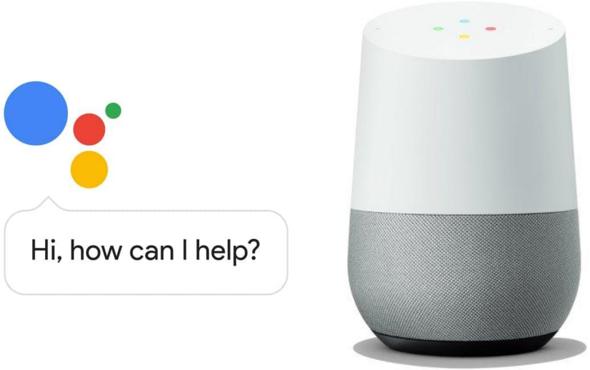
The Singular Values of Convolutional Layers

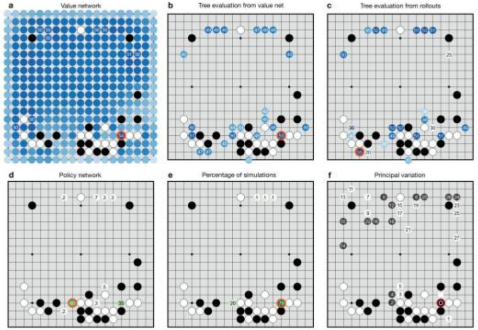
Hanie Sedghi
Google Brain

Joint work with Vineet Gupta and Phil Long

Neural Networks

Tremendous practical impact with deep learning





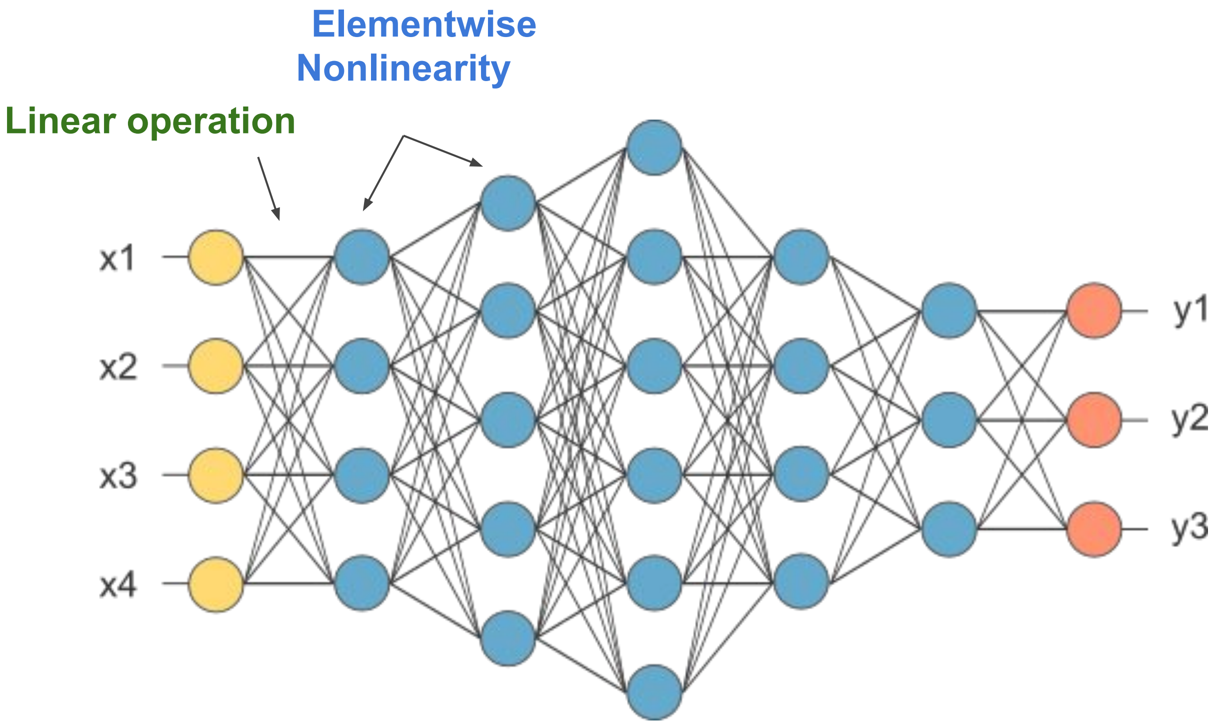
AlphaGo

ImageNet Dataset

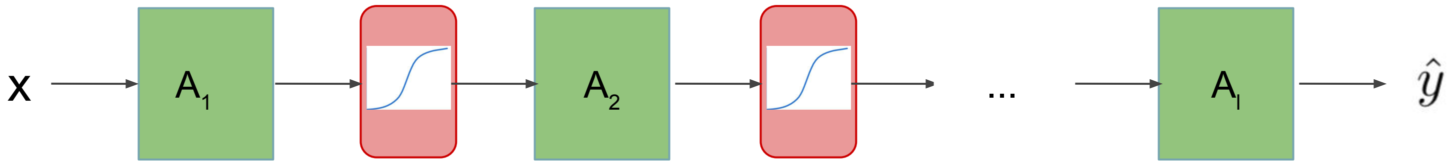
Russakovsky, Olga, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang et al. "Imagenet large scale visual recognition challenge." International Journal of Computer Vision 115, no. 3 (2015): 211-252. [web]

3

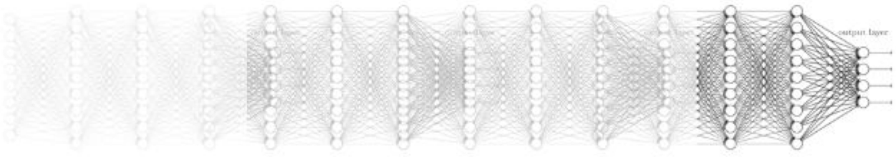
Deep Network Architecture



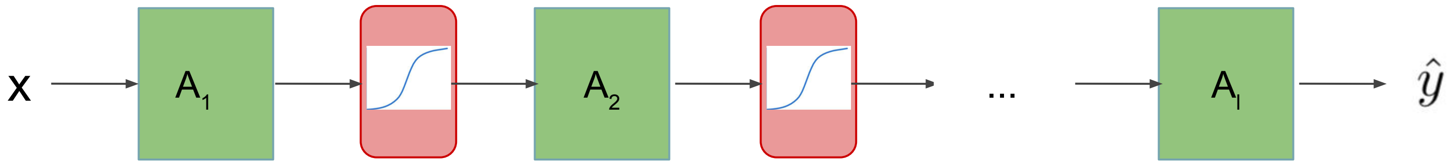
Exploding and vanishing gradients



- Gradients backpropagate
- Danger of explosion (NaN) and vanishing (very small changes)...
- … also in the forward direction

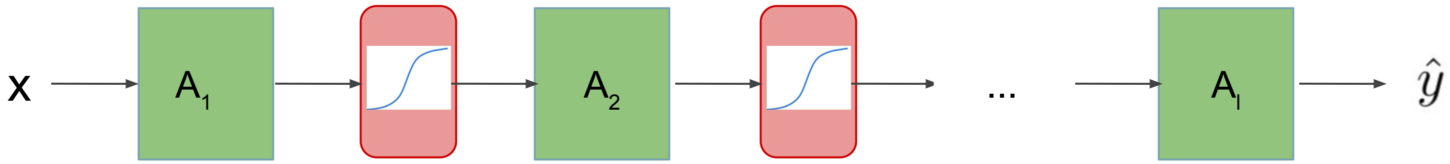


Key: singular values of linear layers



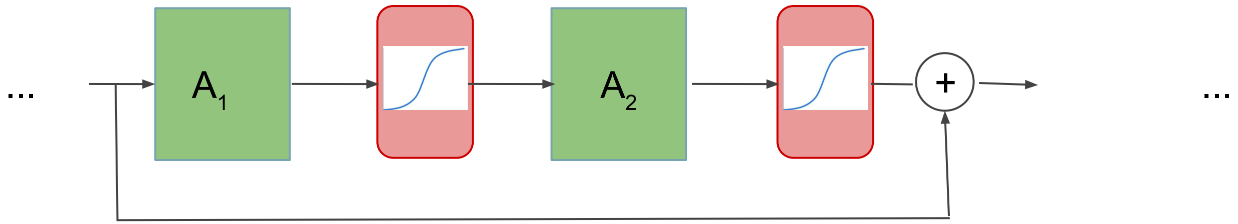
- Main threats are linear layers
- Singular values bound factor by which layer increases or decreases length of its input

Operator norm



- Network Lipschitz constant ≤ product of operator norms of linear layers
- Motivates **regularization via control of operator norms.**

Residual networks

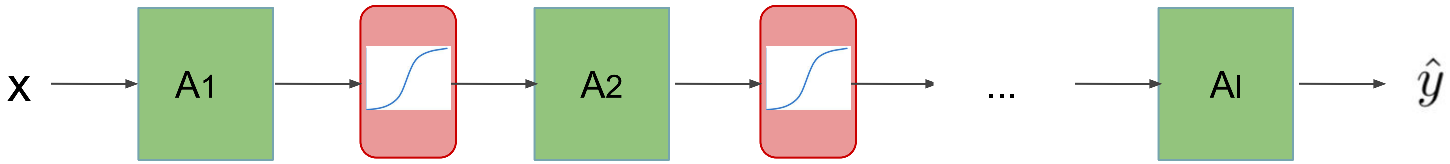


Operator norms of A_1 and A_2 are small

$$\Rightarrow$$

Singular values of block near 1

Control the operator norm



- **Regularization** [Drucker and Le Cun, 1992; Hein and Andriushchenko, 2017; Yoshida and Miyato, 2017; Miyato et al., 2018]
- **Generalization** [Bartlett et al. 2017]
- **Robustness to adversarial examples** [Cisse et al. 2017]

Operator norm for Convolution

- **Regularize** networks by reducing the operator norm of the linear transformation.
- Authors have identified operator norm as **important**, but they did not succeed in finding operator norm for convolution.
- Resorted to **approximations** (Yoshida and Miyato, 2017; Miyato et al., 2018; Gouk et al., 2018a).

Our Contribution:

- Characterize singular values of convolutional layers
- Simple, fast algorithm
- Regularizer (via projection)

Convolution Layer

Discrete-value Convolution

Linear combination of pixels

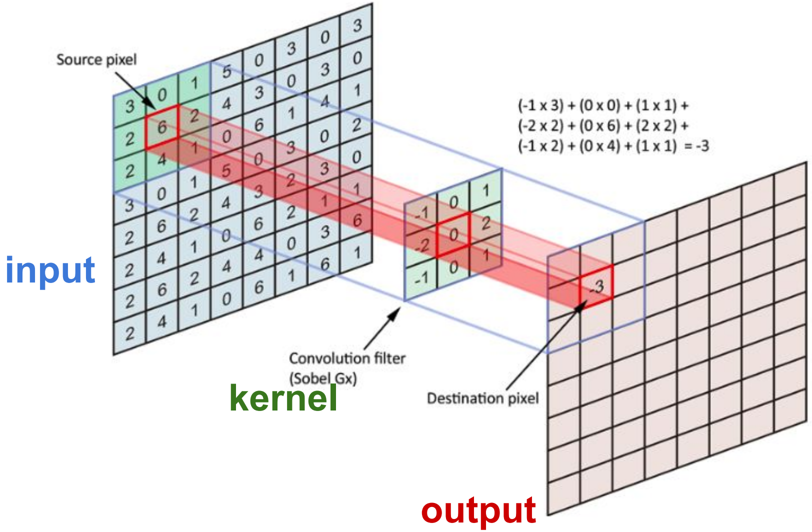
Applied locally

$$Y_{ij} = \sum_{p \in [n]} \sum_{q \in [n]} X_{i+p,j+q} K_{p,q}$$

output

input

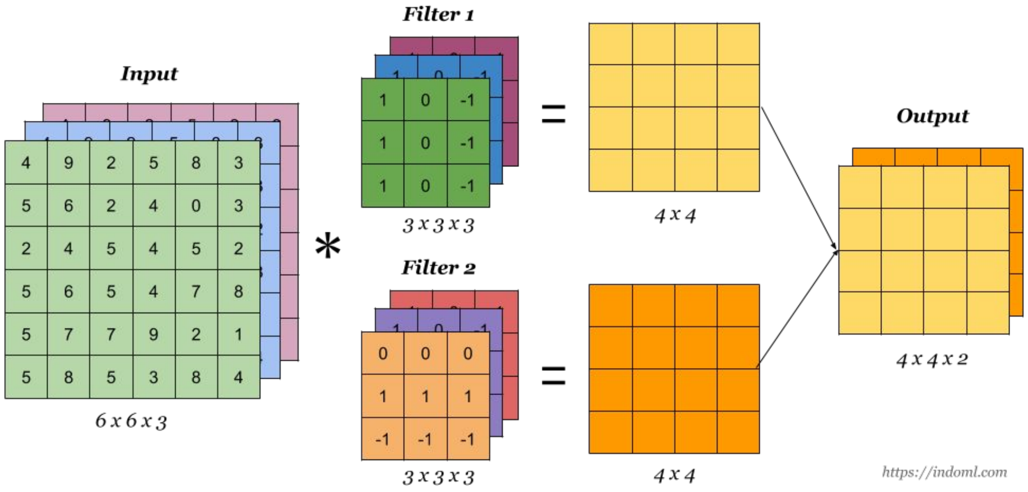
kernel



(Reproduced from medium.freecodecamp.org.)

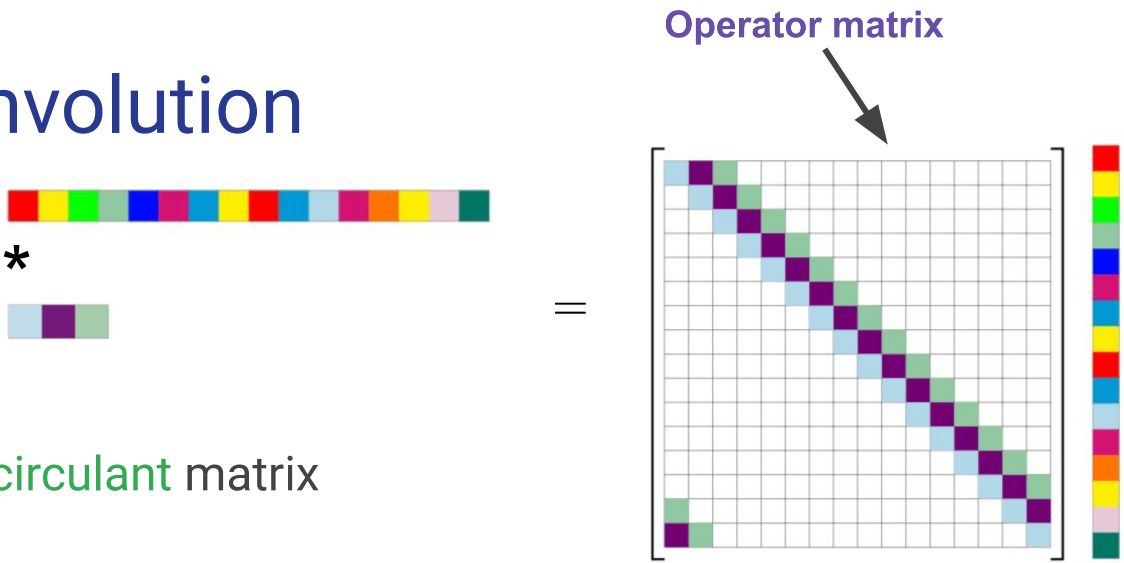
Multi-channel convolutional layer

$$Y_{crs} = \sum_{d \in [m]} \sum_{p \in [n]} \sum_{q \in [n]} X_{d,r+p,s+q} K_{p,q,c,d}.$$



1D Circular Convolution

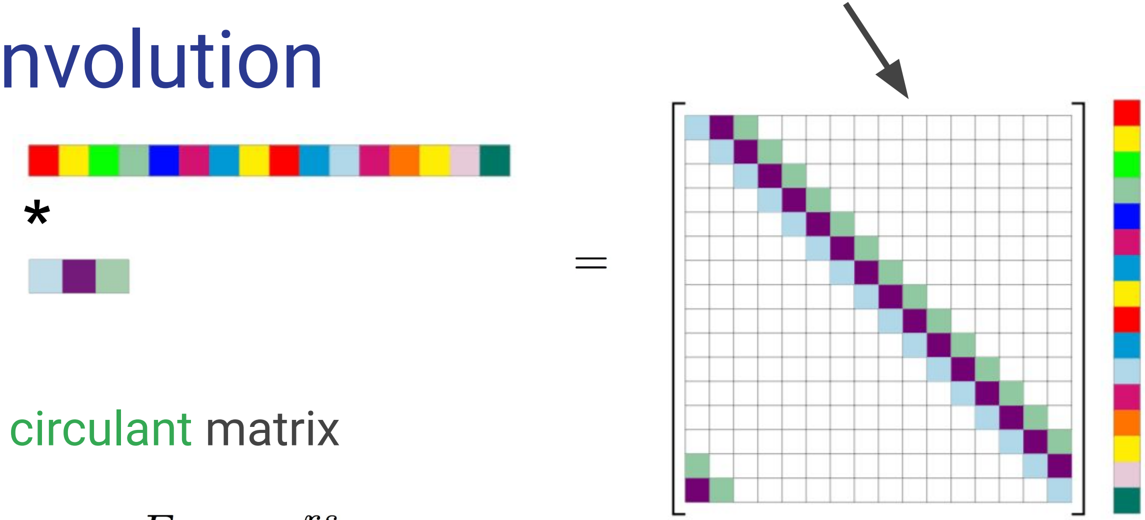
$$\forall i \; Y_i = \sum_{p \in [n]} X_{i+p} K_p$$



The operator matrix is a circulant matrix

1D Circular Convolution

$$\forall i \; Y_i = \sum_{p \in [n]} X_{i+p} K_p$$



The operator matrix is a circulant matrix

Discrete Fourier Transform

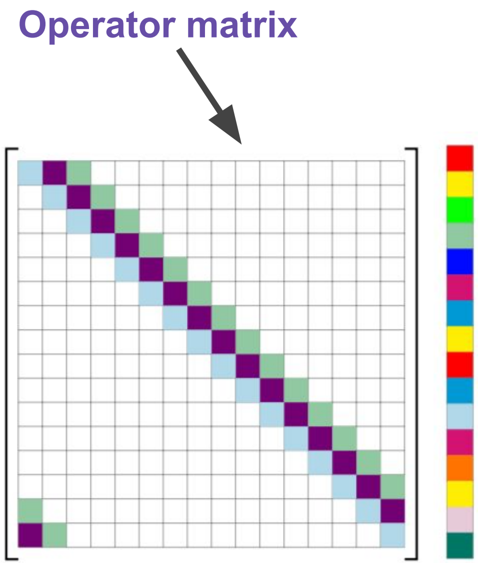
$$F_{rs} = \omega^{rs}$$

$$\omega = \exp(2\pi i / n)$$

Column of F are eigenvectors.

1D Circular Convolution

$$\forall i \; Y_i = \sum_{p \in [n]} X_{i+p} K_p$$



The operator matrix is a circulant matrix

Discrete Fourier Transform $\quad F_{rs} = \omega^{rs}$

$$\omega = \exp(2\pi i / n)$$

Column of F are eigenvectors.

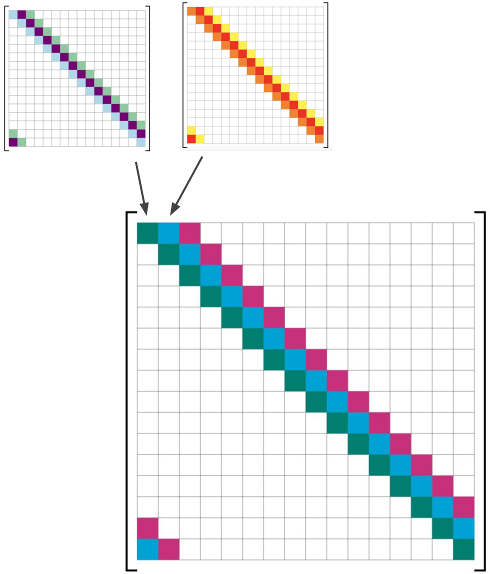
Singular values $\quad \{|(FK)_u|, u \in n\}$

2D Single-channel Convolution

Operator matrix is a doubly-block circulant matrix

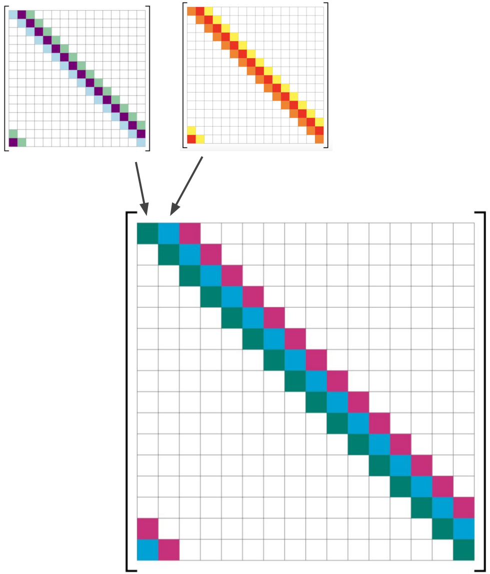
$$\forall ij,\ Y_{ij} = \sum_{p\in[n]} \sum_{q\in[n]} X_{i+p,j+q} K_{p,q}$$

$$A = \begin{bmatrix} \mathrm{circ}(K_{0,:}) & \mathrm{circ}(K_{1,:}) & \dots & \mathrm{circ}(K_{n-1,:}) \\ \mathrm{circ}(K_{n-1,:}) & \mathrm{circ}(K_{0,:}) & \dots & \mathrm{circ}(K_{n-2,:}) \\ \vdots & \vdots & \vdots & \vdots \\ \mathrm{circ}(K_{1,:}) & \mathrm{circ}(K_{2,:}) & \dots & \mathrm{circ}(K_{0,:}) \end{bmatrix}$$



2D Single-channel Convolution

Operator matrix is a doubly-block circulant matrix



$$\forall ij,\ Y_{ij} = \sum_{p\in[n]} \sum_{q\in[n]} X_{i+p,j+q} K_{p,q}$$

$$A = \begin{bmatrix} \mathrm{circ}(K_{0,:}) & \mathrm{circ}(K_{1,:}) & \dots & \mathrm{circ}(K_{n-1,:}) \\ \mathrm{circ}(K_{n-1,:}) & \mathrm{circ}(K_{0,:}) & \dots & \mathrm{circ}(K_{n-2,:}) \\ \vdots & \vdots & \vdots & \vdots \\ \mathrm{circ}(K_{1,:}) & \mathrm{circ}(K_{2,:}) & \dots & \mathrm{circ}(K_{0,:}) \end{bmatrix}$$

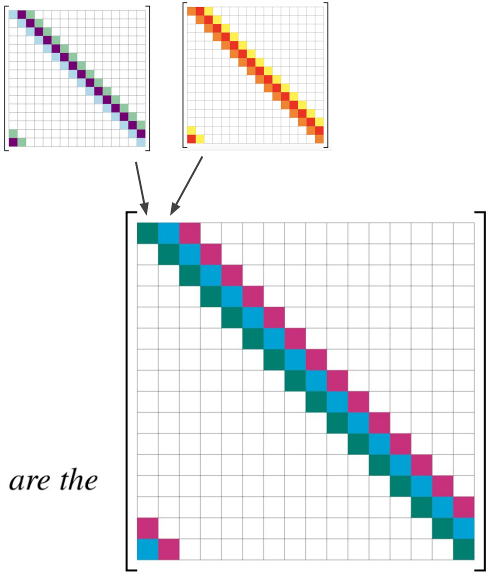
Theorem ([Jain, 1989]) *For any $n^2 \times n^2$ doubly block circulant matrix A, the eigenvectors of A are the columns of $\frac{1}{n}(F \otimes F)$*

2D Single-channel Convolution

Operator matrix is a doubly-block circulant matrix

$$\forall ij,\ Y_{ij} = \sum_{p\in[n]} \sum_{q\in[n]} X_{i+p,j+q} K_{p,q}$$

$$A = \begin{bmatrix} \mathrm{circ}(K_{0,:}) & \mathrm{circ}(K_{1,:}) & \dots & \mathrm{circ}(K_{n-1,:}) \\ \mathrm{circ}(K_{n-1,:}) & \mathrm{circ}(K_{0,:}) & \dots & \mathrm{circ}(K_{n-2,:}) \\ \vdots & \vdots & \vdots & \vdots \\ \mathrm{circ}(K_{1,:}) & \mathrm{circ}(K_{2,:}) & \dots & \mathrm{circ}(K_{0,:}) \end{bmatrix}$$



Theorem ([Jain, 1989]) *For any $n^2 \times n^2$ doubly block circulant matrix A, the eigenvectors of A are the columns of $\frac{1}{n}(F \otimes F)$*

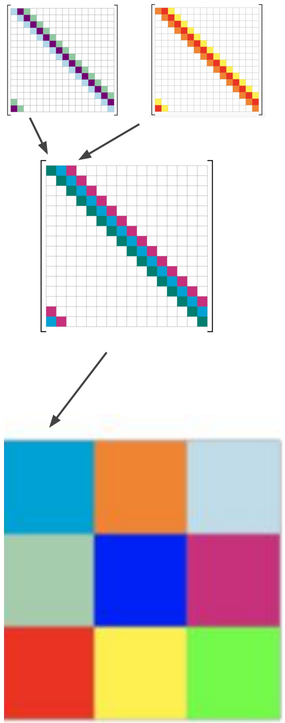
Theorem *For the matrix A defined in (1), the eigenvalues of A are the entries of $F^T K F$, and its singular values are their magnitudes. That is, the singular values of A are*

$$\left\{ \left| (F^T K F)_{u,v} \right| \ : \ u, v \in [n] \right\}.$$

2D Multi-channel Convolution

$$Y_{crs} = \sum_{d\in[m]} \sum_{p\in[n]} \sum_{q\in[n]} X_{d,r+p,s+q} K_{p,q,c,d}.$$

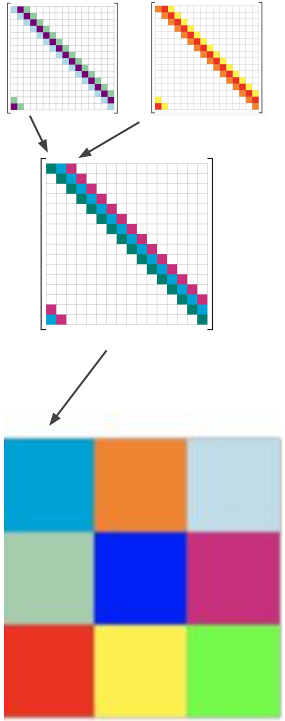
$$M = \begin{bmatrix} B_{00} & B_{01} & \cdots & B_{0(m-1)} \\ B_{10} & B_{11} & \cdots & B_{1(m-1)} \\ \vdots & \vdots & \cdots & \vdots \\ B_{(m-1)0} & B_{(m-1)1} & \cdots & B_{(m-1)(m-1)} \end{bmatrix}$$



2D Multi-channel Convolution

$$Y_{crs} = \sum_{d\in[m]} \sum_{p\in[n]} \sum_{q\in[n]} X_{d,r+p,s+q} K_{p,q,c,d}.$$

$$M = \begin{bmatrix} B_{00} & B_{01} & \cdots & B_{0(m-1)} \\ B_{10} & B_{11} & \cdots & B_{1(m-1)} \\ \vdots & \vdots & \cdots & \vdots \\ B_{(m-1)0} & B_{(m-1)1} & \cdots & B_{(m-1)(m-1)} \end{bmatrix} \left[\begin{array}{c} X_{0:0} \\ \vdots \\ X_{0:n^2} \\ \hline X_{1:0} \\ \vdots \\ X_{1:n^2} \\ \hline \vdots \\ \hline X_{(m-1):0} \\ \vdots \\ X_{(m-1):n^2} \end{array}\right]$$

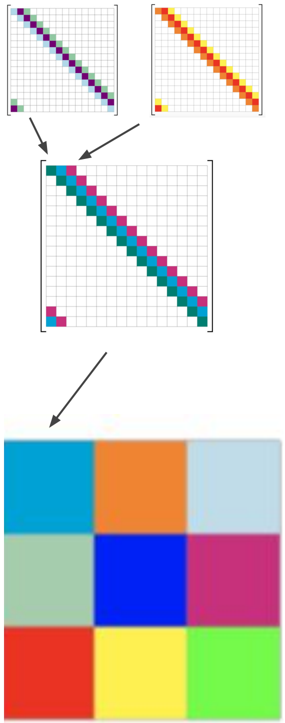


2D Multi-channel Convolution

$$Y_{crs} = \sum_{d \in [m]} \sum_{p \in [n]} \sum_{q \in [n]} X_{d,r+p,s+q} K_{c,d,p,q}.$$

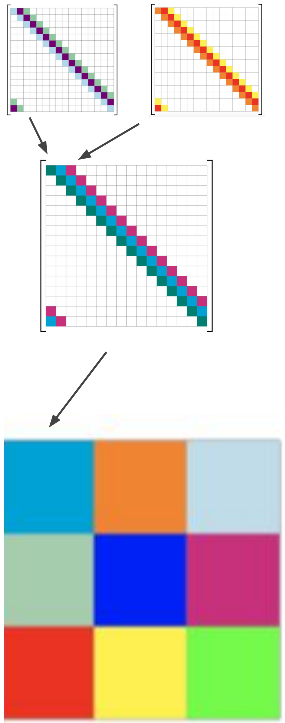
$$M = \begin{bmatrix} B_{00} & B_{01} & \cdots & B_{0(m-1)} \\ B_{10} & B_{11} & \cdots & B_{1(m-1)} \\ \vdots & \vdots & \cdots & \vdots \\ B_{(m-1)0} & B_{(m-1)1} & \cdots & B_{(m-1)(m-1)} \end{bmatrix}$$

$$\boxed{\sigma(M) = \bigcup_{u \in [n], v \in [n]} \sigma\left(\left((F^T K_{:,:,c,d} F)_{u,v} \right)_{cd} \right)}$$



Proof Sketch

All blocks in M have the same eigenvectors, so...



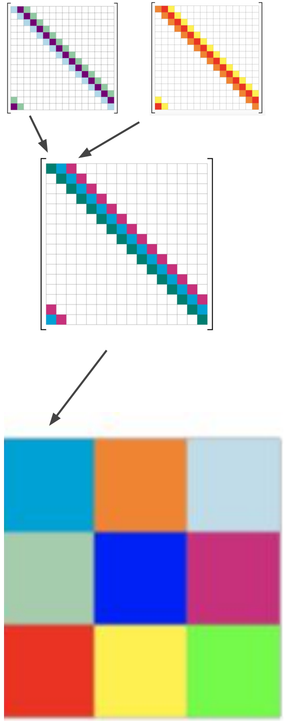
$$M = \begin{bmatrix} B_{00} & B_{01} & \cdots & B_{0(m-1)} \\ B_{10} & B_{11} & \cdots & B_{1(m-1)} \\ \vdots & \vdots & \cdots & \vdots \\ B_{(m-1)0} & B_{(m-1)1} & \cdots & B_{(m-1)(m-1)} \end{bmatrix}$$

$$= \quad I_m \otimes \frac{1}{n}(F \otimes F) \begin{bmatrix} D_{00} & D_{01} & \cdots & D_{0(m-1)} \\ D_{10} & D_{11} & \cdots & D_{1(m-1)} \\ \vdots & \vdots & \cdots & \vdots \\ D_{(m-1)0} & D_{(m-1)1} & \cdots & D_{(m-1)(m-1)} \end{bmatrix} [I_m \otimes \frac{1}{n}(F \otimes F)]^*$$

Proof Sketch

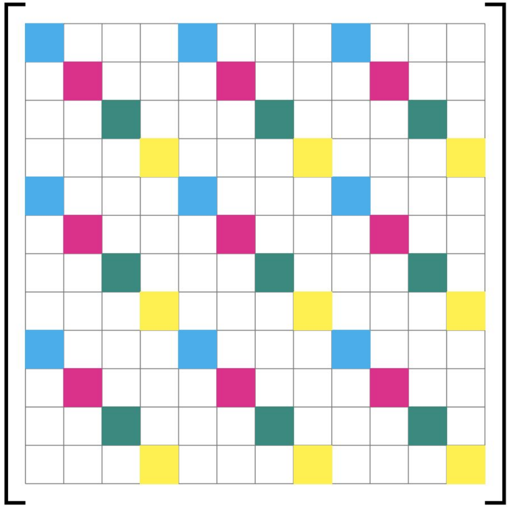
We need the singular values of

$$\begin{bmatrix} D_{00} & D_{01} & \cdots & D_{0(m-1)} \\ D_{10} & D_{11} & \cdots & D_{1(m-1)} \\ \vdots & \vdots & \cdots & \vdots \\ D_{(m-1)0} & D_{(m-1)1} & \cdots & D_{(m-1)(m-1)} \end{bmatrix}$$

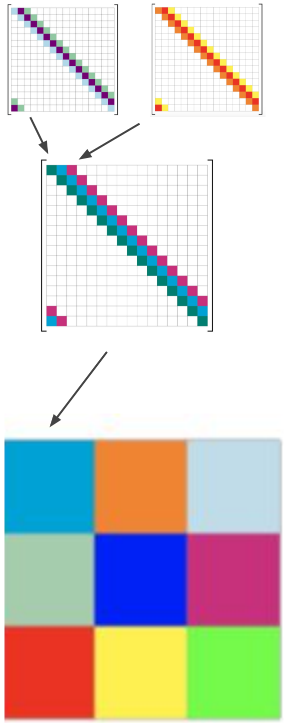


Proof Sketch

We need the singular values of

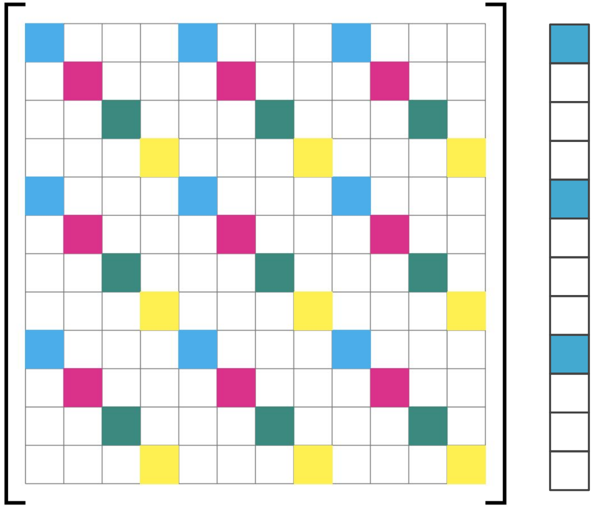


$$m = 3, n = 2$$

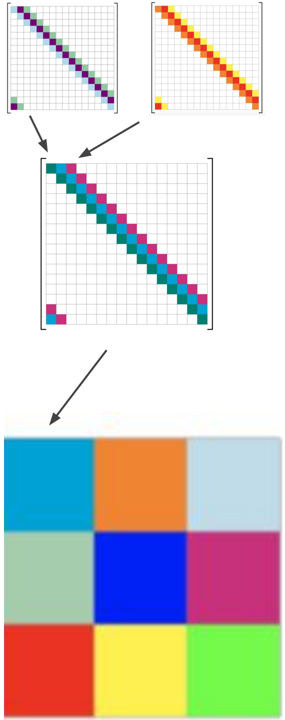


Proof Sketch

If (v_1, v_2, v_3) is a right singular vector of blue matrix, then $(v_1,0,0,0,v_2,0,0,0,v_3,0,0,0)$ is a right singular vector of the whole.



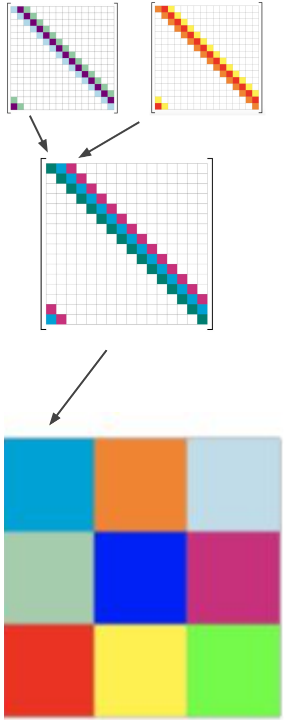
$$m = 3, n = 2$$



2D Multi-channel Convolution

$$Y_{crs} = \sum_{d\in[m]} \sum_{p\in[n]} \sum_{q\in[n]} X_{d,r+p,s+q} K_{p,q,c,d}.$$

$$M = \begin{bmatrix} B_{00} & B_{01} & \cdots & B_{0(m-1)} \\ B_{10} & B_{11} & \cdots & B_{1(m-1)} \\ \vdots & \vdots & \cdots & \vdots \\ B_{(m-1)0} & B_{(m-1)1} & \cdots & B_{(m-1)(m-1)} \end{bmatrix}$$



```
def SingularValues(kernel, input_shape):
  transform_coefficients = np.fft.fft2(kernel, input_shape, axes=[0, 1])
  return np.linalg.svd(transform_coefficients, compute_uv=False)
```

Computational Complexity

```
def SingularValues(kernel, input_shape):
  transform_coefficients = np.fft.fft2(kernel, input_shape, axes=[0, 1])
  return np.linalg.svd(transform_coefficients, compute_uv=False)
```

$$O(n^2m^2(m + \log n))$$

vs

$$O((n^2m)^3) = O(n^6m^3)$$

Application: Regularization

- Regularize deep convolutional networks by bounding the operator norm of each layer
- Improves generalization [Bartlett et al. 2017, Neyshabur et al. 2017]
- Improves robustness to adversarial attacks [Cisse et al. 2017]

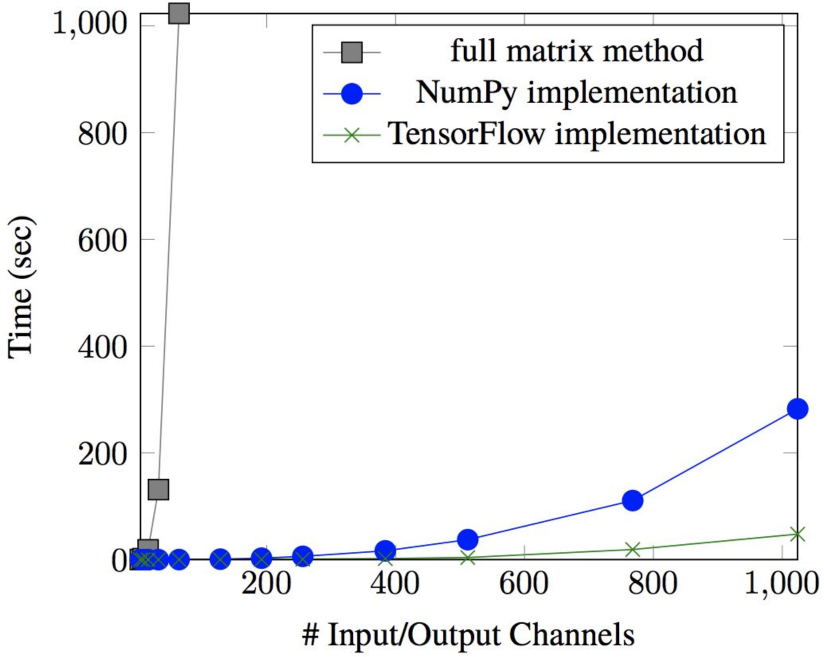
Theorem [Lefkimmiatis et al., 2013] (paraphrased): Clipping the singular values of a matrix A at c projects A into set of matrices whose operator norm is bounded by c.

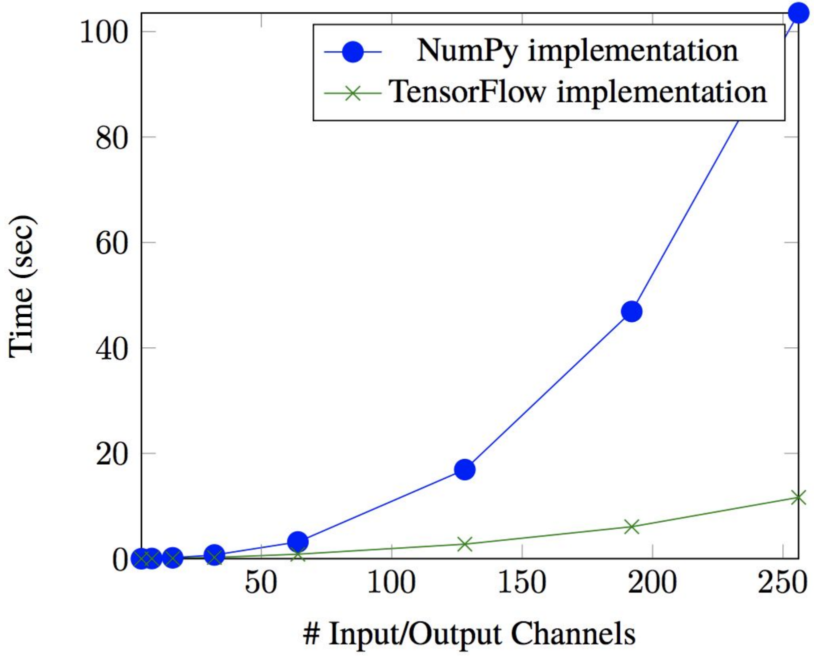
Bounding the Operator Norm

- Clip singular values
- Problem: larger neighborhoods
- Solution: alternating projections, i.e.
    - project into set with bounded operator norm
    - project into set of convolutions with $k \times k$ neighborhoods
- For projection onto intersection, can use Dykstra's algorithm
- In experiments, use simpler algorithm

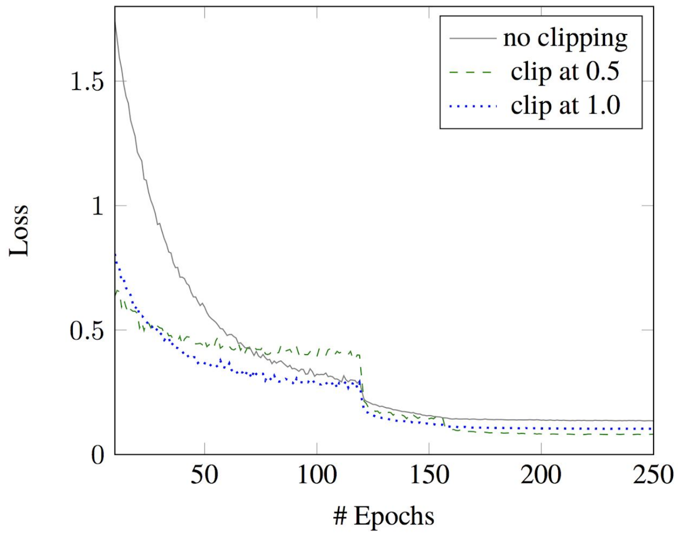
Experiments

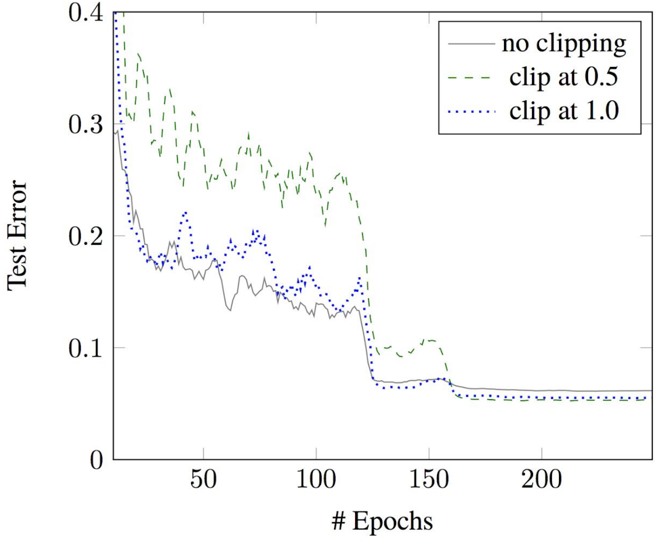
Experiments: Efficiency





Effect of Clipping on test error



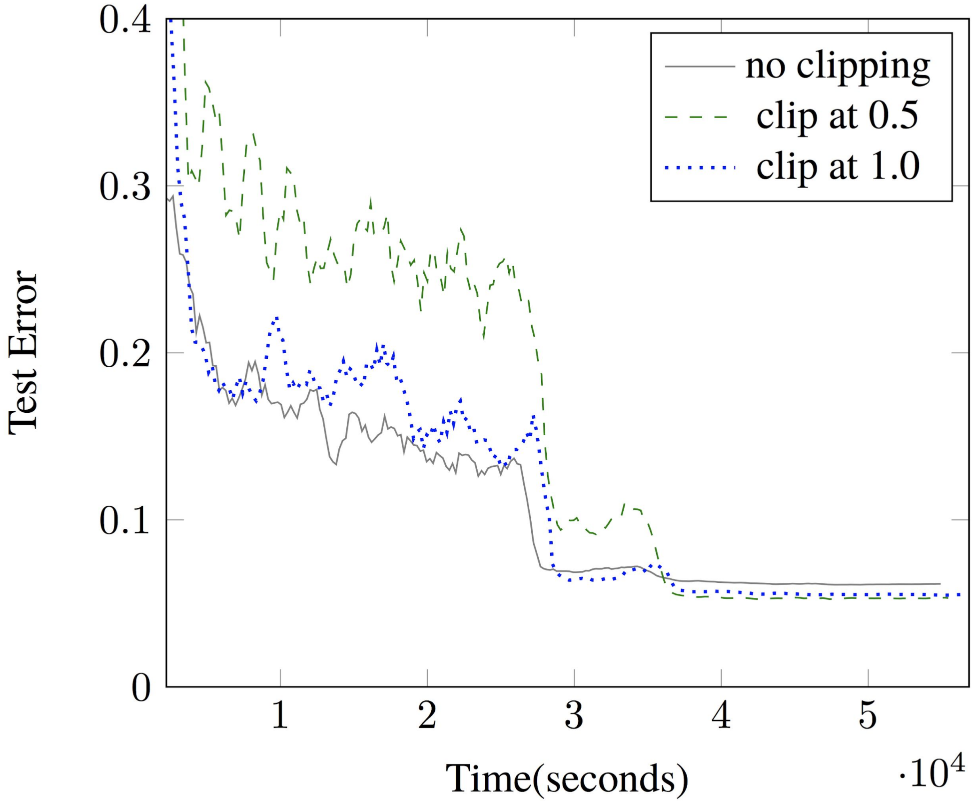


ResNet-32 on CIFAR10 dataset

Clipping to 0.5 and 1.0 yielded test errors of 5.3% and 5.5% respectively.

Baseline error rate 6.2%

Effect of Clipping on test error



ResNet-32

CIFAR10 dataset

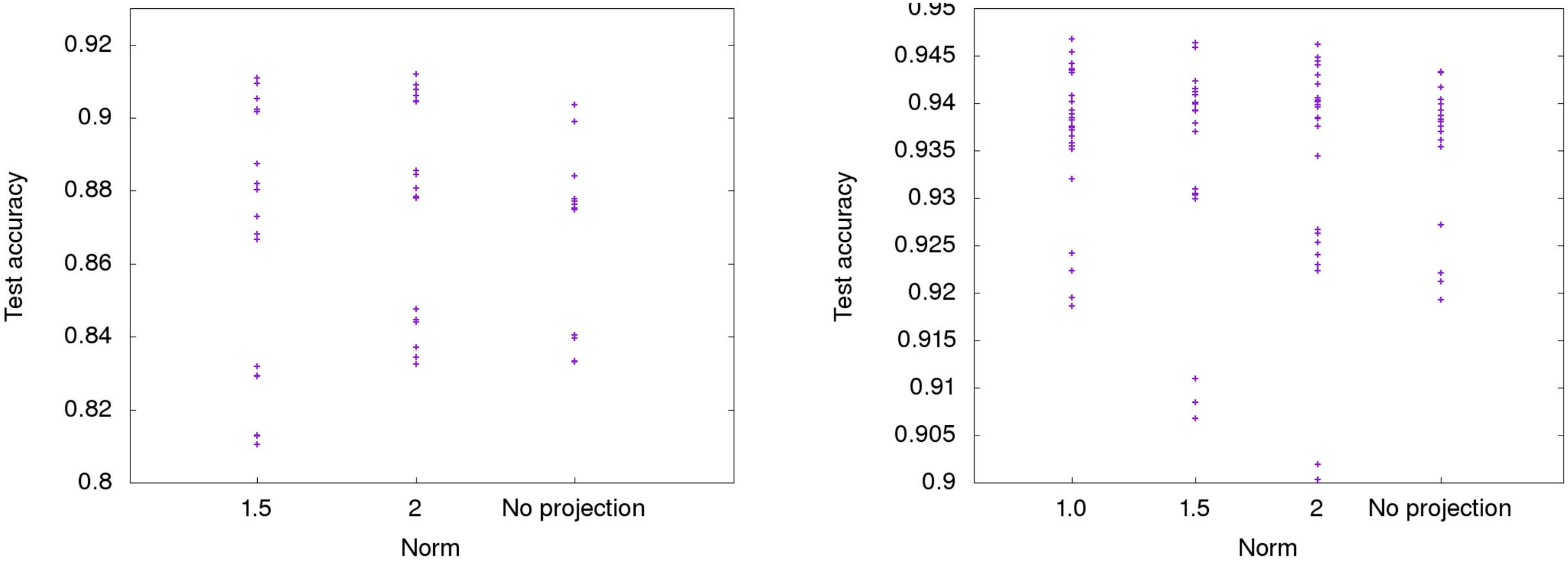
The projection does not slow down the training that much.

On batch normalization

- Earlier baseline uses batch normalization. which rescales weights
- Complicated interaction between batch norm and our method
- Repeated experiments without batchnorm

Robustness to hyperparameter changes

Operator-norm regularization and batch normalization are not redundant, and neither dominates the other.

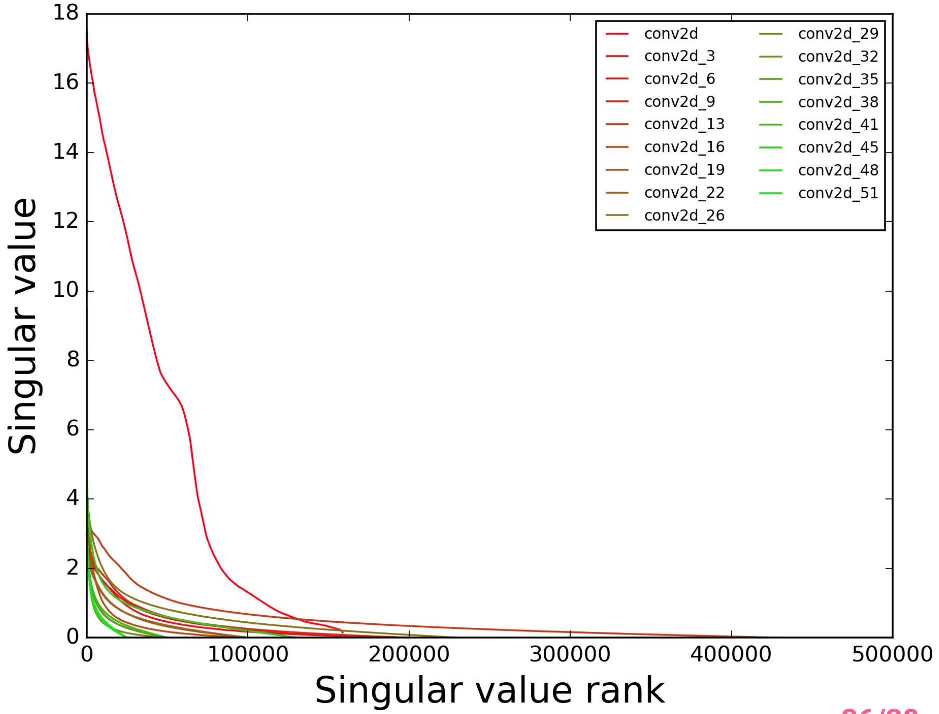


(a) Without batch normalization (b) With batch normalization

Singular values for ResNet V2

Layers closer to the input are plotted with colors with a greater share of red.

The transformations with the largest operator norms are closest to the input.



Conclusion

- Characterized singular values of a 2D multichannel convnet.
- Provided efficient & practical method for deriving them for deep networks.
- This opens the door to various regularizers.
- We showed an effective projection into set of bounded norm operators.

Future work

- Experiments on more datasets.
- Improve state of the art models such as Generative Adversarial Networks.

Paper: to appear in ICLR 2019

Code: https://github.com/brain-research/conv-sv

Thank You!