Matrix and tensor methods in ML
Lecture 1

Maxim Rakhuba

CS department
Higher School of Economics

June 21, 2022

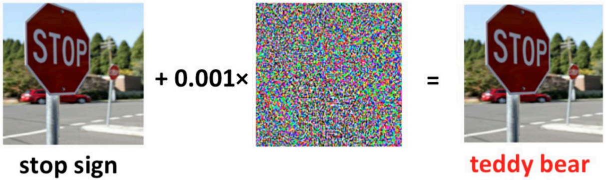
Lipschitz continuity

Lipschitz continuity and robustness

$$\|\mathcal{N}\mathcal{N}(x) - \mathcal{N}\mathcal{N}(y)\|_2 \leq L_{\mathcal{N}\mathcal{N}} \|x - y\|_2 \quad \forall x, y.$$

Bounding $L_{\mathcal{N}\mathcal{N}}$:

- improves generalisation;
- increases robustness against adversarial examples.



Lipschitz continuity

$$\mathcal{N}\mathcal{N} = \omega_L \circ f_{L-1} \circ \omega_{L-1} \circ \cdots \circ f_1 \circ \omega_1,$$

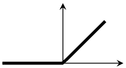
where

$$\omega_k(x) = W_k x + b_k, \qquad f_k \text{ — nonlinearity.}$$

Estimating $L_{\mathcal{N}\mathcal{N}}$

Since $L_{g_1 \circ g_2} \leq L_{g_1} L_{g_2}$:

$$L_{\mathcal{N}\mathcal{N}} \leq L_{f_1} \dots L_{f_L} L_{\omega_1} \dots L_{\omega_{L-1}}.$$

- For $f_k \equiv \mathrm{ReLU}$  :

$$\|f(x) - f(y)\|_2 \leq 1 \cdot \|x - y\|_2.$$

- For $\omega(x) = Wx + b$:

$$\| Wx + b - (Wy + b) \|_2 = \| W(x-y) \|_2 \leq \|W\|_2 \|x-y\|_2$$

$$\|W\|_2 = \sup_{x \neq 0} \frac{\|Wx\|_2}{\|x\|_2}$$

How to compute $\|W\|_2$?

$\|U\Sigma V^\top\|_2 = \|\Sigma\|_2 = \sigma_1(W)$

Let $W = U\Sigma V^\top$ – SVD of $W \in \mathbb{R}^{M\times N}$. Then

$$\|W\|_2 = \sigma_1(W) \equiv \sqrt{\lambda_1(W^\top W)}.$$

For $M = N$, computing SVD is $\mathcal{O}(N^3)$.

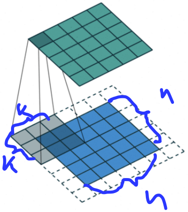
N	1000	5000	10000
Time	0.1 s	16 s	105 s
$\mathrm{Mem}(W)$	8 Mb	200 Mb	800 Mb

Works for small fully connected layers.

Convolutional layer

$$(C * X)_{q_1 q_2} \equiv \sum_{p_1=0}^{k-1} \sum_{p_2=0}^{k-1} \mathcal{C}_{p_1 p_2} \mathcal{X}_{p_1+q_1, p_2+q_2}$$

('$*$' – convolution operation)



Multichannel convolution

$$\mathcal{Y}_{:,:,j} = \sum_{i=1}^{m} \mathcal{C}_{:,:,i,j} * \mathcal{X}_{:,:,i},$$

where $\mathcal{C} \in \mathbb{R}^{k \times k \times m_{in} \times m_{out}}$ – convolution kernel:

- k – filter size;
- m_{in} – number of input channels (e.g., 3 for RGB);
- m_{out} – number of output channels.

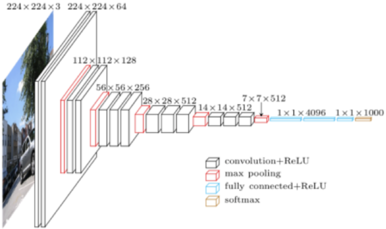
Convolutional layer

Multichannel convolution is a linear map, so

$$\mathrm{vec}(\mathcal{Y}) = W\,\mathrm{vec}(\mathcal{X}),$$

where W is of the size $m_{out}n^2 \times m_{in}n^2$.

Example



$n = 224$ and $m_{out} = m_{in} = 64 \Longrightarrow W \in \mathbb{R}^{3\,000\,000 \times 3\,000\,000}$.

What to do?

Outline

Approximating the largest singular value

Exact computation of SVD

Power method

$\sigma_1(W) = \sqrt{\lambda_1(W^\top W)}$, so we can apply the *power method*:

$$x_{k+1} := \frac{(W^\top W)x_k}{\|(W^\top W)x_k\|_2},$$
$$\sigma_{k+1} := \frac{\|Wx_{k+1}\|_2}{\|x_{k+1}\|_2}.$$

We also have fast matrix-vector product on $A = W^\top W$.

$$A^k x_0 = A^k(\alpha_1 v_1 + \alpha_2 v_2 + \dots + \alpha_n v_n) =$$
$$= \alpha_1 \lambda_1^k v_1 + \alpha_2 \lambda_2^k v_2 + \dots =$$
$$= \lambda_1^k \left(\alpha_1 v_1 + \alpha_2 \left(\frac{\lambda_2}{\lambda_1}\right)^k + \dots \right)$$

Spectral normalization

[Miyato et. al '18] apply power method to the reshaped core:

$$R = \texttt{reshape}(\mathcal{C}, [k_1 k_2 m_{in}, m_{out}]), \quad \mathcal{C} \in \mathbb{R}^{k_1 \times k_2 \times m_{in} \times m_{out}}.$$

Why is this related to the actual singular values?

Spectral normalization

[Miyato et. al '18] apply power method to the reshaped core:

$$R = \texttt{reshape}(\mathcal{C}, [k_1 k_2 m_{in}, m_{out}]), \quad \mathcal{C} \in \mathbb{R}^{k_1 \times k_2 \times m_{in} \times m_{out}}.$$

Why is this related to the actual singular values?

Theorem (Singla, Feizi, ICML'21)

$$\sigma_1(W) \leq \sqrt{k_1 k_2} \min\left(\|R\|_2, \|S\|_2, \|T\|_2, \|U\|_2\right),$$

where R, S, T, U are $\mathcal{C}$, reshaped into matrices of the sizes:

$$k_1 k_2 m_{in} \times m_{out}, \quad m_{in} \times k_1 k_2 m_{out}, \quad m_{in} k_1 \times m_{out} k_2, \quad m_{in} k_2 \times m_{out} k_1.$$

Constrainting Lipschitz constant

Projection step [Gouk et. al '2021]

$$\pi_\lambda(W) = \frac{1}{\max\left\{1, \frac{\|W\|_2}{\lambda}\right\}} W,$$

where $\lambda \geq 1$ is a hyperparameter. Then

$$L_{\mathcal{N}\mathcal{N}} \leq \lambda^{\#\text{layers}}.$$

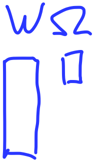
Better methods for estimating $\sigma_1(W)$

- Optimizing on a Krylov subspace:

$$\mathcal{K}_k(W^\top W, x_0) \equiv \mathrm{span}(x_0, (W^\top W)x_0, \ldots, (W^\top W)^{k-1}x_0).$$

- $\|W - UV^\top\|_F \to \min_{U,V\in\mathbb{R}^{N\times r}}$ using alternating optimiation:

- Randomized algorithms (sampling vectors from $\mathrm{Im}(W^\top W)$):



Outline

Approximating the largest singular value

Exact computation of SVD

Matrix of a convolutional layer

If $m \equiv m_{in} = m_{out} = 1$, convolution is 1D and periodic, then

$$y_q = (c * x)_q \equiv \sum_{p=0}^{n-1} c_{(q-p) \bmod n} x_p$$

or in the matrix form $y = Wx$, where:

$$F^* F = n I$$

$$W = \mathrm{circ}(c) \equiv \begin{bmatrix} c_0 & c_{n-1} & \dots & c_1 \\ c_1 & c_0 & & c_2 \\ \vdots & \vdots & \ddots & \vdots \\ c_{n-1} & c_{n-2} & \dots & c_0 \end{bmatrix} \quad - \quad \text{circulant.}$$

$$F = \left(e^{-\frac{2\pi i}{n} pq}\right)_{p,q=0}^{n-1}$$

[illegible]

$$W = c_0 I + c_1 P + c_2 P^2 + \dots + c_{n-1} P^{n-1}$$

$$P = \begin{pmatrix} 0 & & & 1 \\ 1 & 0 & & \\ & \ddots & \ddots & \\ 0 & & 1 & 0 \end{pmatrix}$$

Singular values of a circulant matrix

Diagonalizing a circulant

$$\mathrm{circ}(c) = F^{-1}\mathrm{diag}(Fc)F,$$

where $F_{pq} = e^{-\frac{2\pi i}{n}pq}$ is the Fourier matrix and $F^*F = nI$.

$$F^{-1} = \frac{1}{n}F^*$$

SVD of a circulant

$$\mathrm{circ}(c) = \frac{1}{n}F^*\,\mathrm{diag}(Fc)\,F =$$

$$= \left(\frac{1}{\sqrt{n}}F^*\right)\mathrm{diag}(Fc)\left(\frac{1}{\sqrt{n}}F\right) =$$

$$= \left(\frac{1}{\sqrt{n}}F^*\,\mathrm{diag}(e^{i\theta})\right)\mathrm{diag}(|Fc|)\left(\frac{1}{\sqrt{n}}F\right)$$

Singular values of a circulant matrix

Diagonalizing a circulant

$$\mathrm{circ}(c) = F^{-1}\mathrm{diag}(Fc)F,$$

where $F_{pq} = e^{-\frac{2\pi i}{n}pq}$ is the Fourier matrix and $F^*F = nI$.

SVD of a circulant

$$\mathrm{circ}(c) = \left(\frac{1}{\sqrt{n}}F^*\,\mathrm{diag}(e^{i\theta})\right)\mathrm{diag}(|Fc|)\left(\frac{1}{\sqrt{n}}F\right).$$

The `numpy` code for singular values:

```
s = np.linalg.fft(c)
s = np.abs(s)
```

Complexity: $\mathcal{O}(n\log n)$.

Matrix of a convolutional layer

Let convolution be 1D, periodic and multichannel ($m \equiv m_{in} = m_{out} > 1$):

$$y_{qi} = \sum_{j=1}^{m} \sum_{p} \mathcal{C}_{(q-p) \bmod n \, ij} x_{pj}$$

or in the matrix form $\mathrm{vec}(\mathcal{Y}) = W \, \mathrm{vec}(\mathcal{X})$, where:

Matrix of a convolutional layer

Let convolution be 1D, periodic and multichannel ($m \equiv m_{in} = m_{out} > 1$):

$$y_{qi} = \sum_{j=1}^{m} \sum_{p} \mathcal{C}_{(q-p) \bmod n \, ij} x_{pj}$$

or in the matrix form $\mathrm{vec}(\mathcal{Y}) = W \, \mathrm{vec}(\mathcal{X})$, where:

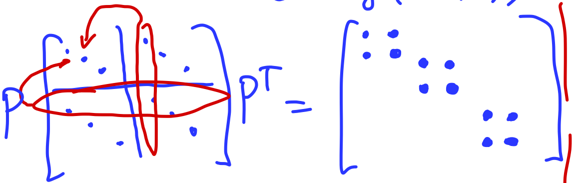
$$W = \begin{bmatrix} \mathrm{circ}(\mathcal{C}_{:,0,0}) & \dots & \mathrm{circ}(\mathcal{C}_{:,0,n-1}) \\ \vdots & \ddots & \vdots \\ \mathrm{circ}(\mathcal{C}_{:,n-1,0}) & \dots & \mathrm{circ}(\mathcal{C}_{:,n-1,n-1}) \end{bmatrix}.$$

Singular values of a convolutional layer

$$W = \begin{bmatrix} \mathrm{circ}(\mathcal{C}_{:,0,0}) & \dots & \mathrm{circ}(\mathcal{C}_{:,0,n-1}) \\ \vdots & \ddots & \vdots \\ \mathrm{circ}(\mathcal{C}_{:,n-1,0}) & \dots & \mathrm{circ}(\mathcal{C}_{:,n-1,n-1}) \end{bmatrix} =$$

$$= \begin{bmatrix} F^{-1}\mathrm{diag}(FC_{:,0,0})F & \dots & F^{-1}\mathrm{diag}(FC_{:,0,n-1})F \\ \vdots & \ddots & \vdots \\ F^{-1}\mathrm{diag}(FC_{:,n-1,0})F & \dots & F^{-1}\mathrm{diag}(FC_{:,n-1,n-1})F \end{bmatrix} =$$

$$= \begin{bmatrix} F^{-1} & & 0 \\ & \ddots & \\ 0 & & F^{-1} \end{bmatrix} \begin{bmatrix} \mathrm{diag}(FC_{:,0,0}) & \dots & \mathrm{diag}(FC_{:,0,n-1}) \\ \vdots & & \vdots \\ \mathrm{diag}(FC_{:,n-1,0}) & \dots & \mathrm{diag}(FC_{:,n-1,n-1}) \end{bmatrix} \begin{bmatrix} F & & 0 \\ & \ddots & \\ 0 & & F \end{bmatrix}$$



S = fft(C, axis=[0])

s = svd(S, axis=[1,2], compute_uv=false)

Singular values of a convolutional layer

Theorem (Sedghi et. al, ICLR '19)

Let $\widehat{\mathcal{C}} \in \mathbb{R}^{n \times n \times m \times m}$ *be* $\mathcal{C} \in \mathbb{R}^{k \times k \times m \times m}$ *padded with zeroes. Let*

$$P_{ij}^{(p_1 p_2)} = (F^\top \widehat{\mathcal{C}}_{:,:,i,j} F)_{p_1 p_2}. \tag{1}$$

Then the singular values of W are:

$$\sigma(W) = \bigcup_{p_1, p_2 \in \{1, \ldots, n\}} \sigma\left(P^{(p_1 p_2)}\right). \tag{2}$$

```
s = np.fft.fft2(kernel, input_shape, axes=[0, 1])
s = np.linalg.svd(s, compute_uv=False)
```

$K^T = -K$

$\exp(K)$

Singular values of a convolutional layer

Theorem (Sedghi et. al, ICLR '19)

Let $\widehat{\mathcal{C}} \in \mathbb{R}^{n\times n\times m\times m}$ be $\mathcal{C} \in \mathbb{R}^{k\times k\times m\times m}$ padded with zeroes. Let

$$P_{ij}^{(p_1 p_2)} = (F^\top \widehat{\mathcal{C}}_{:,:,i,j} F)_{p_1 p_2}. \tag{1}$$

Then the singular values of W are:

$$\sigma(W) = \bigcup_{p_1, p_2 \in \{1,\dots,n\}} \sigma\left(P^{(p_1 p_2)}\right). \tag{2}$$

```
s = np.fft.fft2(kernel, input_shape, axes=[0, 1])
s = np.linalg.svd(s, compute_uv=False)
```

Complexity: $\mathcal{O}(m^2 n^2 (m + \log n))$.

Conclusions

- Power method: fast, but not accurate;
- Exact computation: still not feasible for high resolution;
- Why are the derived formulas useful for the non-periodic case?