Derivation of Backpropagation in Convolutional Neural Network (CNN)

Zhifei Zhang

University of Tennessee, Knoxvill, TN

October 18, 2016

Abstract— Derivation of backpropagation in convolutional neural network (CNN) is conducted based on an example with two convolutional layers. The step-by-step derivation is helpful for beginners. First, the feedforward procedure is claimed, and then the backpropagation is derived based on the example.

1 Feedforward

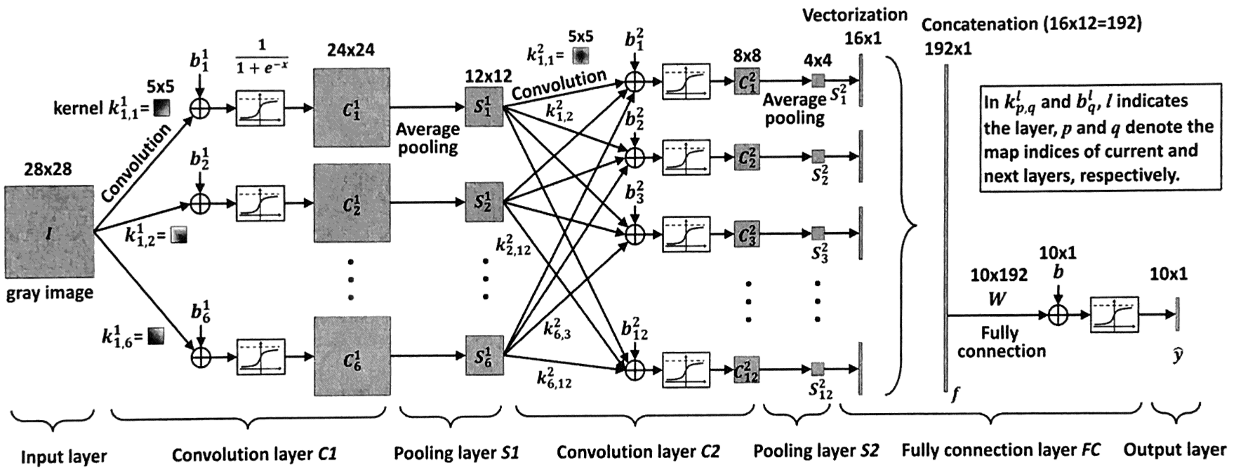


Figure 1: The structure of CNN example that will be discussed in this paper. It is exactly the same to the structure used in the demo of Matlab DeepLearnToolbox [1]. All later derivation will use the same notations in this figure.

1.1 Initialization of Parameters

The parameters are:

- **C1 layer**, $k^1_{1,p}$ (size 5×5) and b^1_p (size 1×1), $p = 1, 2, \cdots 6$
- **C2 layer**, $k^2_{p,q}$ (size 5×5) and b^2_q (size 1×1), $q = 1, 2, \cdots 12$
- **FC layer**, W (size 10×192) and b (size 10×1)

All bias, b_p^1, b_q^2, and b, are initialize to zero. The others are drew randomly from a uniform distribution defined based on the kernel size and number of input and output maps on corresponding layers [2] (see section 4.6 of [2]).

$$k_{1,p}^1 \sim U\left(\pm\sqrt{\frac{6}{(1+6)\times 5^2}}\right) \quad \checkmark \tag{1}$$

$$k_{p,q}^2 \sim U\left(\pm\sqrt{\frac{6}{(6+12)\times 5^2}}\right) \quad \checkmark \tag{2}$$

$$W \sim U\left(\pm\sqrt{\frac{6}{192+10}}\right) \quad \checkmark \tag{3}$$

where $U(\pm x)$ denotes a uniform distribution with upper and lower bounds of $\pm x$. Totally, the number of parameters is $(5\times 5+1)\times 6+(5\times 5\times 6+1)\times 12+10\times 192+10=3898$.

1.2 Convolution Layer C1

$$C_p^1 = \sigma(I * k_{1,p}^1 + b_p^1), \text{ where } \sigma(x) = \frac{1}{1+\exp^{-x}} \tag{4}$$

$$C_p^1(i,j) = \sigma\left(\sum_{u=-2}^{2}\sum_{v=-2}^{2} I(i-u, j-v)\cdot k_{1,p}^1(u,v) + b_p^1\right) \tag{5}$$

where $p = 1, 2, \cdots, 6$ because there are 6 feature maps on C1 layer, $*$ denotes the convolution, and i, j are row and column indices of the feature map. Only keeping those parts of the convolution that are computed without the zero-padded edges, the size of C_p^1 is 24×24, rather than 28×28 like I.

1.3 Pooling Layer S1

$$S_p^1(i,j) = \frac{1}{4}\sum_{u=0}^{1}\sum_{v=0}^{1} C_p^1(2i-u, 2j-v),\ i,j = 1,2,\cdots,12 \tag{6}$$

1.4 Convolution Layer C2

$$C_q^2 = \sigma\left(\sum_{p=1}^{6} S_p^1 * k_{p,q}^2 + b_q^2\right) \tag{7}$$

$$C_q^2(i,j) = \sigma\left(\sum_{p=1}^{6}\sum_{u=-2}^{2}\sum_{v=-2}^{2} S_p^1(i-u, j-v)\cdot k_{p,q}^2(u,v) + b_q^2\right) \tag{8}$$

where $q = 1, 2, \cdots, 12$ because there are 12 feature maps on C2 layer. Only keeping those parts of the convolution that are computed without the zero-padded edges, the size of C_q^2 is 8×8, rather than 12×12 like S_p^1.

1.5 Pooling Layer S2

$$S_q^2(i,j) = \frac{1}{4}\sum_{u=0}^{1}\sum_{v=0}^{1} C_q^2(2i-u, 2j-v),\ i,j = 1,2,\cdots,4 \tag{9}$$

1.6 Vectorization and Concatenation

Each S_q^2 is a 4×4 matrix, and there are 12 such matrices on the S2 layer. First, each S_q^2 is vectorized by column scan, then all 12 vectors are concatenated to form a long vector with the length of $4 \times 4 \times 12 = 192$. We denote this process by

$$f = F\left(\{S_q^2\}_{q=1,2,\cdots,12}\right), \tag{10}$$

and the reverse process is

$$\{S_q^2\}_{q=1,2,\cdots,12} = F^{-1}(f). \tag{11}$$

1.7 Fully Connection Layer FC

$$\hat{y} = \sigma(W \times f + b) \tag{12}$$

1.8 Loss Function

Assuming the true label is y, the loss function is express by

$$L = \frac{1}{2}\sum_{i=1}^{10}(\hat{y}(i) - y(i))^2 \tag{13}$$

2 Backpropagation

In the backpropagation, we'll update the parameters from the back to start, namely W and b, $k_{p,q}^2$ and b_q^2, $k_{1,p}^1$ and b_p^1.

2.1 ΔW (size 10×192)

$$\begin{aligned}
\Delta W(i,j) &= \frac{\partial L}{\partial W(i,j)} && (14)\\
&= \frac{\partial L}{\partial \hat{y}(i)} \cdot \frac{\partial \hat{y}(i)}{\partial W(i,j)} && (15)\\
&= (\hat{y}(i) - y(i)) \cdot \frac{\partial}{\partial W(i,j)}\sigma\left(\sum_{j=1}^{192} W(i,j) \times f(j) + b(i)\right) && (16)\\
&= (\hat{y}(i) - y(i)) \cdot \hat{y}(i)(1 - \hat{y}(i)) \cdot f(j) && (17) \checkmark
\end{aligned}$$

Let $\Delta\hat{y}(i) = (\hat{y}(i) - y(i)) \cdot \hat{y}(i)(1 - \hat{y}(i))$, whose size is 10×1, then

$$\begin{aligned}
\Delta W(i,j) &= \Delta\hat{y}(i) \cdot f(j) && (18)\\
\Longrightarrow \Delta W &= \Delta\hat{y} \times f^T && (19)
\end{aligned}$$

2.2 Δb (size 10×1)

$$\begin{aligned}
\Delta b(i) &= \frac{\partial L}{\partial b(i)} && (20)\\
&= \frac{\partial L}{\partial \hat{y}(i)} \cdot \frac{\partial \hat{y}(i)}{\partial b(i)} && (21)\\
&= (\hat{y}(i) - y(i)) \cdot \frac{\partial}{\partial b(i)} \sigma\left(\sum_{j=1}^{192} W(i,j) \times f(j) + b(i)\right) && (22)\\
&= (\hat{y}(i) - y(i)) \cdot \hat{y}(i)(1 - \hat{y}(i)) && (23)\\
\Longrightarrow \Delta b &= \Delta \hat{y} && (24)
\end{aligned}$$

2.3 $\Delta k^2_{p,q}$ (size 5×5)

Because of concatenation, vectorization, and pooling, we need to compute the back-propagation error ΔC^2_q on C2 layer before calculating $\Delta k^2_{p,q}$.

$$\begin{aligned}
\Delta f(j) &= \frac{\partial L}{\partial f} && (25)\\
&= \sum_{i=1}^{10} \frac{\partial L}{\partial \hat{y}(i)} \cdot \frac{\partial \hat{y}(i)}{\partial f(j)} && (26)\\
&= (\hat{y}(i) - y(i)) \cdot \frac{\partial}{\partial f(j)} \sigma\left(\sum_{j=1}^{192} W(i,j) \times f(j) + b(i)\right) && (27)\\
&= \sum_{i=1}^{10} (\hat{y}(i) - y(i)) \cdot \hat{y}(i)(1 - \hat{y}(i)) \cdot W(i,j) && (28)\\
&= \sum_{i=1}^{10} \Delta \hat{y}(i) \cdot W(i,j) && (29)\\
\Longrightarrow \Delta f &= W^T \times \Delta \hat{y} && (30)
\end{aligned}$$

$$\nearrow = \sum_{j=1}^{10} W^T(j,i) \cdot \Delta y(i) = W_{j1}\Delta y_1 + W_{j2}\Delta y_2 + \cdots + W_{j10}\Delta y_{10}$$

$$\therefore \Delta f = \begin{bmatrix} \Delta f(1) \\ \vdots \\ \Delta f(n) \end{bmatrix} = \begin{bmatrix} W_{11}\Delta y_1 + W_{12}\Delta y_2 + \cdots + W_{110}\Delta y_{10} \\ \vdots \\ W_{j1}\Delta y_1 + W_{j2}\Delta y_2 + \cdots + W_{j10}\Delta y_{10} \\ \vdots \\ W_{n1}\Delta y_1 + W_{n2}\Delta y_2 + \cdots + W_{n10}\Delta y_{10} \end{bmatrix} ✓$$

$$= \begin{bmatrix} W_{11} & W_{12} & \cdots & W_{10} \\ \vdots & \vdots & & \\ W_{j1} & W_{j2} & \cdots & W_{j10} \\ \vdots & \vdots & & \\ W_{n1} & W_{n2} & \cdots & W_{n10} \end{bmatrix} \cdot \begin{bmatrix} \Delta y_1 \\ \Delta y_2 \\ \vdots \\ \Delta y_{10} \end{bmatrix}$$

$$= W^T \cdot \Delta y$$

From section 1.6, we reshape the long error vector Δf (size 192×1) by

$$\{\Delta S^2_q\}_{q=1,2,\cdots,12} = F^{-1}(\Delta f), \quad ✓ \tag{31}$$

which gets the error on S2 layer (twelve 4×4 error maps). Because there is no parameters on S2 layer, we do not need to do any derivative stuff. Then, upsampling is performed to obtain the error on C2 layer.

$$\Delta C^2_q(i,j) = \frac{1}{4} \Delta S^2_q\left(\lceil i/2 \rceil, \lceil j/2 \rceil\right), \; i,j = 1,2,\cdots,8 \quad ✓ \tag{32}$$

in my cnn : $8 \times 8 \xrightarrow{\text{upsampling}} 16 \times 16$

floor?

where $\lceil \cdot \rceil$ denotes the ceiling function. Note that the size of ΔS_q^2 and ΔC_q^2 are 4×4 and 8×8, respectively. Now, we are ready to derive $\Delta k_{p,q}^2$.

$$\Delta k_{p,q}^2(u,v) = \frac{\partial L}{\partial k_{p,q}^2(u,v)} \tag{33}$$

$$= \sum_{i=1}^{8}\sum_{j=1}^{8} \frac{\partial L}{\partial C_q^2(i,j)} \cdot \frac{\partial C_q^2(i,j)}{\partial k_{p,q}^2(u,v)} \tag{34}$$

$$= \sum_{i=1}^{8}\sum_{j=1}^{8} \Delta C_q^2(i,j) \cdot \frac{\partial}{\partial k_{p,q}^2(u,v)} \sigma\left(\sum_{p=1}^{6}\sum_{u=-2}^{2}\sum_{v=-2}^{2} S_p^1(i-u,j-v) \cdot k_{p,q}^2(u,v) + b_q^2\right) \tag{35}$$

$$= \sum_{i=1}^{8}\sum_{j=1}^{8} \Delta C_q^2(i,j) \cdot C_q^2(i,j)\left(1 - C_q^2(i,j)\right) \cdot S_p^1(i-u,j-v) \tag{36}$$

Let

$$\Delta C_{q,\sigma}^2(i,j) = \Delta C_q^2(i,j) \cdot C_q^2(i,j)\left(1 - C_q^2(i,j)\right), \tag{37}$$

which is actually the error before sigmoid function on C2 layer. Therefore,

$$C_{q,\sigma}^2(i,j) = \sum_{p=1}^{6}\sum_{u=-2}^{2}\sum_{v=-2}^{2} S_p^1(i-u,j-v) \cdot k_{p,q}^2(u,v) + b_q^2 \tag{38}$$

Rotating S_p^1 180 degrees, we get $S_{p,rot180}^1$, thus $S_{p,rot180}^1(u-i,v-j) = S_p^1(i-u,j-v)$. Therefore, $\Delta k_{p,q}^2$ can be expressed by

$$\Delta k_{p,q}^2(u,v) = \sum_{i=1}^{8}\sum_{j=1}^{8} S_{p,rot180}^1(u-i,v-j) \cdot \Delta C_{q,\sigma}^2(i,j) \tag{39}$$

$$\Longrightarrow \Delta k_{p,q}^2 = S_{p,rot180}^1 * \Delta C_{q,\sigma}^2 \qquad \text{see equation (5) for definition of convolution} \tag{40}$$ ✓

2.4 Δb_q^2 (size 1×1)

$\Delta C_{q,\sigma}^2$: d_l2_output_before_activation , shape = (12, 16, 16)

$$\Delta b_q^2 = \frac{\partial L}{\partial b_q^2} \tag{41}$$

$$= \sum_{i=1}^{8}\sum_{j=1}^{8} \frac{\partial L}{\partial C_q^2(i,j)} \cdot \frac{\partial C_q^2(i,j)}{\partial b_q^2} \tag{42}$$

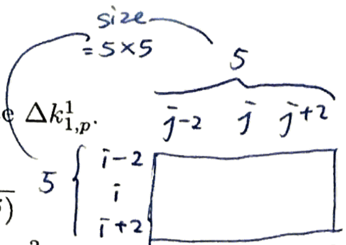
$$= \sum_{i=1}^{8}\sum_{j=1}^{8} \Delta C_q^2(i,j) \cdot \frac{\partial}{\partial b_q^2} \sigma\left(\sum_{p=1}^{6}\sum_{u=-2}^{2}\sum_{v=-2}^{2} S_p^1(i-u,j-v) \cdot k_{p,q}^2(u,v) + b_q^2\right) \tag{43}$$

$$= \sum_{i=1}^{8}\sum_{j=1}^{8} \Delta C_q^2(i,j) \cdot C_q^2(i,j)\left(1 - C_q^2(i,j)\right) \tag{44}$$

$$= \sum_{i=1}^{8}\sum_{j=1}^{8} \Delta C_{q,\sigma}^2(i,j) \qquad q = 1, 2, \cdots, 12 \tag{45}$$ ✓

2.5 $\Delta k_{1,p}^1$ (size 5×5)

Similar to the derivation of $\Delta k_{p,q}^2$, we should first obtain ΔS_p^1, the error on S1 layer. Then, upsampling will be performed to get ΔC_p^1, the error on C1 layer. Finally, following the same way, we can calculate $\Delta k^1_{1,p}$.



$i,j=1,\cdots,20$

$u=-2,\cdots,2$

$v=-2,\cdots 2.$

$i+u=-1,0,\cdots,22$

$j+v=-1,0,\cdots,22$

长度是24

$$\Delta S^1_p(i,j) = \frac{\partial L}{\partial S^1_p(i,j)} \tag{46}$$
$$= \sum_{q=1}^{12}\sum_{u=-2}^{2}\sum_{v=-2}^{2} \frac{\partial L}{\partial C^2_{q,\sigma}(i+u,j+v)} \cdot \frac{\partial C^2_{q,\sigma}(i+u,j+v)}{\partial S^1_p(i,j)} \tag{47}$$
$$= \sum_{q=1}^{12}\sum_{u=-2}^{2}\sum_{v=-2}^{2} \Delta C^2_{q,\sigma}(i+u,j+v) \cdot \frac{\partial}{\partial S^1_p(i,j)} \left(\sum_{p=1}^{6}\sum_{u=-2}^{2}\sum_{v=-2}^{2} S^1_p(i,j)\cdot k^2_{p,q}(u,v) + b^2_q\right) \tag{48}$$
$$= \sum_{q=1}^{12}\sum_{u=-2}^{2}\sum_{v=-2}^{2} \Delta C^2_{q,\sigma}(i+u,j+v) \cdot k^2_{p,q}(u,v) \tag{49}$$

~~$\Delta C^2_{q,\sigma}$的长度尺寸~~
~~必须是~~ Rotating $k^2_{p,q}$ 180 degrees, we get $k^2_{p,q,rot180}(-u,-v) = k^2_{p,q}(u,v)$. Therefore,
~~24，才能有效~~

$C^2_{q,\sigma}$ 索引了 $i+u$, $j+v$

$$\Delta S^1_p(i,j) = \sum_{q=1}^{12}\sum_{u=-2}^{2}\sum_{v=-2}^{2} \Delta C^2_{q,\sigma}(i-(-u),j-(-v)) \cdot k^2_{p,q,rot180}(-u,-v) \tag{50}$$

∴ $C^2_{q,\sigma}$ 的尺寸也是24

$$\Longrightarrow \Delta S^1_p = \sum_{q=1}^{12} \Delta C^2_{q,\sigma} * k^2_{p,q,rot180} \tag{51}$$ ✓

可是正向计算的结果，尺寸是16

(paddle, size=4)

By upsampling, we get the error on C1 layer,

矩阵四周各加4行0

(paddle)

$$\Delta C^1_p(i,j) = \frac{1}{4}\Delta S^1_p\left(\lceil i/2\rceil, \lceil j/2\rceil\right),\ i,j=1,2,\cdots,24 \tag{52}$$ ✓

Now, we are ready to calculate $\Delta k^1_{1,p}$,

$$\Delta k^1_{1,p}(u,v) = \frac{\partial L}{\partial k^1_{1,p}(u,v)} \tag{53}$$
$$= \sum_{i=1}^{24}\sum_{j=1}^{24} \frac{\partial L}{\partial C^1_p(i,j)} \cdot \frac{\partial C^1_p(i,j)}{\partial k^1_{1,p}(u,v)} \tag{54}$$

I = Image

$$= \sum_{i=1}^{24}\sum_{j=1}^{24} \Delta C^1_p(i,j) \cdot \frac{\partial}{\partial k^1_{1,p}(u,v)} \sigma\left(\sum_{u=-2}^{2}\sum_{v=-2}^{2} I(i-u,j-v)\cdot k^1_{1,p}(u,v) + b^1_p\right) \tag{55}$$
$$= \sum_{i=1}^{24}\sum_{j=1}^{24} \Delta C^1_p(i,j)\cdot C^1_p(i,j)\left(1 - C^1_p(i,j)\right)\cdot I(i-u,j-v) \tag{56}$$

By the same token, rotate I 180 degrees, and let

$$\Delta C^1_{p,\sigma}(i,j) = \Delta C^1_p(i,j)\cdot C^1_p(i,j)\left(1 - C^1_p(i,j)\right). \tag{57}$$

$\Delta C'_{p,\sigma}$: d_l1_output_before_activation, shape = (6,40,40)

Finally,

$$\Delta k^1_{1,p}(u,v) = \sum_{i=1}^{24}\sum_{j=1}^{24} I_{rot180}(u-i,v-j)\cdot \Delta C^1_{p,\sigma}(i,j) \tag{58}$$
$$\Longrightarrow \Delta k^1_{1,p} = I_{rot180} * \Delta C^1_{p,\sigma} \tag{59}$$ ✓

2.6 Δb^1_p (size 1×1)

$$
\begin{aligned}
\Delta b^1_p &= \frac{\partial L}{\partial b^1_p} \tag{60}\\
&= \sum_{i=1}^{24}\sum_{j=1}^{24} \frac{\partial L}{\partial C^1_p(i,j)} \cdot \frac{\partial C^1_p(i,j)}{\partial b^1_p} \tag{61}\\
&= \sum_{i=1}^{24}\sum_{j=1}^{24} \Delta C^1_p(i,j) \cdot \frac{\partial}{\partial b^1_p}\sigma\left(\sum_{u=-2}^{2}\sum_{v=-2}^{2} I(i-u,j-v)\cdot k^1_{1,p}(u,v) + b^1_p\right) \tag{62}\\
&= \sum_{i=1}^{24}\sum_{j=1}^{24} \Delta C^1_p(i,j) \cdot C^1_p(i,j)\left(1 - C^1_p(i,j)\right) \tag{63}\\
&= \sum_{i=1}^{24}\sum_{j=1}^{24} \Delta C^1_{p,\sigma}(i,j) \tag{64}
\end{aligned}
$$

3 Parameter Update

We need to set a learning rate $\alpha \in (0,1]$.

$$
\begin{aligned}
k^1_{1,p} &\leftarrow k^1_{1,p} - \alpha \cdot \Delta k^1_{1,p} \tag*{(65) $\checkmark$}\\
b^1_p &\leftarrow b^1_p - \alpha \cdot \Delta b^1_p \tag*{(66) $\checkmark$}\\
k^2_{p,q} &\leftarrow k^2_{p,q} - \alpha \cdot \Delta k^2_{p,q} \tag*{(67) $\checkmark$}\\
b^2_q &\leftarrow b^2_q - \alpha \cdot \Delta b^2_q \tag*{(68) $\checkmark$}\\
W &\leftarrow W - \alpha \cdot \Delta W \tag*{(69) $\checkmark$}\\
b &\leftarrow b - \alpha \cdot \Delta b \tag*{(70) $\checkmark$}
\end{aligned}
$$

References

[1] Palm, Rasmus Berg. “Prediction as a candidate for learning deep hierarchical models of data.” *Technical University of Denmark* 5 (2012).

[2] LeCun, Yann A., Leon Bottou, Genevieve B. Orr, and Klaus-Robert MÃijller. “Efficient backprop.” In *Neural networks: Tricks of the trade*, pp. 9-48. Springer Berlin Heidelberg, 2012.