ 0 & \cdots & l_{nn} \end{bmatrix}$

- **Diagonalization**
  - A square matrix $\mathbf{A} \in \mathbb{R}^{n \times n}$ can be factored into
    - $\mathbf{A} = \mathbf{P}\mathbf{D}\mathbf{P}^{-1}$
    - , where $\mathbf{P} \in \mathbb{R}^{n \times n}$, $\mathbf{D} = diag([\lambda_1, \dots, \lambda_n])$, and λ_i are the e-val of **A**
  - iff the e-val of A form a basis of $\mathbb{R}^n$

SVD Theorem

- $\mathbf{A}_{m\times n}$ is a rectangular matrix of rank $r \in [0, \min(m, n)]$
- The SVD of **A** is a decomposition of the form
  - $\mathbf{A}_{m\times n} = \mathbf{U}_{m\times m}\mathbf{\Sigma}_{m\times n}\mathbf{V}_{n\times n}^{T}$

  - $\mathbf{U}_{m\times m}$: orthogonal matrix with column vectors $\mathbf{u}_i$
  - $\mathbf{V}_{n\times n}$: orthogonal matrix with column vectors $\mathbf{v}_i$
  - $\mathbf{\Sigma}_{m\times n}$: $\Sigma_{ii} = \sigma_i \geq 0$ and $\Sigma_{ij} = 0,\ i \neq j$

SVD Theorem

- $\mathbf{A}_{m\times n} = \mathbf{U}_{m\times m}\mathbf{\Sigma}_{m\times n}\mathbf{V}_{n\times n}^{T}$

- $\mathbf{U}_{m\times m}$: orthogonal matrix with column vectors $\mathbf{u}_i$
  - $\mathbf{u}_i$ are called the left-singular vectors
- $\mathbf{V}_{n\times n}$: orthogonal matrix with column vectors $\mathbf{v}_i$
  - $\mathbf{v}_i$ are called the right-singular vectors
- $\mathbf{\Sigma}_{m\times n}$: $\Sigma_{ii} = \sigma_i \geq 0$ and $\Sigma_{ij} = 0,\ i \neq j$
  - σ_i are called the "singular values"
  - Conventionally, the singular values are sorted, i.e. $\sigma_1 \geq \sigma_2 \geq \cdots \geq 0$
  - Unique!!

SVD – Mathematical Meaning

- SVD vs. Eigendecomposition
  - Eigendecomposition: $\mathbf{S} = \mathbf{S}^T = \mathbf{PDP}^T$ (Of course, S is SPD)
  - SVD: $\mathbf{S} = \mathbf{U\Sigma V}^{\mathrm{T}}$

- Then,
  - When $\mathbf{U} = \mathbf{P} = \mathbf{V}$, $\mathbf{D} = \mathbf{\Sigma}$
  - SVD of SPD matrices is their eigendecomposition!

SVD – Mathematical Meaning

- What's the direct connection between eigendecomposition and SVD
  - Without any constraint (i.e. SPD)

- $\mathbf{A}^T\mathbf{A} \in \mathbb{R}^{n \times n}$ is symmetric, positive semi-definite matrix
  - Any matrix $A \in \mathbb{R}^{m \times n}$

- The symmetric, positive semi-definite matrix can be diagonalized as:
  - $\mathbf{A}^T\mathbf{A} = \mathbf{PDP}^{\mathrm{T}} = \mathbf{P}\begin{bmatrix} \lambda_1 & \dots & 0 \\ \dots & \dots & \dots \\ 0 & \dots & \lambda_n \end{bmatrix}\mathbf{P}^T$
  - where **P** is an orthogonal matrix composed of the orthonormal eigenbasis
  - $\lambda_i \geq 0$

SVD – Mathematical Meaning

- $\mathbf{A}^T\mathbf{A} \in \mathbb{R}^{n\times n}$is symmetric, positive semi-definite matrix
- $\mathbf{A}^T\mathbf{A} = \mathbf{PDP}^{\mathrm{T}} = \mathbf{P}\begin{bmatrix} \lambda_1 & \dots & 0 \\ \dots & \dots & \dots \\ 0 & \dots & \lambda_n \end{bmatrix}\mathbf{P}^T$
- $\mathbf{A}^T\mathbf{A} = \left(\mathbf{U\Sigma V}^{\mathrm{T}}\right)^T\left(\mathbf{U\Sigma V}^{\mathrm{T}}\right) = \boldsymbol{V}\mathbf{\Sigma}^{\mathrm{T}}\mathbf{U}^{\mathrm{T}}\mathbf{U\Sigma V}^{\mathrm{T}}$
- Here, remark that **U**, **V** are the orthonormal matrices
- Thus, $\mathbf{U}^T\mathbf{U} = \mathbf{I}$

SVD – Mathematical Meaning

- $\mathbf{A}^T\mathbf{A} \in \mathbb{R}^{n \times n}$ is symmetric, positive semi-definite matrix
- $\mathbf{A}^T\mathbf{A} = \mathbf{PDP}^\mathrm{T} = \mathbf{P}\begin{bmatrix} \lambda_1 & \dots & 0 \\ \dots & \dots & \dots \\ 0 & \dots & \lambda_n \end{bmatrix}\mathbf{P}^T$
- $\mathbf{A}^T\mathbf{A} = \boldsymbol{V}\boldsymbol{\Sigma}^\mathrm{T}\mathbf{U}^\mathrm{T}\mathbf{U}\boldsymbol{\Sigma}\mathbf{V}^\mathrm{T} = \mathbf{V}\boldsymbol{\Sigma}^\mathrm{T}\boldsymbol{\Sigma}\mathbf{V}^\mathrm{T} = \mathbf{V}\begin{bmatrix} \sigma_1^2 & \dots & 0 \\ \dots & \dots & \dots \\ 0 & \dots & \sigma_n^2 \end{bmatrix}\mathbf{V}^\mathrm{T}$
- $\mathbf{P}^T = \mathbf{V}^T,\ \sigma_i^2 = \lambda_i$
- Thus, eigenvectors of $\mathbf{A}^T\mathbf{A}$ are the right-singular vectors $\mathbf{V}$ of $\mathbf{A}$
- eigenvalues of $\mathbf{A}^T\mathbf{A}$ are the squared singular values of $\boldsymbol{\Sigma}$

SVD – Mathematical Meaning

- $\mathbf{AA}^T \in \mathbb{R}^{m\times m}$ is symmetric, positive semi-definite matrix
- $\mathbf{AA}^T = \mathbf{SDS}^{\mathrm{T}} = \mathbf{S}\begin{bmatrix} \lambda_1 & \dots & 0 \\ \dots & \dots & \dots \\ 0 & \dots & \lambda_n \end{bmatrix}\mathbf{S}^T$
- $\mathbf{AA}^T = \mathbf{U\Sigma V}^{\mathrm{T}}\mathbf{V\Sigma}^{\mathrm{T}}\mathbf{U}^{\mathrm{T}} = \mathbf{U\Sigma\Sigma}^{\mathrm{T}}\mathbf{U}^{\mathrm{T}} = \mathbf{U}\begin{bmatrix} \sigma_m^2 & \dots & 0 \\ \dots & \dots & \dots \\ 0 & \dots & \sigma_m^2 \end{bmatrix}\mathbf{U}^{\mathrm{T}}$
- $\mathbf{S}^T = U^T,\ \sigma_i^2 = \lambda_i$
- Thus, eigenvectors of $\mathbf{AA}^T$ are the left-singular vectors $\mathbf{U}$ of **A**
- Eigenvalues of $\mathbf{AA}^T$ are the squared singular values of $\mathbf{\Sigma}$

SVD – Mathematical Meaning

- $\mathbf{A} = \mathbf{U\Sigma V}^{\mathrm{T}}$

- Eigenvectors of $\mathbf{AA}^T$ are the left-singular vectors $\mathbf{U}$ of $\mathbf{A}$
- Eigenvectors of are the right-singular vectors $\mathbf{V}$ of $\mathbf{A}$
- Eigenvalues of $\mathbf{AA}^T$ and $\mathbf{A}^T\mathbf{A}$ are the squared singular values of $\mathbf{\Sigma}$

SVD – Matrix Approximation

- $\mathbf{A} = \mathbf{U\Sigma V}^{\mathrm{T}} = \sum_{i=1}^{r} \sigma_i \mathbf{u}_i \mathbf{v}_i^T \equiv \sum_{i=1}^{r} \sigma_i \mathbf{A}_i$

- Remark that $\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_r \geq 0$

- Thus, we can easily find the rank-k approximation by:
  - $\widehat{\mathbf{A}}(k) = \sum_{i=1}^{k} \sigma_i \mathbf{u}_i \mathbf{v}_i^T = \sum_{i=1}^{k} \sigma_i \mathbf{A}_i$

- This is called “Low-rank Approximation”

SVD & Pseudo-inverse Matrix

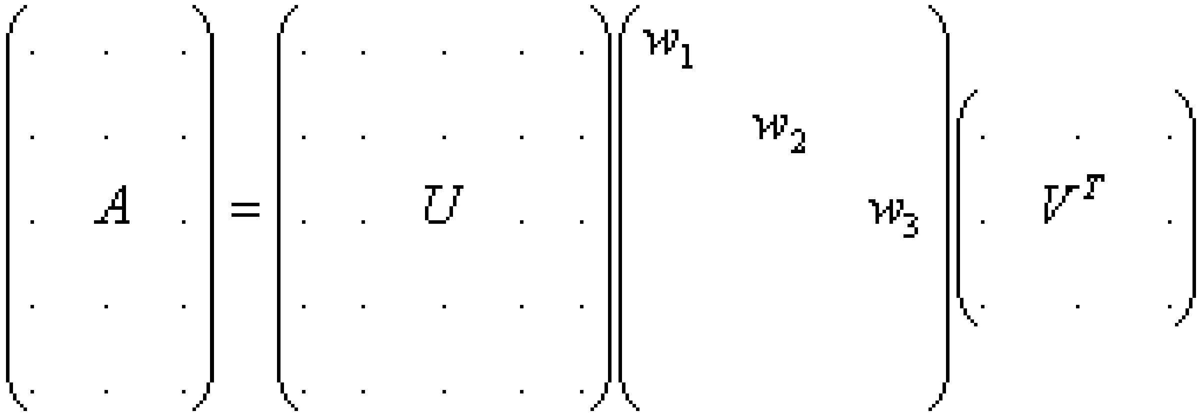
- $A = U\Sigma V^T$
- Basically, they satisfies
  - U : Orghogonal
    - $AA^T = U(\Sigma\Sigma^T)U^T$
    - $U^{-1} = U^T$
  - V : Orghogonal
    - $A^T A = V(\Sigma^{\mathrm{T}}\Sigma)V^{\mathrm{T}}$
    - $V^{-1} = V^T$

- $A^+ = (A^T A)^{-1}A^T = \left(V(\Sigma^{\mathrm{T}}\Sigma)V^{\mathrm{T}}\right)^{-1} V\Sigma^T U^T = V(\Sigma^{-1}\Sigma^{-T})V^T V\Sigma^{\mathrm{T}}\mathrm{U}^{\mathrm{T}}$
- Thus, $A^+ = V\Sigma^{-1}U^T$

Fully Connected Layers: Singular Value Decomposition

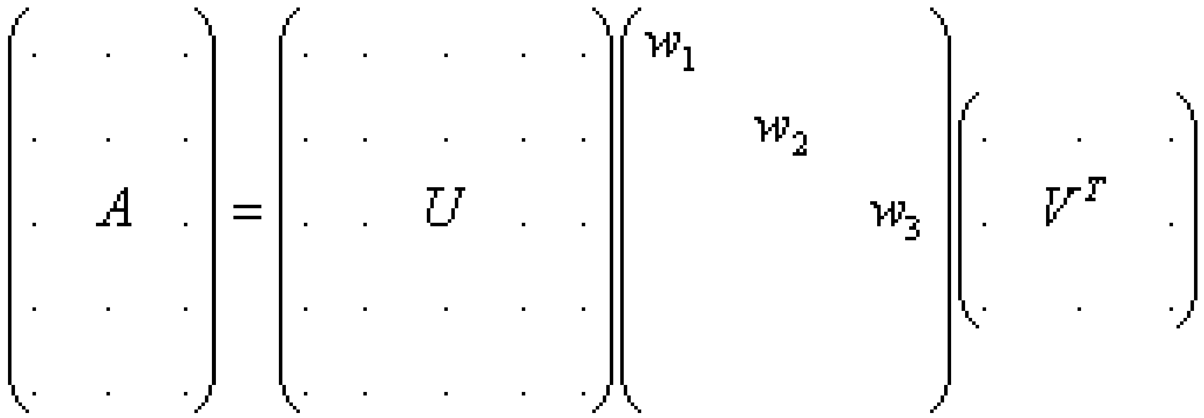
- Most weights are in the fully connected layers (according to Denton et al.)
- $W = USV^\top$
  - $W \in \mathbb{R}^{m\times k}, U \in \mathbb{R}^{m\times m}, S \in \mathbb{R}^{m\times k}, V^\top \in \mathbb{R}^{k\times k}$
- S is diagonal, decreasing magnitudes along the diagonal

http://www.alglib.net/matrixops/general/i/svd1.gif



Singular Value Decomposition

- By only keeping the t singular values with largest magnitude:
- $\widetilde{W} = \widetilde{U}\tilde{S}\widetilde{V}^{\top}$
  - $\widetilde{W} \in \mathbb{R}^{m\times k}, \widetilde{U} \in \mathbb{R}^{m\times t}, \tilde{S} \in \mathbb{R}^{t\times t}, \widetilde{V}^{\top} \in \mathbb{R}^{t\times k}$
- $Rank(\widetilde{W}) = t$



http://www.alglib.net/matrixops/general/i/svd1.gif

SVD: Compression

- $W = USV^\top, W \in \mathbb{R}^{m\times k}, U \in \mathbb{R}^{m\times m}, S \in \mathbb{R}^{m\times k}, V^\top \in \mathbb{R}^{k\times k}$
- $\widetilde{W} = \widetilde{U}\tilde{S}\widetilde{V}^\top, \widetilde{W} \in R^{m\times k}, \widetilde{U} \in R^{m\times t}, \tilde{S} \in R^{t\times t}, \tilde{V}^\top \in R^{t\times k}$

- Storage for W: $O(mk)$
- Storage for $\widetilde{W}$: $O(mt + t + tk)$
- Compression Rate: $O\left(\frac{mk}{t(m+k+1)}\right)$
- Theoretical error: $\left\|A\widetilde{W} - AW\right\|_F \leq s_{t+1}\|A\|_F$

Gong, Yunchao, et al. "Compressing deep convolutional networks using vector quantization." *arXiv preprint arXiv:1412.6115* (2014).

SVD: Compression Results

- Trained on ImageNet 2012 database, then compressed
- 5 convolutional layers, 3 fully connected layers, softmax output layer

Approximation method	Number of parameters	Approximation hyperparameters	Reduction in weights	Increase in error
Standard FC	NM			
FC layer 1: Matrix SVD	$NK + KM$	$K = 250$ $K = 950$	$13.4\times$ $3.5\times$	0.8394% 0.09%
FC layer 2: Matrix SVD	$NK + KM$	$K = 350$ $K = 650$	$5.8\times$ $3.14\times$	0.19% 0.06%
FC layer 3: Matrix SVD	$NK + KM$	$K = 250$ $K = 850$	$8.1\times$ $2.4\times$	0.67% 0.02%

K refers to rank of approximation, t in the previous slides.

Denton, Emily L., et al. "Exploiting linear structure within convolutional networks for efficient evaluation." *Advances in Neural Information Processing Systems*. 2014.

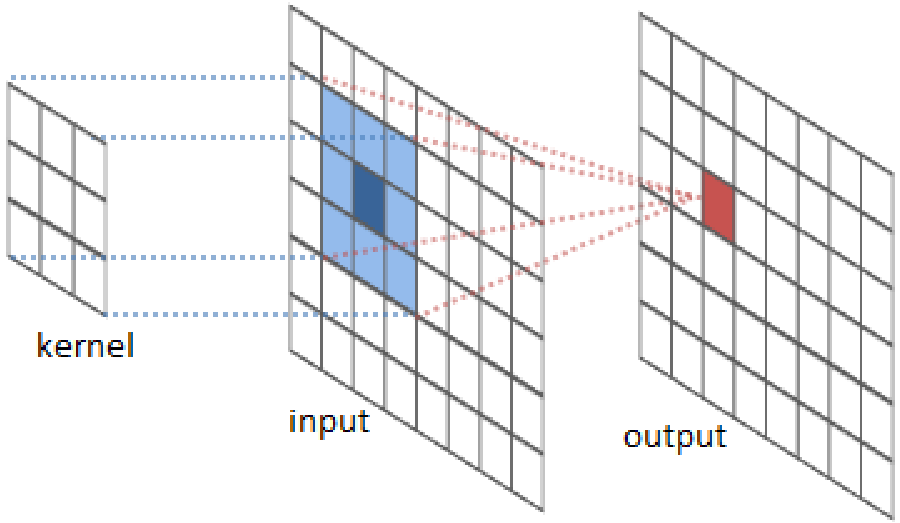
SVD: Side Benefits

- Reduced memory footprint
  - Reduced in the dense layers by 5-13x
- Speedup: $A\widetilde{W}, A \in \mathbb{R}^{n \times m}$, computed in $O(nmt + nt^2 + ntk)$ instead of $O(nmk)$
  - Speedup factor is $O\left(\frac{mk}{t(m+t+k)}\right)$
- Regularization
  - "Low-rank projections effectively decrease number of learnable parameters, s uggesting that they might improve generalization ability."
  - Paper applies SVD after training

Denton, Emily L., et al. "Exploiting linear structure within convolutional networks for efficient evaluation." *Advances in Neural Information Processing Systems*. 2014.

Convolutions: Matrix Multiplication

Most time is spent in the convolutional layers

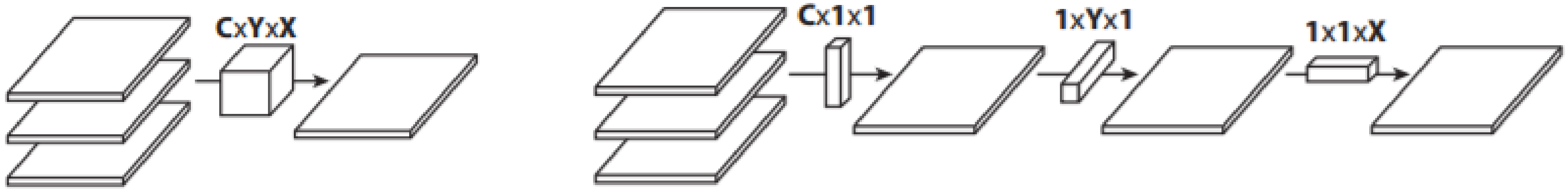


$$F(x, y) = I * W$$

http://stackoverflow.com/questions/15356153/how-do-convolution-matrices-work

Flattened Convolutions

- Replace $c \times y \times x$ convolutions with $c \times 1 \times 1$, $1 \times y \times 1$, and $1 \times 1 \times x$ convolutions



(a) 3D convolution (b) 1D convolutions over different directions

Jin, Jonghoon, Aysegul Dundar, and Eugenio Culurciello. "Flattened convolutional neural networks for feedforward acceleration." *arXiv preprint arXiv:1412.5474* (2014).

Flattened Convolutions

$$\hat{F}(x,y) = I * \widehat{W}$$
$$= \sum_{x'=1}^{X}\left(\sum_{y'=1}^{Y}\left(\sum_{c=1}^{C} I(c, x-x', y-y')\alpha(c)\right)\beta(y')\right)\gamma(x')$$
$$\alpha \in \mathbb{R}^C, \beta \in \mathbb{R}^Y, \gamma \in \mathbb{R}^X$$

- Compression and Speedup:
  - Parameter reduction: $\mathrm{O}(XYC)$ to $\mathrm{O}(X+Y+C)$
  - Operation reduction: $O(mnCXY)$ to $O\big(mn(C+X+Y)\big)$ (where $\mathrm{W_f} \in \mathbb{R}^{m\times n}$)

Jin, Jonghoon, Aysegul Dundar, and Eugenio Culurciello. "Flattened convolutional neural networks for feedforward acceleration." *arXiv prep rint arXiv:1412.5474* (2014).

Flattening = MF

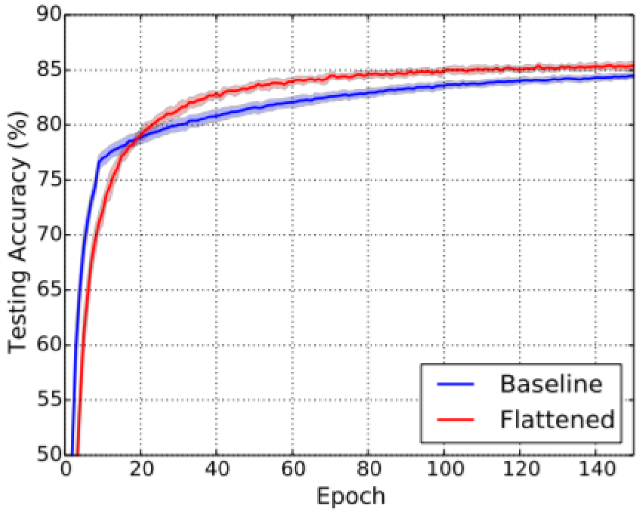
$$\hat{F}(x,y) = \sum_{x=1}^{X} \sum_{y'=1}^{Y} \sum_{c=1}^{C} I(c, x-x', y-y')\alpha(c)\,\beta(y')\,\gamma(x')$$
$$= \sum_{x=1}^{X} \sum_{y'=1}^{Y} \sum_{c=1}^{C} I(c, x-x', y-y')\,\hat{W}(c, x', y')$$

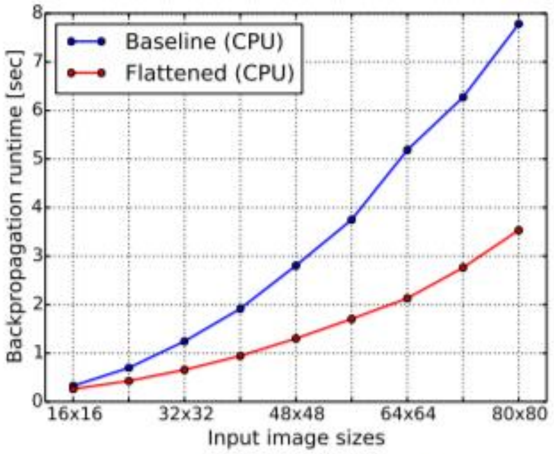
- $\hat{W} = \alpha \otimes \beta \otimes \gamma, Rank\left(\hat{W}\right) = 1$
- $\hat{W}_S = \sum_{k=1}^{K} \alpha_k \otimes \beta_k \otimes \gamma_k$, Rank K
- SVD: Can reconstruct the original matrix as $A = \sum_{k=1}^{K} w_k u_k \otimes v_k$

Denton, Emily L., et al. "Exploiting linear structure within convolutional networks for efficient evaluation." *Advances in Neural Information Processing Systems*. 2014.

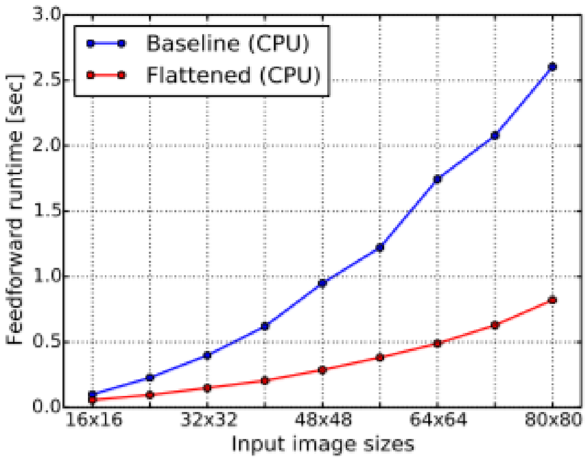
Flattening: Speedup Results

- 3 convolutional layers (5x5 filters) with 96, 128, and 256 channels
- Used stacks of 2 rank-1 convolutions





(c) Backpropagation on CPU



(a) Feedforward on CPU

Jin, Jonghoon, Aysegul Dundar, and Eugenio Culurciello. "Flattened convolutional neural networks for feedforward acceleration." *arXiv preprint arXiv:1412.5474* (2014).

Today’s Lecture

- Matrix Factorization (SVD)
- Network Pruning
- Network Quantization
- Network Pruning + Quantization