Matrix Operations I

- For efficient implementations, we should conduct convolutional operations by matrix-matrix and matrix-vector operations
- Let's collect images of all channels as the input

$$
\begin{aligned}
&Z^{\text{in},i}\\
=&\begin{bmatrix} z^i_{1,1,1} & z^i_{2,1,1} & \cdots & z^i_{a^{\text{in}},b^{\text{in}},1} \\ \vdots & \vdots & \ddots & \vdots \\ z^i_{1,1,d^{\text{in}}} & z^i_{2,1,d^{\text{in}}} & \cdots & z^i_{a^{\text{in}},b^{\text{in}},d^{\text{in}}} \end{bmatrix}\\
\in& R^{d^{\text{in}} \times a^{\text{in}} b^{\text{in}}}.
\end{aligned}
$$

Matrix Operations II

- Let all filters

$$W = \begin{bmatrix} w^1_{1,1,1} & w^1_{2,1,1} & \cdots & w^1_{h,h,d^{\text{in}}} \\ \vdots & \vdots & \ddots & \vdots \\ w^{d^{\text{out}}}_{1,1,1} & w^{d^{\text{out}}}_{2,1,1} & \cdots & w^{d^{\text{out}}}_{h,h,d^{\text{in}}} \end{bmatrix}$$
$$\in R^{d^{\text{out}} \times hhd^{\text{in}}}$$

be variables (parameters) of the current layer

Matrix Operations III

- Usually a bias term is considered

$$\boldsymbol{b} = \begin{bmatrix} b_1 \\ \vdots \\ b_{d^{\text{out}}} \end{bmatrix} \in R^{d^{\text{out}} \times 1}$$

- Operations at a layer

$$\begin{aligned} S^{\text{out},i} &= W\phi(Z^{\text{in},i}) + \boldsymbol{b}\mathbb{1}^T_{a^{\text{out}}b^{\text{out}}} \\ &\in R^{d^{\text{out}} \times a^{\text{out}}b^{\text{out}}}, \end{aligned} \quad (1)$$

Matrix Operations IV

where

$$\mathbb{1}_{a^{\text{out}} b^{\text{out}}} = \begin{bmatrix} 1 \\ \vdots \\ 1 \end{bmatrix} \in R^{a^{\text{out}} b^{\text{out}} \times 1}.$$

- $\phi(Z^{\text{in},i})$ collects all sub-images in $Z^{\text{in},i}$ into a matrix.

Matrix Operations V

Specifically,

$$\phi(Z^{\text{in},i}) =$$

$$\begin{bmatrix} z^i_{1,1,1} & z^i_{1+s,1,1} & & z^i_{1+(a^{\text{out}}-1)s,1+(b^{\text{out}}-1)s,1} \\ z^i_{2,1,1} & z^i_{2+s,1,1} & & z^i_{2+(a^{\text{out}}-1)s,1+(b^{\text{out}}-1)s,1} \\ \vdots & \vdots & \ldots & \vdots \\ z^i_{h,h,1} & z^i_{h+s,h,1} & & z^i_{h+(a^{\text{out}}-1)s,h+(b^{\text{out}}-1)s,1} \\ \vdots & \vdots & & \vdots \\ z^i_{h,h,d^{\text{in}}} & z^i_{h+s,h,d^{\text{in}}} & & z^i_{h+(a^{\text{out}}-1)s,h+(b^{\text{out}}-1)s,d^{\text{in}}} \end{bmatrix}$$

$$\in R^{hhd^{\text{in}} \times a^{\text{out}} b^{\text{out}}}$$

Activation Function I

- Next, an activation function scales each element of $S^{\text{out},i}$ to obtain the output matrix $Z^{\text{out},i}$.

$$Z^{\text{out},i} = \sigma(S^{\text{out},i}) \in R^{d^{\text{out}} \times a^{\text{out}} b^{\text{out}}}. \tag{2}$$

- For CNN, commonly the following RELU activation function

$$\sigma(x) = \max(x, 0) \tag{3}$$

  is used
- Later we need that $\sigma(x)$ is differentiable, but the RELU function is not.

Activation Function II

- Past works such as Krizhevsky et al. (2012) assume

$$\sigma'(x) = \begin{cases} 1 & \text{if } x > 0 \\ 0 & \text{otherwise} \end{cases}$$

The Function $\phi(Z^{\text{in},i})$ I

- In the matrix-matrix product

$$W\phi(Z^{\text{in},i}),$$

  each element is the inner product between a filter and a sub-image

- Clearly ϕ is a linear mapping, so there exists a $0/1$ matrix P_ϕ such that

$$\phi(Z^{\text{in},i}) \equiv \text{mat}\left(P_\phi \text{vec}(Z^{\text{in},i})\right)_{hhd^{\text{in}} \times a^{\text{out}} b^{\text{out}}}, \quad \forall i, \tag{4}$$

The Function $\phi(Z^{\text{in},i})$ II

- $\text{vec}(M)$: all M's columns concatenated to a vector $\boldsymbol{v}$

$$\text{vec}(M) = \begin{bmatrix} M_{:,1} \\ \vdots \\ M_{:,b} \end{bmatrix} \in R^{ab\times 1}, \text{ where } M \in R^{a\times b}$$

- $\text{mat}(\boldsymbol{v})$ is the inverse of $\text{vec}(M)$

$$\text{mat}(\boldsymbol{v})_{a\times b} = \begin{bmatrix} v_1 & & v_{(b-1)a+1} \\ \vdots & \cdots & \vdots \\ v_a & & v_{ba} \end{bmatrix} \in R^{a\times b}, \tag{5}$$

The Function $\phi(Z^{\text{in},i})$ III

where

$$\boldsymbol{v} \in R^{ab \times 1}.$$

- P_ϕ is a huge matrix:

$$P_\phi \in R^{hhd^{\text{in}} a^{\text{out}} b^{\text{out}} \times d^{\text{in}} a^{\text{in}} b^{\text{in}}}$$

and

$$\phi : R^{d^{\text{in}} \times a^{\text{in}} b^{\text{in}}} \rightarrow R^{hhd^{\text{in}} \times a^{\text{out}} b^{\text{out}}}$$

- The representation of using P_ϕ makes some subsequent derivations easier

The Function $\phi(Z^{\mathrm{in},i})$ IV

- Past works using the form (4) include, for example, Vedaldi and Lenc (2015)

Optimization Problem I

- We collect all weights to a vector variable $\boldsymbol{\theta}$.

$$\boldsymbol{\theta} = \begin{bmatrix} \text{vec}(W^1) \\ \boldsymbol{b}^1 \\ \vdots \\ \text{vec}(W^L) \\ \boldsymbol{b}^L \end{bmatrix} \in R^n, \quad n: \text{total \# variables}$$

- The output of the last layer L is a vector $\boldsymbol{z}^{L+1,i}(\boldsymbol{\theta})$.
- Consider any loss function such as the squared loss

$$\xi_i(\boldsymbol{\theta}) = ||\boldsymbol{z}^{L+1,i}(\boldsymbol{\theta}) - \boldsymbol{y}^i||^2.$$

Optimization Problem II

- The optimization problem is

$$\min_{\boldsymbol{\theta}} f(\boldsymbol{\theta}),$$

where

$$f(\boldsymbol{\theta}) = \frac{1}{2C}\boldsymbol{\theta}^T\boldsymbol{\theta} + \frac{1}{l}\sum\nolimits_{i=1}^{l} \xi(\boldsymbol{z}^{L+1,i}(\boldsymbol{\theta}); \boldsymbol{y}^i, Z^{1,i})$$

C: regularization parameter.

- The formulation is almost the same as that for fully connected networks

Optimization Problem III

- Note that we divide the sum of training losses by the number of training data

  Thus the second term becomes the average training loss

References I

A. Krizhevsky, I. Sutskever, and G. E. Hinton. ImageNet classification with deep convolutional neural networks. In F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 25*, pages 1097–1105. 2012.

A. Vedaldi and K. Lenc. MatConvNet: Convolutional neural networks for matlab. In *Proceedings of the 23rd ACM International Conference on Multimedia*, pages 689–692, 2015.