An Elaboration of Convolutional Neural Network

M. M. Chiou

***Abstract*— The computation of Convolution Neural Network (CNN) has been elaborated. One illustrative example of CNN with 6 layers is present.**

***Index Terms*—**

Conovlutional neural network (CNN), originally proposed by LeCun [1], is a neural network model with three key architectural ideas: local receptive fields, weight sharing, and sub-sampling in the spatial domain. The network is designed for the recognition of two-dimensional visual patterns. CNN has two strengths. First, feature extraction and classification are integrated into one structure which are adaptive. Second, it is relatively invariant to gemetric, local distoritons in the image. CNN has been used for applications including hand-written digit recognition, face detection, and face recognition.

The report is structured as follows. Section I describes architectural aspects of the convolutional neural networks. The forward and backward problems (computation) are present in section II and III, respectively. The illustrative example is given in section IV.

I. CNN Network Model Overview

CNNs are desinged to process 2D image. Fig.1 shows the

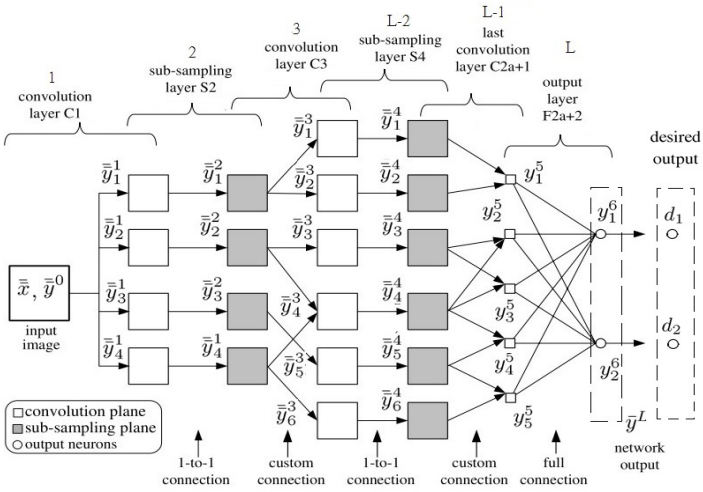

Fig. 1. Layers architecture in a CNN. Note that the last convolution layer is fully connected. $\bar{\bar{y}}_n^\ell, \ell = 1, 2, \cdots, L-2$, is the nth feature map (2-D output) in ℓ layer.

layers architecture in a CNN, which consists of three main types of layers: (i) convolution layers, (ii) sub-sampling layers, and (iii) an output layer. $\bar{\bar{x}} = \bar{\bar{y}}^0$ is the input image. $\bar{\bar{y}}_n^\ell, \ell = 1, 2, \cdots, L-2$, is the nth feature map in ℓ layer. y_n^ℓ, $\ell = L-1, L$ are the nth output of the ℓ layer. Noting that the super script ℓ denotes the order of layer, instead of power of that parameter. Network layers are arranged in a feed-forward structure: each convolution layer is followed by a sub-sampling layer, and each subsampling layer is followed by convolution layer. The last convolution layer is followed by the output layer. The convolution and sub-sampling layers are considered as 2D layers, while the output layer is considered as a 1D layer. Each 2D layer has several planes, and a plane consists of neurons that are arranged in a 2D array. The output of a plane is called a feature map (That is, the output of the convolution layer and sub-sampling layer are called feature map).

In a convolutional layer, each plane is connected to one or more feature maps of the preceding layer. A connection is associated with a convolution mask, which is a 2D matrix of adjustable entries called weights. Each plane computes the convolution between its 2D intputs and its convolution masks. The convolution outputs are summed together and then added with an adjustable scalar, known as bias term. An activation function is applied on the result to obtain the plane's output. The plane output is a 2D matrix called a feature map; this name arises because each convolution output indicates the presence of a visual feature at a given pixel location. A convolution layer produces one or more feature maps. Each feature map is connected to one plane in the next sub-sampling layer.

A sub-sampling layer has the same number of planes as the preceding convolution layer. A subsampling plane divides its 2D input into non-overlapping blocks of size 2x2 pixels. For each block, the sum of four pixels is calculated; this sum is multiplied by an adjustable weight before being added to a bias term. The result is passed through an activation function to produce an output for the 2x2 block. Each subsampling plane reduces its input size by half, along each dimension. A feature map in a sub-sampling layer is connected to one or more planes in the next convolution layer.

In the last convolution layer, each plane is connected to exactly one preceding feature map. This layer uses convolution masks that have exactly the same size as its input feature maps. Each plane in the last convolution layer will produce one scalar output. The outputs from all planes in this layer are connected to the output layer.

The output layer can be constructed from sigmoidal neurons or radial-basis function (RBF) neurons. The sigmoidal neurons for the output layer is considered in this work. The outputs of output layer are considered as the network outputs. In applications of visual pattern classification, these outputs indicate the category of the input image.

II. Mathematical Model

Table.I summarizes the notation used to describe the functional aspects of CNN. The symbol ℓ denotes the index of a network layer. The layer index ℓ goes from 1 to L, and L is the number of network layers. a is a positive integer, and $L = 2a + 2$. There will be $a + 1$ convolution layer, a

TABLE I
MATHEMATICAL NOTATION FOR CNN

description	symbol
input image size	$H_0 \times W_0$
input image pixel	$x(i,j)$ or $y_1^0(i,j)$
layer index	ℓ
number of layers	$L = 2a + 2$
convolution layers	C^1, C^3, $\cdots$, C^{2a+1}
sub-sampling layers	S^1, S^3, $\cdots$, S^{2a}
output layer	F^{2a+2}
activation function of layer ℓ	f_ℓ
number of feature maps in layer ℓ	N_ℓ
size of convolution mask for layer	$h_\ell \times w_\ell$
convolution mask from feature map m in layer $S^{\ell-1}$ to feature map n in layer C^ℓ	$\{w_{m,n}^\ell(i,j)\}$
weight for feature map n in layer S^ℓ	w_n^ℓ
bias for feature map n in convolution layer C^ℓ	b_n^ℓ
bias for feature map n in sub-sampling layer S^ℓ	b_n^ℓ
feature map n in layer ℓ	$y_n^\ell(i,j)$
size of a feature map in layer ℓ	$H_\ell \times W_\ell$

subsampling layer and one output layer. Let N_ℓ be the number of feature maps in layer ℓ, and $f_\ell(\cdot)$ be the activation function of layer ℓ. $\bar{\bar{y}}_n^\ell$ be the nth feature map (output) of layer ℓ.

A. Convolution layer

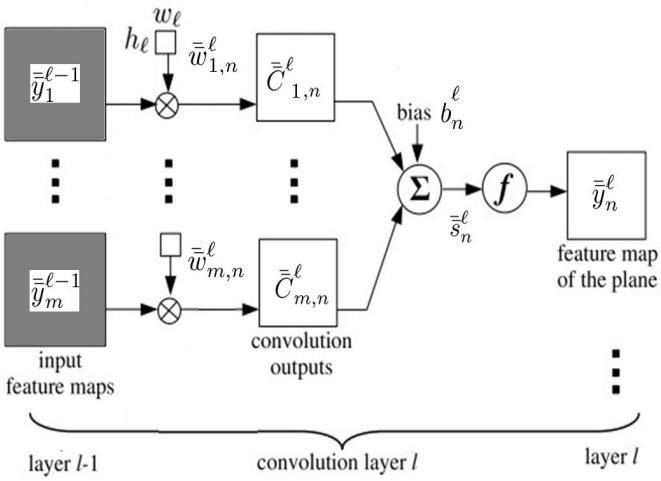

Fig. 2. A convolution layer in a CNN.

Fig.2 shows a convolution layer in CNN. Considering the nth feature map in a convolution layer ℓ, $\ell = 1, 3, \cdots, 2a+1$, $\bar{\bar{w}}_{m,n}^\ell = \{w_{m,n}^\ell(i,j)\}$ is the convolution mask, which will take in feature map m in layer $(\ell - 1)$ and generate feature map n in layer ℓ. b_n^ℓ is the bias term assoicated with feature map n.

Fig.3 shows an illustrative example for the second convolution layer, layer 3. The second and fourth feature maps in second layer, $\bar{\bar{y}}_2^2$ and $\bar{\bar{y}}_4^2$, serve as input sources for the fourth feature map in third layer, $\bar{\bar{y}}_4^3$.

Fig.4 shows another illustrative example for the first convolution layer. The feature maps of 1, 2, 3 and 4 in first layer,

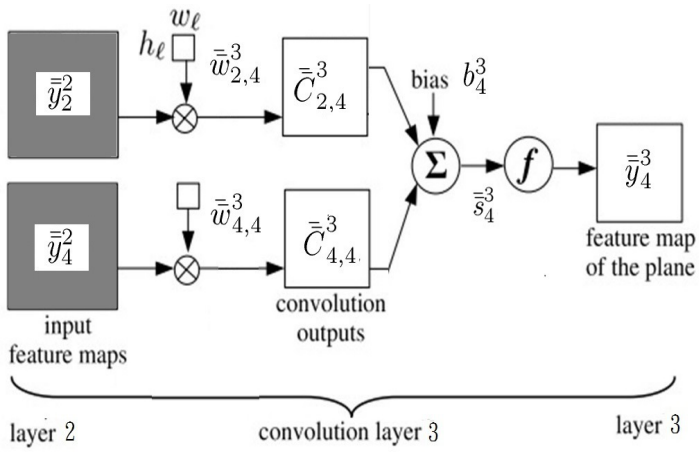

Fig. 3. The second convolution layer (layer 3) as an illustrative example.

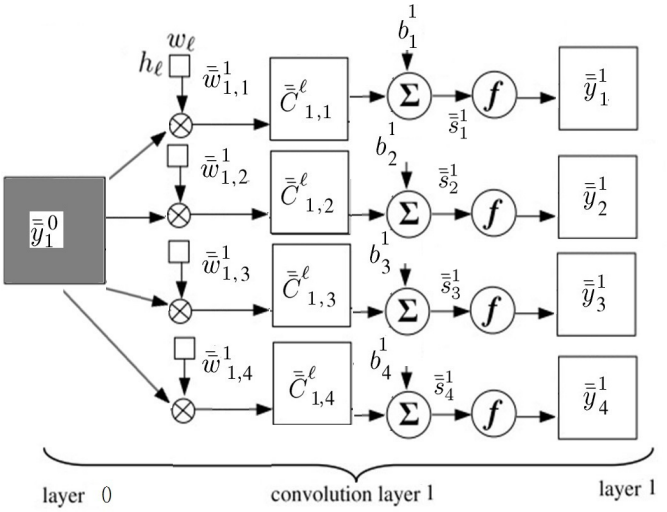

Fig. 4. First convolution layer as an illustrative example.

$\bar{\bar{y}}_1^1$, $\bar{\bar{y}}_2^1$, $\bar{\bar{y}}_3^1$, $\bar{\bar{y}}_4^1$, have idendical input source of the image map $\bar{\bar{y}}_1^0$ in the input layer; however, different convolution masks are applied, $\bar{\bar{w}}_{1,1}^1$, $\bar{\bar{w}}_{1,2}^1$, $\bar{\bar{w}}_{1,3}^1$, $\bar{\bar{w}}_{1,4}^1$.

Fig.5 shows the schematic of U^ℓ and $V^{\ell+1}$, which are applied to describe the connection between successive layer. U_n^ℓ stores the next elements connection for nth neuron (feature map) in layer ℓ to $\ell + 1$ layer. $V_m^{\ell+1}$ stores the preceding elements conection for mth neuron (feature map) in layer $\ell+1$ to ℓ layer. In this illustrative example, the first neuron in layer ℓ is connected to the first, second and fourth neurons in the layer $\ell + 1$ (next layer), and it renders

$$U_1^\ell = [U_1^\ell(1), U_1^\ell(2), U_1^\ell(3)] = [1, 2, 4]$$

The second neuron in layer $\ell + 1$ is connected to the first, second and third neruons in the ℓ layer (preceding neuron), and it renders

$$V_2^{\ell+1} = [V_2^{\ell+1}(1), V_2^{\ell+1}(2), V_2^{\ell+1}(3)] = [1, 2, 3]$$

Note that the U in ℓ layer, U^ℓ, and V in $\ell + 1$ layer, $V^{\ell+1}$

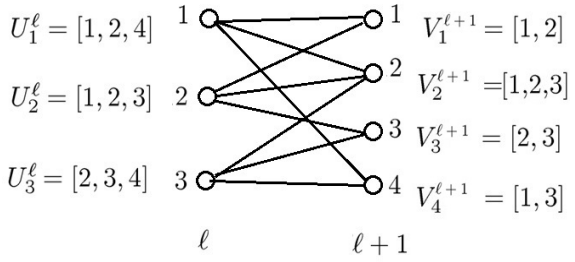

Fig. 5. Schematic of U^ℓ and $V^{\ell+1}$, which are applied to describe the connection between successive layer.

carry equivalent information. From fig.5, it indicates that

$$\text{if } m \in U_n^\ell \quad \rightarrow n \in V_m^{\ell+1}$$
$$\text{if } n \in V_m^{\ell+1} \rightarrow m \in U_n^\ell$$

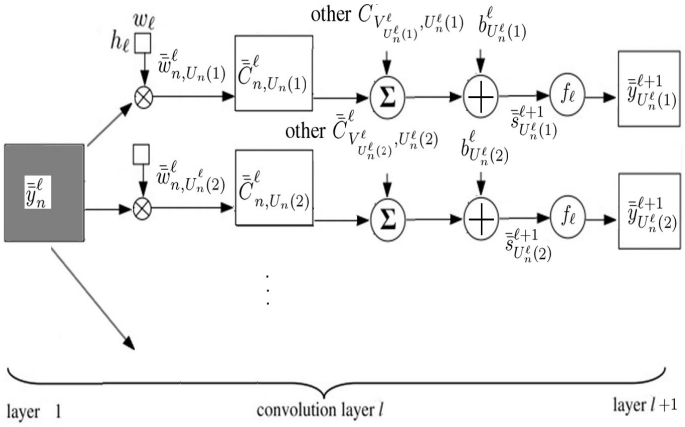

Fig. 6. Convolution layer one-multiple mapping (U_n^ℓ).

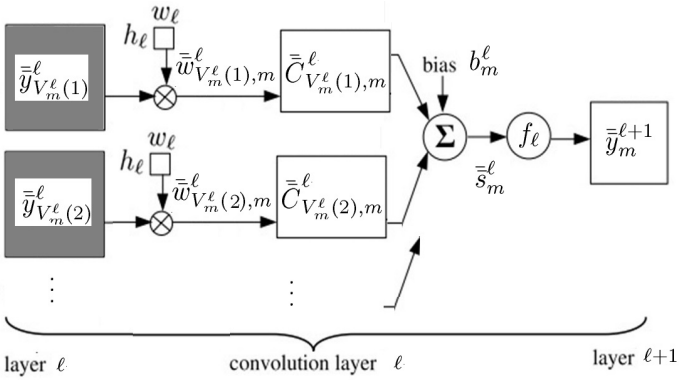

Fig. 7. Convolution layer multiple-one mapping (V_m^ℓ).

Fig.6 and Fig.7 show the schematic of convolution layer by exploiting the concept of U (one-multiple mapping) and V (multiple-one mapping).

Feature map n of convolution layer ℓ is calculated as

$$\bar{\bar{y}}_n^\ell = f_\ell\left(\sum_{m\in V_n^\ell} \bar{\bar{y}}_m^{\ell-1} \otimes \bar{\bar{w}}_{m,n}^\ell + b_n^\ell\right) \qquad (1)$$

$$= f_\ell\left(\sum_{m\in V_n^\ell} \bar{\bar{C}}_{m,n}^\ell + b_n^\ell\right) = f_\ell(\bar{\bar{s}}_n^\ell)$$

where

$$\bar{\bar{C}}_{m,n}^\ell = \bar{\bar{y}}_m^{\ell-1} \otimes \bar{\bar{w}}_{m,n}^\ell$$

where $\otimes$ is the 2D convolution operator.

The computation of convolution operation can be explicitly expressed as

$$\bar{\bar{C}}_{m,n}^\ell(i,j) = \sum_{i'=1}^{h_\ell}\sum_{j'=1}^{w_\ell} \bar{\bar{y}}_m^{\ell-1}(i'-1+i, j'-1+j) \times \bar{\bar{w}}_{m,n}^\ell(i',j')$$
$$1 \le i \le H_{\ell-1} - h_\ell + 1, \quad 1 \le j \le W_{\ell-1} - w_\ell + 1$$

Or

$$\bar{\bar{C}}_{m,n}^\ell(i,j) = \sum_{i'=i}^{h_\ell+i-1}\sum_{j'=j}^{w_\ell+j-1} \bar{\bar{y}}_m^{\ell-1}(i',j') \times \bar{\bar{w}}_{m,n}^\ell(i'-i+1, j'-j+1)$$
$$1 \le i \le H_\ell, \quad 1 \le j \le W_\ell$$
$$\rightarrow 1 \le i'-i+1 \le h_\ell, \quad 1 \le j'-j+1 \le w_\ell$$

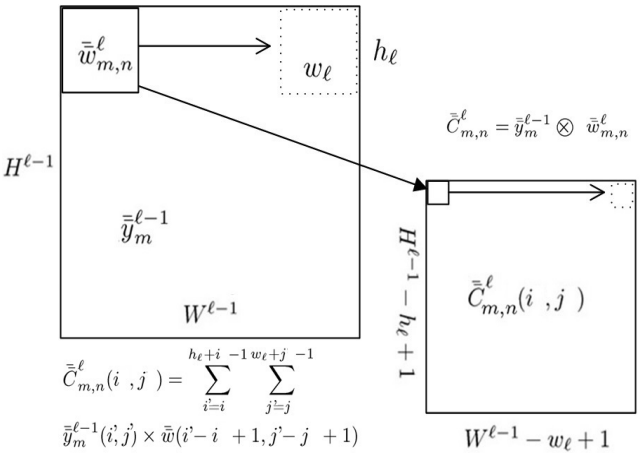

Fig. 8. Schematic of convolution operation, $\bar{\bar{C}}_{m,n}^\ell = \bar{\bar{y}}_m^{\ell-1} \otimes \bar{\bar{w}}_{m,n}^\ell$.

Fig.8 shows the schematic of convolution operation. If the size of input feature maps $\bar{\bar{y}}_m^{\ell-1}$ is $H_{\ell-1} \times W_{\ell-1}$ pixels and the size of convolution masks $\bar{\bar{w}}_{m,n}^\ell$ is $h_\ell \times w_\ell$, the size of output feature map $\bar{\bar{y}}_n^\ell$ is

$$H_\ell \times W_\ell = (H_{\ell-1} - h_\ell + 1) \times (W_{\ell-1} - w_\ell + 1)$$

For the last convolution layer, the output will become a scalar. From (1), the nth neuron for the last convolution layer $(L-1)$ can be expressed as

$$y_n^{L-1} = f_{L-1}(s_n^{L-1})$$
$$s_n^{L-1} = s_n^{L-1}(1,1) = b_n^{L-1} + \sum_{m\in V_n^{L-1}}\sum_{i'=1}^{h_{L-1}}\sum_{j'=1}^{w_{L-1}} y_m^{L-2}(i',j') \times w_{m,n}^{L-1}(i',j')$$

From (1), the nth neuron for the other convolutioh layer ($\ell = 2a+1, a = 0, 1, 2, \cdots$) can be expressed as

$$\bar{\bar{y}}_n^\ell = f_\ell(\bar{\bar{s}}_n^\ell)$$
$$s_n^\ell(i,j) = b_n^\ell + \sum_{m \in V_n^\ell} \sum_{i'=i}^{h_\ell+i-1} \sum_{j'=j}^{w_\ell+j-1} y_m^{\ell-1}(i',j') \times w_{m,n}^\ell(i'-i+1, j'-j+1)$$

B. Sub-sampling layer

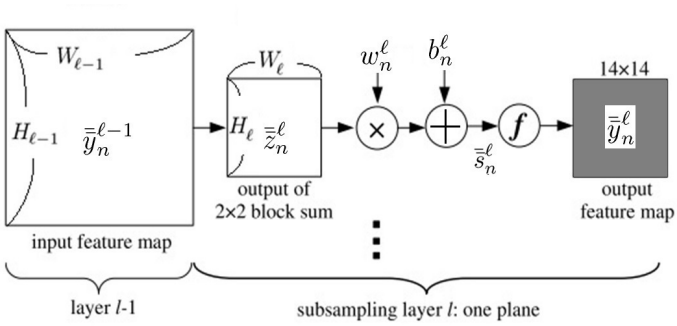

Fig. 9. A sub-sampling layer in a CNN, where $W^\ell = W^{\ell-1}/2$ and $H^\ell = H^{\ell-1}/2$.

Fig.9 shows a sub-sampling layer in a CNN. Considering the feature map n in a sub-sampling layer ℓ, $\ell = 2, 4, \cdots, 2a$, the feature map n of convolution layer $\ell - 1$ is divided into non-overlapping blocks of size 2×2 pixels. $\bar{\bar{z}}_n^\ell$ is a matrix, and its element is obtained by summing the four pixels in each block,

$$z_n^\ell(i,j) = y_n^{\ell-1}(2i-1, 2j-1) + y_n^{\ell-1}(2i-1, 2j) + y_n^{\ell-1}(2i, 2j-1) + y_n^{\ell-1}(2i, 2j) \quad (2)$$

The feature map n of sub-sampling layer ℓ is calculated as

$$\bar{\bar{y}}_n^\ell = f_\ell(\bar{\bar{z}}_n^\ell \times w_n^\ell + b_n^\ell) = f_\ell(\bar{\bar{s}}_n^\ell) \quad (3)$$

where w_n^ℓ and b_n^ℓ are the weight and bias term, respectively. The size of feature map $\bar{\bar{y}}_n^\ell$ in sub-sampling layer ℓ is

$$H_\ell = H_{\ell-1}/2, \quad W_\ell = W_{\ell-1}/2$$

C. Output Layer Fully Connected Layer

The output of sigmoidal neuron n is calculated as

$$y_n^L = f_L\left(\sum_{m=1}^{N_{L-1}} y_m^{L-1} w_{m,n}^L + b_n^L\right) \quad (4)$$
$$= f_L(s_n^L) \quad (5)$$

where N^L is the number of output sigmoidal neuron, $w_{m,n}^L$ denotes the weight from feature map m of the last convolutional layer, to neuron n of the output layer. b_n^L be the bias term associated with neuron of layer L. The outputs of all sigmoidal neurons form the network outputs:

$$\bar{y} = \left[y_1^L, y_2^L, \cdots, y_{N_L}^L\right]$$

III. Network Training

Training algorithm are devised to perform different visual recognition tasks. The objective of network training is to minimize an error function, which is defined in terms of the network actual outputs and the desired outputs.

Table IV summarizes the definitions used in CNN training, where $\bar{\bar{x}}^k$ is the kth training image, and $\bar{d}^k$ is the corresponding desired output vector. K is the number of input images in the training set. The error function $E(\bar{w})$ is defined as

$$E(\bar{w}) = \frac{1}{KN_L} \sum_{k=1}^{K} \sum_{n=1}^{N_L} (y_n^{L,k} - d_n^k)^2 \quad (6)$$

where $y_n^{L,k}$ is the nth neuron output of output layer for kth pattern., which is function of the network parameters.

There are two major approaches to CNN training. The first approach is known as online training, which updates network parameters after each training sample is presented. This approach requires less memory, but it is less stable because each training sample can push the network parameters along a new direction. The second approach is known as batch training, which updates network weights and biases after all training samples are presented. The batch training is focused in this work, because online training can be seen as a special case of batch training. An evaluation of network outputs for all training samples and an update of all network parameters are referred to collectively as a training epoch.

The error gradient, $\delta_{w_{m,n}}^\ell = \partial E/\partial w_{m,n}^\ell$ (for convolution layer), $\delta_{w_n}^\ell = \partial E/\partial w_n^\ell$ (for sub-sampling layer) and $\delta_{b_n}^\ell = \partial E/\partial b_n^\ell$ (for both convolution and sub-sampling layers) are computed through error sensitivities δ_n^ℓ. The error sensitivity of neuron (i,j) in feature map n of convolution layer or sub-sampling layer ℓ, or output layer is defined as

$$\delta_n^\ell(i,j) = \frac{\partial E}{\partial s_n^\ell(i,j)}, \quad (7)$$
$$\begin{cases} \ell = 1, 3, \cdots, 2a+1, & \text{for convolution layer} \\ \ell = 2, 4, \cdots, 2a, & \text{for sub-sampling layer} \\ \ell = L, & \text{for output layer} \end{cases}$$

A. Error Sensitivity Computation

1) Output Layer: Using the chain rule of differentiation, the error sensitivity of output layer can be computed from (7), (6) and (5), as

$$\delta_n^{L,k} = \frac{\partial E}{\partial s_n^{L,k}} = \frac{1}{K \times N_L} \times 2 \times (y_n^{L,k} - d_n^k) \times \frac{\partial y_n^{L,k}}{\partial s_n^{L,k}}$$
$$= \frac{2}{K \times N_L} e_n^k f_L'(s_n^{L,k}), \quad n = 1, 2, \cdots, N_L \quad (8)$$

where

$$e_n^k = y_n^{L,k} - d_n^k, \quad \frac{\partial y_n^{L,k}}{\partial s_n^{L,k}} = f_L'(s_n^{L,k})$$

The supporting material is

$$y_n^{L,k} = f_L(s_n^{L,k})$$

TABLE II
NOTATION FOR CNN TRAINING ALGORITHM

description	symbol	formula
training image index	k	$k = 1, 2, \cdots, K$
training image k	$\bar{\bar{x}}^k$	$\bar{\bar{x}} = \{x^k(i,j)\}, i = 1, \cdots, H_0; j = 1, \cdots, W_0$
desired output sample k	$\bar{d}^k$	$\bar{d}^k = (d_1^k, d_2^k, \cdots, d_{N_L}^k)^t$
network input or output of layer 0	$y^{0,k}(i,j)$	$y^{0,k} = (x^k(i,j)), i = 1, \cdots, H_0; j = 1, \cdots, W_0$
weighted sum input to neuron (i,j) in convolution layer ℓ, feature map n	$s_n^{\ell,k}(i,j)$	$n = 1, \cdots, N_\ell$; $i = 1, \cdots, H_\ell$; $j = 1, \cdots, W_\ell$
weighted sum input to neuron (i,j) in sub-sampling layer ℓ, feature map n	$s_n^{\ell,k}(i,j)$	$n = 1, \cdots, N_\ell$; $i = 1, \cdots, H_\ell$; $j = 1, \cdots, W_\ell$
output of neuron (i,j) in convolution layer ℓ, feature map n for input image k	$y_n^{\ell,k}(i,j)$	$y_n^{\ell,k}(i,j) = f_\ell(s_n^{\ell,k}(i,j))$
output of neuron (i,j) in sub-sampling layer ℓ, feature map n for input image k	$y_n^{\ell,k}(i,j)$	$y_n^{\ell,k}(i,j) = f_\ell(s_n^{\ell,k}(i,j))$
nth error for image k	e_n^k	$e_n^k = y_n^{L,k} - d_n^k$
error function	$E(\bar{w})$	$E(\bar{w}) = E(\bar{w}) = \frac{1}{N_L}\sum_{k=1}^{K}\sum_{n=1}^{N_L}(y_n^k - d_n^k)^2$
error sensitivity of pixel (i,j) in 2D layer ℓ	$\delta_{i,j}^{\ell,k}$	$\delta_{i,j}^{\ell,k} = \partial E/\partial s_{i,j}^{\ell,k}$
error sensitivity of neuron n in 1D layer ℓ	$\delta_n^{\ell,k}$	$\delta_n^{\ell,k} = \partial E/\partial s_n^{\ell,k}$

2) Last Convolution Layer: The error sensitivity of the last convolution layer $\ell = L-1$ can be expressed as

$$\begin{aligned}\delta_n^{L-1,k} &= \frac{\partial E}{\partial s_n^{L-1,k}} = \sum_{m=1}^{N_L}\frac{\partial E}{\partial s_m^{L,k}}\frac{\partial s_m^{L,k}}{\partial s_n^{L-1,k}} \\ &= \sum_{m=1}^{N_L}\frac{\partial E}{\partial s_m^{L,k}} \times \frac{\partial s_m^{L,k}}{\partial y_n^{L-1,k}}\frac{\partial y_n^{L-1,k}}{\partial s_n^{L-1,k}} \\ &= \sum_{m=1}^{N_L}\delta_m^{L,k} \times w_{n,m}^L \times f'_{L-1}(s_n^{L-1,k}) \\ &= f'_{L-1}(s_n^{L-1,k}) \times \sum_{m=1}^{N_L}\delta_m^{L,k} w_{n,m}^L \end{aligned} \quad (9)$$

where

$$\begin{aligned}\frac{\partial s_m^{L,k}}{\partial y_n^{L-1,k}} &= w_{n,m}^L \\ \frac{\partial y_n^{L-1,k}}{\partial s_n^{L-1,k}} &= f'_{L-1}(s_n^{L-1,k})\end{aligned}$$

some supporting equations are as following

$$\begin{aligned}s_m^{L,k} &= \sum_{n=1}^{N_{L-1}} y_n^{L-1,k} \times w_{n,m}^L + b_m^L \\ y_n^{L-1} &= f_{L-1}(s_n^{L-1})\end{aligned}$$

Note that the feature maps in the last convolution layer $\ell = L-1$ have a size of 1×1 pixel, $\delta_n^{L-1,k}$ is equivalent to $\delta_n^{L-1,k}(1,1)$ (scalar). Similarly,

$$s_n^{L-1} = s_n^{L-1,k}(1,1), \quad s_n^{L-1,k} = s_n^{L,k}(1,1), \quad y_n^{L-1,k} = y_n^{L-1}(1,1)$$

3) Last Sub-sampling Layer: Fig.10 shows the schematic of error sensitivity computation for last sub-sampling layer.

The error sensitivity of the last sub-sampling layer $\ell = L-2$

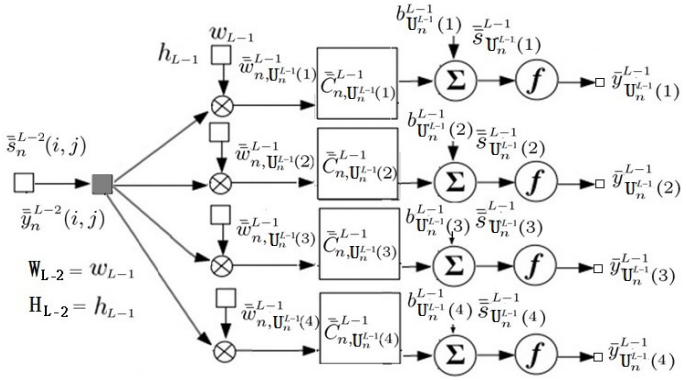

Fig. 10. Schematic of error sensitivity computation for last sub-sampling layer.

can be expressed as

$$\begin{aligned}\delta_n^{L-2,k}(i,j) &= \frac{\partial E}{\partial s_n^{L-2,k}(i,j)} \\ &= \sum_{m=1}^{N_{L-1}}\frac{\partial E}{\partial s_m^{L-1,k}} \times \frac{\partial s_m^{L-1,k}}{\partial s_n^{L-2,k}(i,j)} \\ &= \sum_{m=1}^{N_{L-1}}\delta_m^{L-1,k} \times \frac{\partial s_m^{L-1,k}}{\partial y_n^{L-2}(i',j')}\frac{\partial y_n^{L-2}(i',j')}{\partial s_n^{L-2,k}(i,j)} \\ &= \sum_{m\in U_n^{L-2}}\delta_m^{L-1,k} w_{n,m}^{L-1}(i,j) f'_{L-2}[s_n^{L-2,k}(i,j)] \\ &= f'_{L-2}[s_n^{L-2,k}(i,j)] \sum_{m\in U_n^{L-2}}\delta_m^{L-1,k} w_{n,m}^{L-1}(i,j)\end{aligned} \quad (10)$$

where $n = 1, 2, \cdots, N_{L-2}$; $i = 1, 2, \cdots, H_{L-2}$; $j = 1, 2, \cdots, W_{L-2}$, and

$$\begin{aligned}&\frac{\partial s_m^{L-1,k}}{\partial s_n^{L-2,k}(i,j)} = w_{n,m}^{L-1}(i,j) f'_{L-2}[s_n^{L-2,k}(i,j)], \\ &n \in V_m^{L-1} (\text{i.e.}, m \in U_n^{L-2})\end{aligned}$$

TABLE III
ILLUSTRATIVE MAPPING BETWEEN i_c AND i INDEX.

i	1	2	3	4	5	$\cdots$
i_c	1	1	2	2	3	$\cdots$

Because

$$\frac{\partial s_m^{L-1,k}}{\partial y_n^{L-2,k}(i,j)} = w_{n,m}^{L-1}(i,j), \quad n \in V_m^{L-1}$$
$$\frac{\partial y_n^{L-2,k}(i,j)}{\partial s_n^{L-2,k}(i,j)} = f'_{L-2}(s_n^{L-2,k}(i,j))$$

The supporting materials are

$$s_m^{L-1,k} = b_m^{L-1} + \sum_{n\in V_m^{L-1}} \sum_{i=1}^{h_{L-1}} \sum_{j=1}^{w_{L-1}} y_n^{L-2,k}(i,j) \times w_{n,m}^{L-1}(i,j)$$
$$= b_m^{L-1} + \sum_{n\in V_m^{L-1}} \sum_{i=1}^{h_{L-1}} \sum_{j=1}^{w_{L-1}} f_{L-2}[s_n^{L-2,k}(i,j)] \times w_{n,m}^{L-1}(i,j)$$

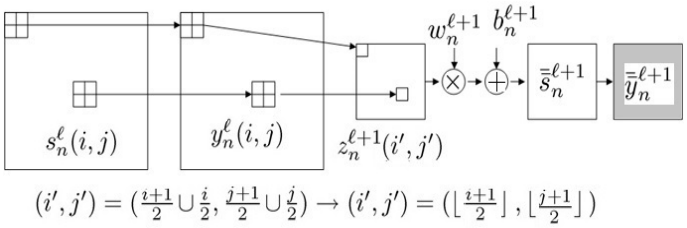

Fig. 11. Schematic in error sensitivity computation for other convolution layer.

4) Other Convolution Layer: Fig.11 shows the error sensitivity computation for other convolution layer, where the error sensitivity of other convolution layer ($\ell = 2a+1$) can be expressed as

$$\delta_n^{\ell,k}(i,j) = \frac{\partial E}{\partial s_n^{\ell,k}(i,j)}$$
$$= \sum_{i'=1,\cdots}^{H_{\ell+1}} \sum_{j'=1,\cdots}^{W_{\ell+1}} \frac{\partial E}{\partial s_n^{\ell+1,k}(i',j')} \times \frac{\partial s_n^{\ell+1,k}(i',j')}{\partial s_n^{\ell,k}(i,j)}$$
$$= f'_\ell[s_n^{\ell,k}(i,j)] \times \delta_n^{\ell+1,k}(i_c,j_c) \times w_n^{\ell+1} \quad (11)$$

where $i_c = \lfloor (i+1)/2 \rfloor$ and $j_c = \lfloor (j+1)/2 \rfloor$, and table.IV lists the illustrative mapping between i_c and i index.
$\frac{\partial s_m^{\ell+1}(i',j')}{\partial s_n^{\ell}(i,j)}$ is computed as

$$\frac{\partial s_n^{\ell+1,k}(i',j')}{\partial s_n^{\ell,k}(i,j)} = \frac{\partial s_n^{\ell+1,k}(i',j')}{\partial z_n^{\ell+1,k}(i',j')} \times \frac{\partial z_n^{\ell+1,k}(i',j')}{\partial s_n^{\ell,k}(i,j)}$$
$$= \begin{cases} w_n^{\ell+1} f_\ell(s_n^{\ell,k}(i,j)), & (i',j') = (\lfloor \frac{i+1}{2} \rfloor, \lfloor \frac{j+1}{2} \rfloor) \\ 0, & \text{otherwise} \end{cases}$$

and

$$\frac{\partial s_n^{\ell+1,k}(i',j')}{\partial z_n^{\ell+1,k}(i',j')} = w_n^{\ell+1}$$
$$\frac{\partial z_n^{\ell+1,k}(i',j')}{\partial s_n^{\ell,k}(i,j)} = f'_\ell(s_n^{\ell,k}(i,j)), \quad (i',j') = (\lfloor \frac{i+1}{2} \rfloor, \lfloor \frac{j+1}{2} \rfloor)$$

some suporting materials are as follows

$$\bar{\bar{s}}_n^{\ell+1,k}(i',j') = \bar{\bar{z}}_n^{\ell+1,k}(i',j') \times w_m^{\ell+1} + b_n^{\ell+1}$$
$$\frac{\partial z_n^{\ell+1,k}(i",j")}{\partial s_n^{\ell,k}(i,j)} = \frac{\partial y_n^{\ell,k}(2i"-1,2j"-1)}{\partial s_n^{\ell,k}(i,j)} + \frac{\partial y_n^{\ell,k}(2i"-1,2j")}{\partial s_n^{\ell,k}(i,j)}$$
$$+ \frac{\partial y_n^{\ell,k}(2i",2j"-1)}{\partial s_n^{\ell,k}(i,j)} + \frac{\partial y_n^{\ell,k}(2i",2j")}{\partial s_n^{\ell,k}(i,j)}$$
$$= \frac{\partial f_\ell(s_n^{\ell,k}(2i"-1,2j"-1))}{\partial s_n^{\ell,k}(i,j)} + \frac{\partial f_\ell(s_n^{\ell,k}(2i"-1,2j"))}{\partial s_n^{\ell,k}(i,j)}$$
$$+ \frac{\partial f_\ell(s_n^{\ell,k}(2i",2j"-1))}{\partial s_n^{\ell,k}(i,j)} + \frac{\partial f_{\ell,k}(s_n^{\ell,k}(2i",2j"))}{\partial s_n^{\ell,k}(i,j)}$$

Intrinsically,

$$\frac{\partial z_n^{\ell+1,k}(i',j')}{\partial s_n^{\ell,k}(i,j)} = \begin{cases} f'_\ell(s_n^{\ell,k}(i,j)), & [(2i'-1,2j'-1) = (i,j) \\ & \cup (2i'-1,2j') = (i,j) \\ & \cup (2i',2j'-1) = (i,j) \\ & \cup (2i',2j') = (i,j)] \\ 0, & \text{otherwise} \end{cases}$$

The supporting material are appended as

$$z_n^{\ell+1}(i,j) = y_n^{\ell}(2i-1,2j-1) + y_n^{\ell}(2i-1,2j)$$
$$+ y_n^{\ell}(2i,2j-1) + y_n^{\ell}(2i,2j)$$

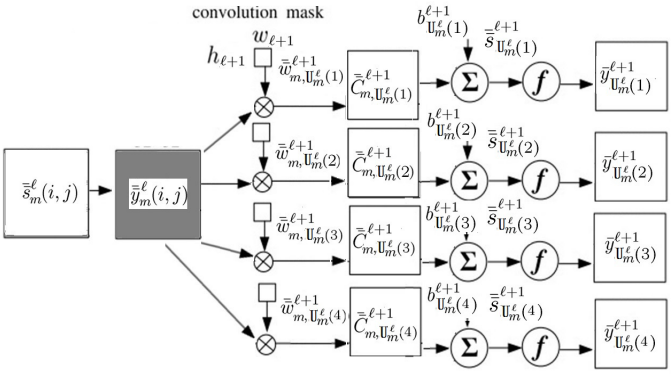

Fig. 12. Schematic in error sensitivity computation for other sub-sampling layer.

5) Other Sub-sampling Layer: Fig.12 shows the error sensitivity computation for other sub-sampling layer. The error sensitivity of other sub-sampling layer ($\ell = 2a$) can be expressed as

$$\delta_n^{\ell,k}(i,j) = \frac{\partial E}{\partial s_n^{\ell,k}(i,j)}$$
$$= \sum_{m=1}^{N_{\ell+1}} \sum_{i'=1}^{H_{\ell+1}} \sum_{j'=1}^{W_{\ell+1}} \frac{\partial E}{\partial s_m^{\ell+1,k}(i',j')} \times \frac{\partial s_m^{\ell+1,k}(i',j')}{\partial s_n^{\ell,k}(i,j)}$$
$$= f'_\ell[s_n^{\ell,k}(i,j)] \times \sum_{m\in U_n^\ell} \sum_{i'=1}^{H_{\ell+1}} \sum_{j'=1}^{W_{\ell+1}}$$
$$\delta_m^{\ell+1,k}(i',j') \times w_{n,m}^{\ell+1}(i-i'+1, j-j'+1) \qquad (12)$$

Noting

$$w_{n,m}^{\ell+1}(i-i'+1, j-j'+1) \neq 0,$$
$$1 \le i-i'+1 \le h_{\ell+1}, \quad 1 \le j-j'+1 \le w_{\ell+1}$$

where

$$\frac{\partial s_m^{\ell+1,k}(i',j')}{\partial s_n^{\ell,k}(i,j)} = \frac{\partial s_m^{\ell+1,k}(i',j')}{\partial y_n^{\ell,k}(i,j)} \frac{\partial y_n^{\ell,k}(i,j)}{\partial s_n^{\ell,k}(i,j)} =$$
$$\begin{cases} w_{n,m}^{\ell+1}(i-i'+1, j-j'+1), & n \in V_m^{\ell+1} (\text{i.e.}, m \in U_n^\ell) \\ \times f'_\ell[s_n^{\ell,k}(i,j)] & 1 \le i-i'+1 \le h_{\ell+1}, \\ & 1 \le j-j'+1 \le w_{\ell+1}, \\ 0, & \text{otherwise} \end{cases}$$

because

$$\frac{\partial s_m^{\ell+1,k}(i',j')}{\partial y_n^{\ell,k}(i,j)} = w_{n,m}^{\ell+1}(i-i'+1, j-j'+1)$$
$$\frac{\partial y_n^{\ell,k}(i,j)}{\partial s_n^{\ell,k}(i,j)} = f'_\ell(s_n^{\ell,k}(i,j))$$

The supporting material is

$$s_m^{\ell+1,k}(i',j') = b_m^{\ell+1} + \sum_{n\in V_m^{\ell+1}} \sum_{i=i'}^{h_{\ell+1}+i'-1} \sum_{j=j'}^{w_{\ell+1}+j'-1}$$
$$f_\ell[s_n^{\ell,k}(i,j)] \times w_{n,m}^{\ell+1,k}(i-i'+1, j-j'+1)$$

Noting that

$$w_{n,m}^{\ell+1}(i-i'+1, j-j'+1) = 0,$$
$$\text{if } i-i'+1 <= 0, \text{ or } i-i'+1 > h_{\ell+1},$$
$$\text{or } j-j'+1 <= 0, \text{ or } j-j'+1 > w_{\ell+1}$$
$$w_{n,m}^{\ell+1}(i-i'+1, j-j'+1) \neq 0,$$
$$1 \le i-i'+1 \le h_{\ell+1}, \quad 1 \le j-j'+1 \le w_{\ell+1}$$

Combining these relations, it is derived that

$$1 \le i-i'+1 \le h_{\ell+1} \rightarrow i-h_{\ell+1}+1 \le i' \le i$$
$$1 \le j-j'+1 \le w_{\ell+1} \rightarrow j-w_{\ell+1}+1 \le j' \le j$$

Therefore, (12) can be re expresed as

$$\delta_n^{\ell,k}(i,j) = f'_\ell[s_n^{\ell,k}(i,j)]$$
$$\times \sum_{m\in U_n^\ell} \sum_{i'=\max(1,i-h_{\ell+1}+1)}^{\min(H_{\ell+1},i)} \sum_{j'=\max(1,j-w_{\ell+1}+1)}^{\min(W_{\ell+1},j)}$$
$$\delta_m^{\ell+1,k}(i',j') \times w_{n,m}^{\ell+1}(i-i'+1, j-j'+1)$$

B. Error Gradient Computation

1) Output Layer, $\ell = L$:

$$\delta_{w_{m,n}}^L = \frac{\partial E}{\partial w_{m,n}^L} = \sum_{k=1}^K \frac{\partial E}{\partial s_n^{L,k}} \times \frac{\partial s_n^{L,k}}{\partial w_{m,n}^L}$$
$$= \sum_{k=1}^K \delta_n^{L,k} y_m^{L-1,k}$$

where $m = 1, 2, \cdots, N_{L-1}$ and $n = 1, 2, \cdots, N_L$. $\frac{\partial s_n^{L,k}}{\partial w_{m,n}^L}$ is computed as

$$\frac{\partial s_n^{L,k}}{\partial w_{m,n}^L} = y_m^{L-1,k}$$

The gradient of bias is computed as

$$\delta_{b_n}^L = \frac{\partial E}{\partial b_n^L} = \sum_{k=1}^K \delta_n^{L,k}$$

where $n = 1, 2, \cdots, N_L$.

2) Last Convolution Layer, $\ell = L-1$: The gradient computation of last convolution layer can be derived as

$$\delta_{w_{m,n}}^{L-1} = \frac{\partial E}{\partial w_{m,n}^{L-1}(i,j)} = \sum_{k=1}^K \frac{\partial E}{\partial s_n^{L-1,k}} \times \frac{\partial s_n^{L-1,k}}{\partial w_{m,n}^{L-1}(i,j)}$$
$$= \sum_{k=1}^K \delta_n^{L-1,k} y_m^{L-2,k}(i,j)$$

where $n = 1, 2, \cdots, N_{L-1}$. $\frac{\partial s_n^{L-1,k}}{\partial w_{m,n}^{L-1}(i,j)}$ is computed as

$$\frac{\partial s_n^{L-1,k}}{\partial w_{m,n}^{L-1}(i,j)} = \frac{\partial s_n^{L-1,k}(1,1)}{\partial w_{m,n}^{L-1}(i,j)} =$$
$$\begin{cases} y_m^{L-2,k}(i-1+1, j-1+1), & m \in V_n^{L-1} (n \in U_m^{L-2}) \\ 0, & \text{otherwise} \end{cases}$$

The supporting material equation is as follows,

$$s_n^{L-1,k}(1,1) = b_n^{L-1} + \sum_{m\in V_n^{L-1}} \sum_{i=1}^{h_{L-1}} \sum_{j=1}^{w_{L-1}}$$
$$y_m^{L-2,k}(i-1+1, j-1+1) \times w_{n,m}^{L-1}(i,j)$$

The gradient of bias is computed as

$$\delta_{b_n}^{L-1} = \frac{\partial E}{\partial b_n^{L-1}} = \sum_{k=1}^K \delta_n^{L-1,k}$$

where $n = 1, 2, \cdots, N_{L-1}$.

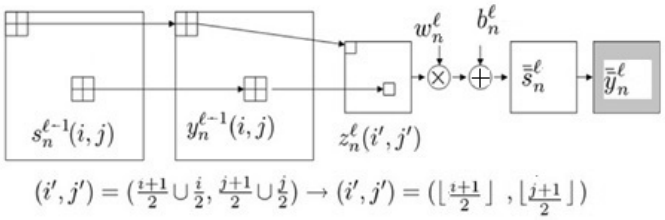

Fig. 13. Schematic of gradient computation for sub-sampling layer.

3) Sub-Sampling Layer, $\ell = 2a$: Fig.13 shows the schematic for gradient computation of sub-sampling layer. The gradient computation of sub-sampling layer can be derived as

$$\delta^{\ell}_{w_n} = \frac{\partial E}{\partial w^{\ell}_n} = \sum_{k=1}^{K}\sum_{i=1}^{H_\ell}\sum_{j=1}^{W_\ell} \frac{\partial E}{\partial s^{\ell,k}_n(i,j)} \frac{\partial s^{\ell,k}_n(i,j)}{\partial w^{\ell}_n}$$
$$= \sum_{k=1}^{K}\sum_{i=1}^{H_\ell}\sum_{j=1}^{W_\ell} \delta^{\ell,k}_n(i,j) z^{\ell,k}_n(i,j)$$

where $n = 1, 2, \cdots, N_\ell$.

$$\delta^{\ell}_{b_n} = \frac{\partial E}{\partial b^{\ell}_n} = \sum_{k=1}^{K}\sum_{i=1}^{H_\ell}\sum_{j=1}^{W_\ell} \delta^{\ell,k}_n(i,j)$$

where $n = 1, 2, \cdots, N_\ell$.

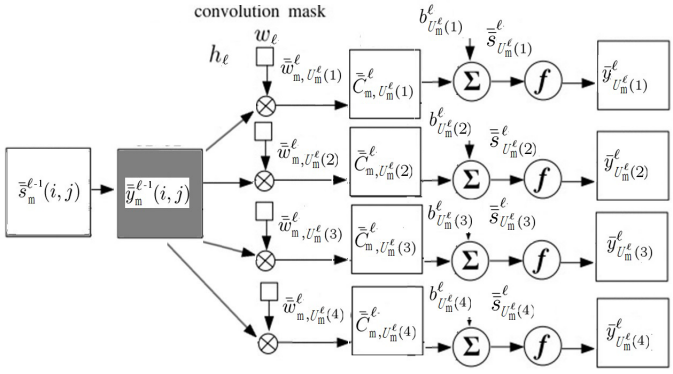

Fig. 14. Schematic of gradient computation for other convolution layer.

4) Other Convolution Layer, $\ell = 2a+1$: Fig.14 shows the schematic of gradient computation for other convolution layer.

The gradient computation of sub-sampling layer can be derived as

$$\delta^{\ell}_{w_{m,n}} = \frac{\partial E}{\partial w^{\ell}_{m,n}(i,j)} = \sum_{k=1}^{K}\sum_{i'=1}^{H_\ell}\sum_{j'=1}^{W_\ell} \frac{\partial E}{\partial s^{\ell,k}_n(i',j')} \times \frac{\partial s^{\ell,k}_n(i',j')}{\partial w^{\ell}_{m,n}(i,j)}$$
$$= \sum_{k=1}^{K}\sum_{i'=1}^{H_\ell}\sum_{j'=1}^{W_\ell} \delta^{\ell,k}_n(i',j') y^{\ell-1,k}_m(i-1+i', j-1+j') \qquad (13)$$

where $n = 1, 2, \cdots, N_\ell$. Note that,

$$1 \le i \le h_\ell, \quad 1 \le j \le w_\ell,$$
$$1 \le i' \le H_\ell, \quad 1 \le j' \le W_\ell,$$
$$\rightarrow 1 \le i - 1 + i' \le h_\ell + H_\ell - 1 = H_{\ell-1}$$
$$1 \le j - 1 + j' \le w_\ell + W_\ell - 1 = W_{\ell-1}$$

$\frac{\partial s^{\ell,k}_n(i',j')}{\partial w^{\ell}_{m,n}(i,j)}$ is computed as

$$\frac{\partial s^{\ell,k}_n(i',j')}{\partial w^{\ell}_{m,n}(i,j)} = \begin{cases} y^{\ell-1,k}_m(i-1+i', j-1+j'), & m \in V^{\ell}_n \\ 0, & \text{otherwise} \end{cases}$$

The supporting material is as follows,

$$s^{\ell,k}_n(i',j') = \sum_{m \in V^{\ell}_n} \sum_{i=1}^{h_\ell}\sum_{j=1}^{w_\ell}$$
$$y^{\ell-1,k}_m(i-1+i', j-1+j') \times w^{\ell}_{m,n}(i,j)$$

$$\delta^{\ell}_{b_n} = \frac{\partial E}{\partial b^{\ell}_n} = \sum_{k=1}^{K}\sum_{i=1}^{H_\ell}\sum_{j=1}^{W_\ell} \frac{\partial E}{\partial s^{\ell,k}_n(i,j)} \frac{\partial s^{\ell,k}_n(i,j)}{\partial b^{\ell}_n}$$
$$= \sum_{k=1}^{K}\sum_{i=1}^{H_\ell}\sum_{j=1}^{W_\ell} \delta^{\ell,k}_n(i,j)$$

where $n = 1, 2, \cdots, N_\ell$.

C. CNN Training Algorithm

Once the error gradient is derived, numerous optimization algorithm for minimizing E can be applied to train the network, which are gradient descent (GD), gradient descent with momentum and variable learning rate (GDMV), resilient back-propagation (RPROP), conjugate gradient (CG) and Levenberg-Marquardt (LM).

GD, GDMV and RPROP algorithms are first-order optimization methods.

CG algorithm can be considered as an intermediate between first- and second-order methods, whereas as the LM algorithm is trust-region method that uses the Gauss-Newton approximation of the Hessian matrix.

Table summarize the main characteristics of these algorithms.

Computation of the Jacobian matrix is similar to computation of the gradient ∇E.

However, the definition of error sensitivities is modified.

For the Jacobian matrix, error sensitivities are defined for each network error e^k_q, where $q = 1, \cdots, N_L$ (instead of the overall error function E).

The details of RPROP algorithm can found in [3].

TABLE IV
NOTATION FOR CNN TRAINING ALGORITHM

algorithm	description
GD	weights are updated along the negative gradient $\Delta\bar{\bar{w}}(t) = -\alpha\nabla E(t)$, where $\alpha > 0$ is a scalar learning rate.
GDMV	weight update is a linear combination of gradient and previous weight update, $\Delta\bar{\bar{w}}(t) = \lambda\Delta\bar{\bar{w}}(t-1) - (1-\lambda)\alpha(t)\nabla E(t)$, where $0 < \lambda < 1$ is momentum parameter and $\alpha(t)$ is adaptive scalar learning rate.
RPROP	weight update depends on the sign of gradient, $\Delta\bar{\bar{w}}_i(t) = -\mathrm{sign}\{\frac{\partial E}{\partial w_i}(t)\} \times \Delta_i(t)$, where $\Delta_i(t)$ is adaptive step specific to weight $\bar{\bar{w}}_i$, as $\Delta_i(t) = \begin{cases} \eta_{\mathrm{inc}}\Delta_i(t-1), & \frac{\partial E}{\partial \bar{\bar{w}}_i}(t)\frac{\partial E}{\partial \bar{\bar{w}}_i(t-1)} > 0 \\ \eta_{\mathrm{dec}}\Delta_i(t-1), & \frac{\partial E}{\partial \bar{\bar{w}}_i(t)}\frac{\partial E}{\partial \bar{\bar{w}}_i(t-1)} < 0 \\ \Delta_i(t-1), & \mathrm{otherwise} \end{cases}$ where $\eta_{\mathrm{inc}} > 1$, $0 < \eta_{\mathrm{dec}} < 1$ are scalar terms.
CG	weights updated along directions mutually conjugated with respect to Hessian matrix, $\Delta\bar{\bar{w}}(t) = \alpha(t)\bar{\bar{s}}(t)$, where $\bar{\bar{s}}$ is the search direction, defined as $\bar{\bar{s}}(t) = \begin{cases} -\nabla E(t), & t = 1(\mathrm{mod}\ P) \\ -\nabla E(t) + \beta(t)\bar{\bar{s}}(t-1) & \mathrm{otherwise} \end{cases}$ where $\alpha(t)$ is learning step derived with line search. $\beta(t)$ is updated according to the Polak-Ribiere formula, $\beta(t) = \frac{|\nabla E(t)T - \nabla E(t-1)|^t \nabla E(t)}{\|\nabla E(t-1)\|^2}$
LM	2nd-order Tylor expansion and Gauss-Newton approximation of Hessian matrix, $\Delta\bar{\bar{w}}(t) = -[\bar{\bar{J}}^t \cdot \bar{\bar{J}} + \mu\bar{\bar{I}}]^{-1} \cdot \Delta E$, where $\bar{\bar{J}}$ is Jacobian matrix defined as $J_{(q-1)K+k,i} = \frac{\partial e_q^k}{\partial w_i}$, $q = 1, \cdots, N_L \quad k = 1, \cdots, K \quad i = 1, \cdots, P$ ∇E is computed through the Jacobian matrix $\bar{\bar{J}}$, and μ is an adaptive parameter controlling the size of the trust region.

IV. Shared Weights of Convolution Layer From MLP Perspective

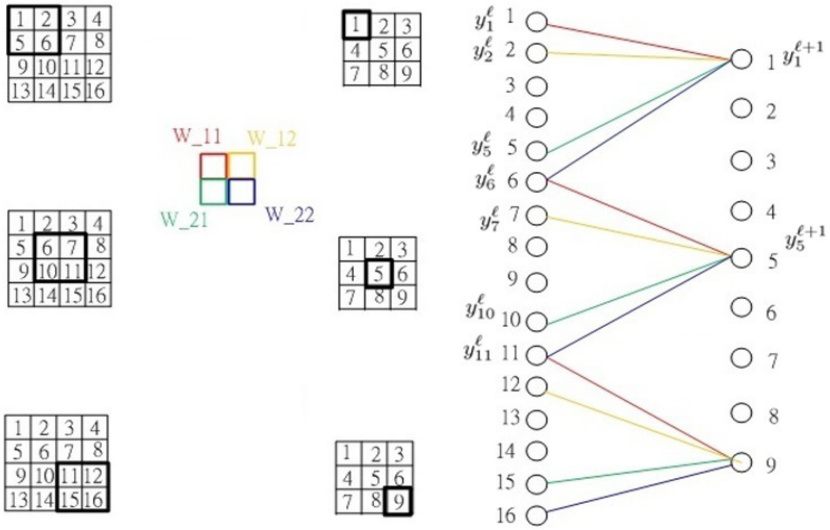

Fig. 15. Schematic of convolution layer from MLP perspective.

Fig.15 shows schematic of convolution layer from MLP perspective. The first and fifth outputs of layer $\ell+1$ can be expressed as

$$y_1^{\ell+1} = f(s_1^{\ell+1}) = f(y_1^\ell \times w_{11} + y_2^\ell \times w_{12} + y_5^\ell \times w_{21} + y_6^\ell \times w_{22})$$
$$y_5^{\ell+1} = f(y_6^\ell \times w_{11} + y_7^\ell \times w_{12} + y_{10}^\ell \times w_{21} + y_{11}^\ell \times w_{22})$$

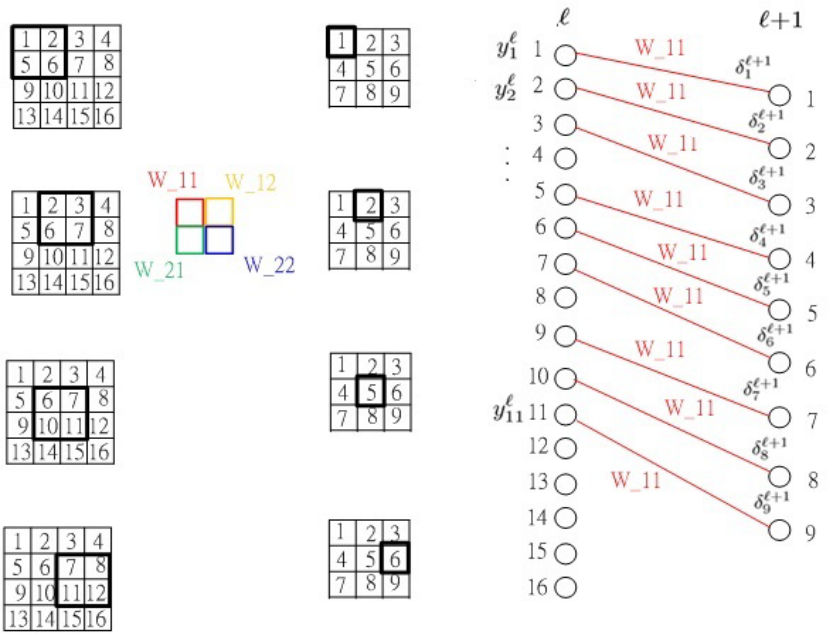

Fig. 16. Schematic of obtaining $\frac{\partial E}{\partial w_{1,1}}$ from $\frac{\partial E}{\partial s_n^{\ell+1}}$.

Fig.16 shows the computation schematic of $\frac{\partial E}{\partial w_{1,1}}$ from MLP perspective. The computation of filter gradient is expressed as

$$\frac{\partial E}{\partial w_{1,1}} = \sum_{n \in \text{using } w_{1,1}} \frac{\partial E}{\partial s_n^{\ell+1}} \frac{\partial s_n^{\ell+1}}{\partial w_{1,1}}$$
$$= \delta_1^{\ell+1} y_1^\ell + \delta_2^{\ell+1} y_2^\ell + \delta_3^{\ell+1} y_3^\ell + \delta_4^{\ell+1} y_5^\ell + \delta_5^{\ell+1} y_6^\ell + \delta_6^{\ell+1} y_7^\ell + \delta_7^{\ell+1} y_9^\ell + \delta_8^{\ell+1} y_{10}^\ell + \delta_9^{\ell+1} y_{11}^\ell$$

V. Illustrative Example of CNN with 6 layer

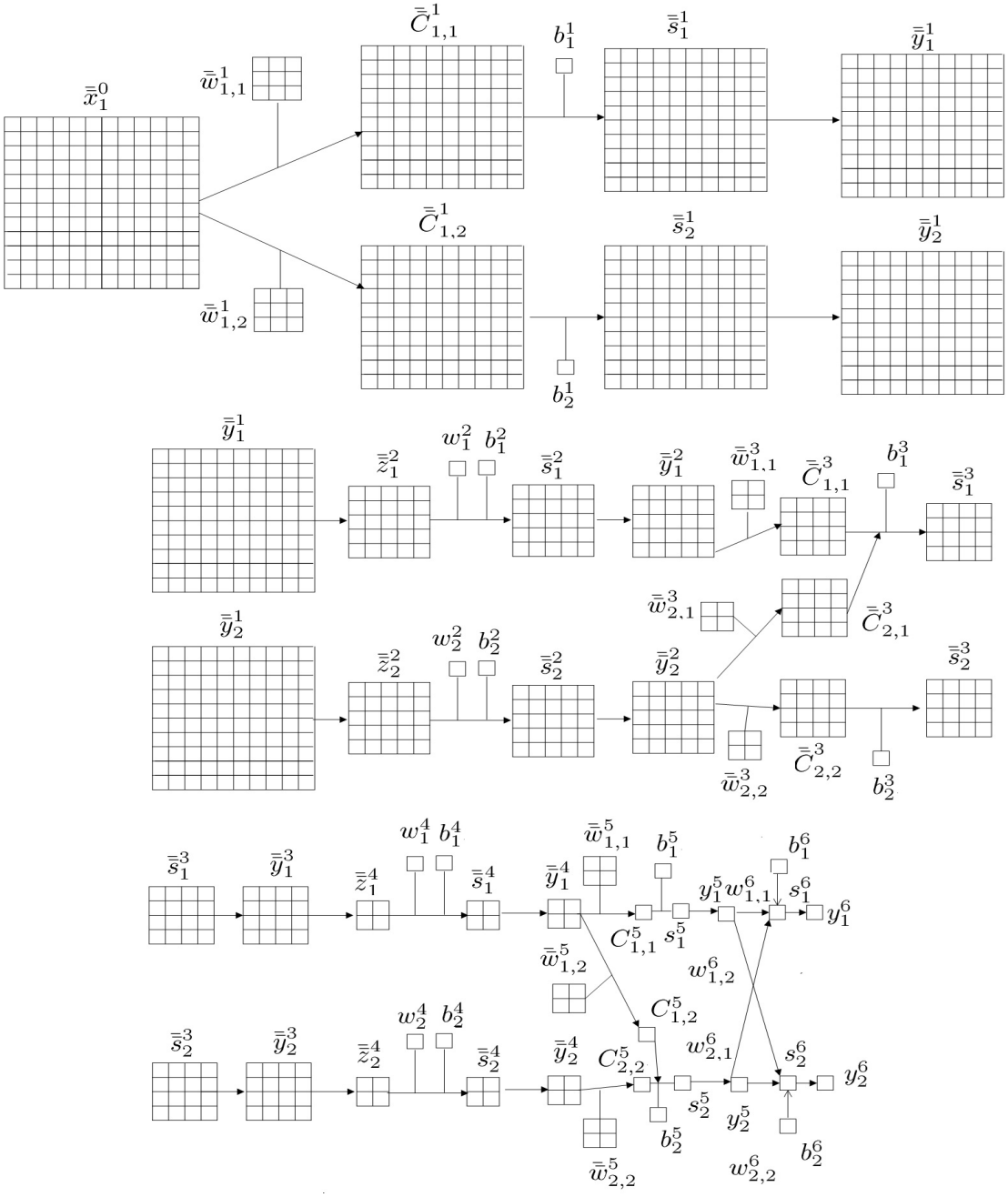

Fig. 17. Architecture of illustrative CNN example 2 with 6 layers.

Fig.17 shows the architecture of illustrative CNN example with 6 layers.

1) Forward Computation: From layer 0 to 1

$$\bar{\bar{y}}_1^1 = f_1\left(\bar{\bar{y}}^0 \otimes \ \bar{\bar{w}}_{1,1}^1 + b_1^1\right)$$
$$\bar{\bar{y}}_2^1 = f_1\left(\bar{\bar{y}}^0 \otimes \ \bar{\bar{w}}_{1,2}^1 + b_2^1\right)$$

where

$$\bar{\bar{C}}_{1,1}^1(i", j") = \sum_{i'=1}^{h_1} \sum_{j'=1}^{w_1} \bar{\bar{y}}^0(i'-1+i", j'-1+j") \times \bar{\bar{w}}_{1,1}^1(i', j')$$
$$1 \le i" \le H_0 - h_1 + 1, \quad 1 \le j" \le W_0 - w_1 + 1$$

$$= \sum_{i'=1}^{3} \sum_{j'=1}^{3} \bar{\bar{y}}^0(i'-1+i", j'-1+j") \times \bar{\bar{w}}^1_{1,1}(i', j')$$
$$1 \le i" \le 12-3+1 = 10, \quad 1 \le j" \le 12-3+1 = 10$$

similarly,

$$\bar{\bar{C}}^1_{1,2}(i", j") = \sum_{i'=1}^{3} \sum_{j'=1}^{3} \bar{\bar{y}}^0_1(i'-1+i", j'-1+j") \times \bar{\bar{w}}^1_{1,2}(i', j')$$
$$1 \le i" \le 12-3+1 = 4, \quad 1 \le j" \le 12-3+1 = 10$$

Feature map 1 and 2 of convolution layer 1 is calculated as

$$\bar{\bar{y}}^1_1 = f_1(\bar{\bar{s}}^1_1) = f_1\left(\bar{\bar{C}}^1_{1,1} + b^1_1\right) = f_1\left(\bar{\bar{y}}^0 \otimes \ \bar{\bar{w}}^1_{1,1} + b^1_1\right)$$
$$\bar{\bar{y}}^1_2 = f_1(\bar{\bar{s}}^1_2) = f_1\left(\bar{\bar{C}}^1_{1,2} + b^1_2\right) = f_1\left(\bar{\bar{y}}^0 \otimes \ \bar{\bar{w}}^1_{1,2} + b^1_2\right)$$

From layer 1 to layer 2

$$z^2_1(i,j) = y^1_1(2i-1, 2j-1) + y^1_1(2i-1, 2j) + y^1_1(2i, 2j-1) + y^1_1(2i, 2j)$$
$$z^2_2(i,j) = y^1_2(2i-1, 2j-1) + y^1_2(2i-1, 2j) + y^1_2(2i, 2j-1) + y^1_2(2i, 2j)$$

The first and second feature map of sub-sampling layer 2 is calculated as

$$\bar{\bar{y}}^2_1 = f_2(\bar{\bar{z}}^2_1 \times w^2_1 + b^2_1) = f_2(\bar{\bar{s}}^2_1), \quad \bar{\bar{y}}^2_2 = f_2(\bar{\bar{z}}^2_2 \times w^2_2 + b^2_2) = f_2(\bar{\bar{s}}^2_2)$$

where w^2_1 and b^2_1 are the weight and bias term, respectively. The size of feature map $\bar{\bar{y}}^2_1$ in sub-sampling layer 2 is

$$H_2 = H_1/2, \quad W_2 = W_1/2$$

From layer 2 to 3

$$\bar{\bar{y}}^3_1 = f_3\left(\sum_{m \in V^3_1} \bar{\bar{y}}^2_m \otimes \ \bar{\bar{w}}^3_{m,1} + b^3_1\right) = f_3(\bar{\bar{C}}^3_{1,1} + \bar{\bar{C}}^3_{2,1} + b^3_1)$$
$$= f_3\left(\bar{\bar{y}}^2_1 \otimes \ \bar{\bar{w}}^3_{1,1} + \bar{\bar{y}}^2_2 \otimes \ \bar{\bar{w}}^3_{2,1} + b^3_1\right)$$
$$\bar{\bar{y}}^3_2 = f_3\left(\sum_{m \in V^3_2} \bar{\bar{y}}^2_m \otimes \ \bar{\bar{w}}^3_{m,2} + b^3_2\right) = f_3(\bar{\bar{C}}^3_{2,2} + b^3_2)$$
$$= f_3\left(\bar{\bar{y}}^3_2 \otimes \ \bar{\bar{w}}^1_{2,2} + b^3_2\right)$$

where

$$\bar{\bar{C}}^3_{1,1}(i", j") = \sum_{i'=1}^{h_3} \sum_{j'=1}^{w_3} \bar{\bar{y}}^2_1(i'-1+i", j'-1+j") \times \bar{\bar{w}}^3_{1,1}(i', j')$$
$$1 \le i" \le H_2 - h_3 + 1, \quad 1 \le j" \le W_2 - w_3 + 1$$
$$= \sum_{i'=1}^{2} \sum_{j'=1}^{2} \bar{\bar{y}}^2_1(i'-1+i", j'-1+j") \times \bar{\bar{w}}^3_{1,1}(i', j')$$
$$1 \le i" \le 5-2+1 = 4, \quad 1 \le j" \le 5-2+1 = 4$$

similarly,

$$\bar{\bar{C}}^3_{2,1}(i", j") = \sum_{i'=1}^{2} \sum_{j'=1}^{2} \bar{\bar{y}}^2_2(i'-1+i", j'-1+j") \times \bar{\bar{w}}^3_{2,1}(i', j')$$
$$\bar{\bar{C}}^3_{2,2}(i", j") = \sum_{i'=1}^{2} \sum_{j'=1}^{2} \bar{\bar{y}}^2_2(i'-1+i", j'-1+j") \times \bar{\bar{w}}^3_{2,2}(i', j')$$
$$1 \le i" \le 4, \quad 1 \le j" \le 4$$

From layer 3 to 4

$$z_1^4(i,j) = y_1^3(2i-1,2j-1) + y_1^3(2i-1,2j) + y_1^3(2i,2j-1) + y_1^3(2i,2j)$$
$$z_2^4(i,j) = y_2^3(2i-1,2j-1) + y_1^3(2i-1,2j) + y_2^3(2i,2j-1) + y_2^3(2i,2j)$$

The first and second feature map of sub-sampling layer 2 is calculated as

$$\bar{\bar{y}}_1^4 = f_2(\bar{\bar{s}}_1^4) = f_2(\bar{\bar{z}}_1^4 \times w_1^4 + b_1^4)$$
$$\bar{\bar{y}}_2^4 = f_2(\bar{\bar{s}}_2^4) = f_2(\bar{\bar{z}}_1^4 \times w_2^4 + b_2^4)$$

where w_1^4 and b_1^4 are the weight and bias term, respectively. The size of feature map $\bar{\bar{y}}_1^4$ in sub-sampling layer 2 is

$$H_4 = H_3/2, \quad W_4 = W_3/2$$

From layer 4 to 5

$$\bar{\bar{y}}_1^5 = f_5(\bar{\bar{s}}_1^5) = f_5\left(\sum_{m\in V_1^5} \bar{\bar{y}}_m^4 \otimes \bar{\bar{w}}_{m,1}^5 + b_1^5\right) = f_3(\bar{\bar{C}}_{1,1}^5 + + b_1^5)$$
$$= f_3\left(\bar{\bar{y}}_1^4 \otimes \bar{\bar{w}}_{1,1}^5 + b_1^5\right)$$
$$\bar{\bar{y}}_2^5 = f_5(\bar{\bar{s}}_2^5) = f_5\left(\sum_{m\in V_2^5} \bar{\bar{y}}_m^4 \otimes \bar{\bar{w}}_{m,2}^5 + b_2^5\right) = f_5(\bar{\bar{C}}_{1,2}^5 + \bar{\bar{C}}_{2,2}^5 + b_2^5)$$
$$= f_5\left(\bar{\bar{y}}_1^5 \otimes \bar{\bar{w}}_{1,2}^5 + \bar{\bar{y}}_2^5 \otimes \bar{\bar{w}}_{2,2}^5 + b_2^5\right)$$

where

$$\bar{\bar{C}}_{1,1}^5(i",j") = \sum_{i'=1}^{h_5}\sum_{j'=1}^{w_5} \bar{\bar{y}}_1^4(i'-1+i",j'-1+j") \times \bar{\bar{w}}_{1,1}^5(i',j')$$
$$1 \le i" \le H_4 - h_5 + 1, \quad 1 \le j" \le W_4 - w_5 + 1$$
$$= \sum_{i'=1}^{2}\sum_{j'=1}^{2} \bar{\bar{y}}_1^4(i'-1+i",j'-1+j") \times \bar{\bar{w}}_{1,1}^5(i',j')$$
$$1 \le i" \le 2-2+1 = 1, \quad 1 \le j" \le 2-2+1 = 1$$

similarly,

$$\bar{\bar{C}}_{1,2}^5(i",j") = \sum_{i'=1}^{2}\sum_{j'=1}^{2} \bar{\bar{y}}_1^4(i'-1+i",j'-1+j") \times \bar{\bar{w}}_{1,2}^5(i',j')$$
$$\bar{\bar{C}}_{2,2}^5(i",j") = \sum_{i'=1}^{2}\sum_{j'=1}^{2} \bar{\bar{y}}_2^4(i'-1+i",j'-1+j") \times \bar{\bar{w}}_{2,2}^5(i',j')$$
$$1 \le i" \le 1, \quad 1 \le j" \le 1$$

From layer 5 to 6,

$$y_1^6 = f_6\left(\sum_{m=1}^{2} y_m^5 w_{m,1}^6 + b_1^6\right)$$
$$y_2^6 = f_6\left(\sum_{m=1}^{2} y_m^5 w_{m,2}^6 + b_2^6\right)$$

2) Backward Computation: Layer 6, the formula is re-written as

$$\delta_n^6 = \frac{1}{2} \times 2 \times (y_n^6 - d_n) \times f_6'(s_n^6), \quad \frac{\partial E}{\partial w_{m,n}^6} = \delta_n^6 y_m^5, \quad \frac{\partial E}{\partial b_n^6} = \delta_n^6$$

Therefore,

$$\delta_1^6 = \frac{\partial E}{\partial s_1^6} = \frac{1}{N_L} \times 2 \times (y_1^6 - d_1) \times \frac{\partial y_1^6}{\partial s_1^6} = \frac{2}{N_L}(y_1^6 - d_1) f_6'(s_1^6)$$
$$\delta_2^6 = \frac{\partial E}{\partial s_2^6} = \frac{1}{N_L} \times 2 \times (y_2^6 - d_2) \times \frac{\partial y_2^6}{\partial s_2^6} = \frac{2}{N_L}(y_2^6 - d_2) f_6'(s_2^6),$$

$$\frac{\partial E}{\partial w_{1,1}^6} = \delta_1^6 y_1^5, \quad \frac{\partial E}{\partial w_{1,2}^6} = \delta_2^6 y_1^5, \quad \frac{\partial E}{\partial w_{2,1}^6} = \delta_1^6 y_2^5, \quad \frac{\partial E}{\partial w_{2,2}^6} = \delta_2^6 y_2^5$$

$$\frac{\partial E}{\partial b_1^6} = \delta_1^6, \quad \frac{\partial E}{\partial b_2^6} = \delta_2^6$$

Layer 5, the formula is re-written as

$$\delta_n^5 = f_5'(s_n^5) \times \sum_{m=1}^{N_6} \delta_m^6 \times w_{n,m}^6, \quad \frac{\partial E}{\partial w_{m,n}^5(i,j)} = \delta_n^5(1,1) y_m^4(i,j), \quad \frac{\partial E}{\partial b_n^5} = \delta_n^5 = \delta_n^5(1,1)$$

Therefore,

$$\delta_1^5 = f_5'(s_1^5) \times \sum_{m=1}^{2} \delta_m^6 w_{1,m}^6, \quad \delta_2^5 = f_5'(s_2^5) \times \sum_{m=1}^{2} \delta_m^6 w_{2,m}^6, \quad \frac{\partial E}{\partial b_n^5} = \delta_n^5(1,1)$$

$$\frac{\partial E}{\partial w_{1,1}^5(i,j)} = \delta_1^5 \times y_1^4(i,j), \quad \frac{\partial E}{\partial w_{1,2}^5(i,j)} = \delta_2^5 \times y_1^4(i,j), \quad \frac{\partial E}{\partial w_{2,2}^5(i,j)} = \delta_2^5 \times y_2^4(i,j)$$

$$\frac{\partial E}{\partial b_1^5} = \delta_1^5, \quad \frac{\partial E}{\partial b_2^5} = \delta_2^5$$

Layer 4, the formula is re-written as

$$\delta_n^4(i,j) = f_4'[s_n^4(i,j)] \sum_{m \in U_n^4} \delta_m^5(1,1) w_{n,m}^5(i,j), \quad \frac{\partial E}{\partial w_n^4} = \sum_{i=1}^{H_4} \sum_{j=1}^{W_4} \delta_n^4(i,j) z_n^4(i,j), \quad \frac{\partial E}{\partial b_n^4} = \sum_{i=1}^{H_4} \sum_{j=1}^{W_4} \delta_n^4(i,j)$$

Therefore,

$$\delta_1^4(i,j) = f_4'[s_1^4(i,j)] \sum_{m \in U_1^4} \delta_m^5(1,1) w_{1,m}^5(i,j) = f_4'[s_1^4(i,j)] \times [\delta_1^5(1,1) w_{1,1}^5(i,j) + \delta_2^5(1,1) w_{1,2}^5(i,j)]$$
$$\delta_2^4(i,j) = f_4'[s_2^4(i,j)] \sum_{m \in U_2^4} \delta_m^5(1,1) w_{2,m}^5(i,j) = f_4'[s_2^4(i,j)] \times \delta_2^5(1,1) w_{2,2}^5(i,j)$$

and

$$\frac{\partial E}{\partial w_1^4} = \sum_{i=1}^{H_4} \sum_{j=1}^{W_4} \delta_1^4(i,j) z_1^4(i,j), \quad \frac{\partial E}{\partial w_2^4} = \sum_{i=1}^{H_4} \sum_{j=1}^{W_4} \delta_2^4(i,j) z_2^4(i,j)$$
$$\frac{\partial E}{\partial b_1^4} = \sum_{i=1}^{H_4} \sum_{j=1}^{W_4} \delta_1^4(i,j), \quad \frac{\partial E}{\partial b_2^4} = \sum_{i=1}^{H_4} \sum_{j=1}^{W_4} \delta_2^4(i,j)$$

Layer 3, the formula are re-written as

$$\delta_n^3(i,j) = f_3'[s_n^3(i,j)] \times \delta_n^4(i_c, j_c) \times w_n^4, \quad \frac{\partial E}{\partial w_{m,n}^3(i,j)} = \sum_{i'=1}^{H_3} \sum_{j'=1}^{W_3} \delta_n^3(i',j') y_m^2(i-1+i', j-1+j'), \quad \frac{\partial E}{\partial b_n^3} = \sum_{i=1}^{H_3} \sum_{j=1}^{W_3} \delta_n^3(i,j)$$

where $i_c = \lfloor \frac{i+1}{2} \rfloor$ and $j_c = \lfloor \frac{j+1}{2} \rfloor$.

Therefore

$$\delta_1^3(i,j) = f_3'[s_1^3(i,j)] \times \delta_1^4(i_c, j_c) \times w_1^4, \quad \delta_2^3(i,j) = f_3'[s_2^3(i,j)] \times \delta_2^4(i_c, j_c) \times w_2^4$$

where $i_c = \lfloor \frac{i+1}{2} \rfloor$ and $j_c = \lfloor \frac{j+1}{2} \rfloor$.

$$\frac{\partial E}{\partial w_{1,1}^3(i,j)} = \sum_{i'=1}^{H_3} \sum_{j'=1}^{W_3} \delta_1^3(i',j') y_1^2(i-1+i', j-1+j'), \quad \frac{\partial E}{\partial w_{2,1}^3(i,j)} = \sum_{i'=1}^{H_3} \sum_{j'=1}^{W_3} \delta_1^3(i',j') y_2^2(i-1+i', j-1+j')$$
$$\frac{\partial E}{\partial w_{2,2}^3(i,j)} = \sum_{i'=1}^{H_3} \sum_{j'=1}^{W_3} \delta_2^3(i',j') y_2^2(i-1+i', j-1+j')$$

$$\frac{\partial E}{\partial b_1^3}=\sum_{i=1}^{H_3}\sum_{j=1}^{W_3}\delta_1^3(i,j),\quad \frac{\partial E}{\partial b_2^3}=\sum_{i=1}^{H_3}\sum_{j=1}^{W_3}\delta_2^3(i,j)$$

Layer 2, the formula is re-written as

$$\delta_n^{\ell,k}(i,j)=f_\ell'[s_n^{\ell,k}(i,j)]\times\sum_{m\in U_n^\ell}\sum_{i'=1}^{H_{\ell+1}}\sum_{j'=1}^{W_{\ell+1}}\delta_m^{\ell+1,k}(i',j')\times w_{n,m}^{\ell+1,k}(i-i'+1,j-j'+1)$$

$$\frac{\partial E}{\partial w_n^2}=\sum_{i=1}^{H_2}\sum_{j=1}^{W_2}\delta_n^2(i,j)z_n^2(i,j),\quad \frac{\partial E}{\partial b_n^2}=\sum_{i=1}^{H_2}\sum_{j=1}^{W_2}\delta_n^2(i,j)$$

Noting

$$w_{n,m}^{\ell+1}(i-i'+1,j-j'+1)\neq 0,$$
$$1\le i-i'+1\le h_{\ell+1},\quad 1\le j-j'+1\le w_{\ell+1}$$

Therefore,

$$\delta_1^2(i,j)=f_2'[s_1^2(i,j)]\times\sum_{m\in U_1^2}\sum_{i'=1}^{H_3}\sum_{j'=1}^{W_3}\delta_m^3(i',j')\times w_{1,m}^3(i-i'+1,j-j'+1)$$
$$=\sum_{i'=1}^{H_3}\sum_{j'=1}^{W_3}\delta_1^3(i',j')\times w_{1,1}^3(i-i'+1,j-j'+1)$$
$$\delta_2^2(i,j)=f_2'[s_2^2(i,j)]\times\sum_{m\in U_2^2}\sum_{i'=i}^{h_3+i-1}\sum_{j'=j}^{w_3+j-1}\delta_m^3(i',j')\times w_{2,m}^3(i-i'+1,j-j'+1)$$
$$=\sum_{i'=1}^{H_3}\sum_{j'=1}^{W_3}\left[\delta_1^3(i',j')\times w_{2,1}^3(i-i'+1,j-j'+1)+\delta_2^3(i',j')\times w_{2,2}^3(i-i'+1,j-j'+1)\right]$$

Noting

$$w_{n,m}^3(i-i'+1,j-j'+1)\neq 0,$$
$$1\le i-i'+1\le h_3,\quad 1\le j-j'+1\le w_3$$

and

$$\frac{\partial E}{\partial w_1^2}=\sum_{i=1}^{H_2}\sum_{j=1}^{W_2}\delta_1^2(i,j)z_1^2(i,j),\quad \frac{\partial E}{\partial w_2^2}=\sum_{i=1}^{H_2}\sum_{j=1}^{W_2}\left[\delta_2^2(i,j)z_2^2(i,j)\right]$$

$$\frac{\partial E}{\partial b_1^2}=\sum_{i=1}^{H_2}\sum_{j=1}^{W_2}\delta_1^2(i,j),\quad \frac{\partial E}{\partial b_2^2}=\sum_{i=1}^{H_2}\sum_{j=1}^{W_2}\delta_2^2(i,j)$$

Layer 1 the formula are re-written as

$$\delta_n^1(i,j)=f_1'[s_n^1(i,j)]\times\delta_n^2(i_c,j_c)\times w_n^2,\quad \frac{\partial E}{\partial w_{m,n}^1(i,j)}=\sum_{i'=1}^{H_1}\sum_{j'=1}^{W_1}\delta_n^1(i',j')y_m^0(i-1+i',j-1+j'),\quad \frac{\partial E}{\partial b_n^1}=\sum_{i=1}^{H_1}\sum_{j=1}^{W_1}\delta_n^1(i,j)$$

where $i_c=\lfloor\frac{i+1}{2}\rfloor$ and $j_c=\lfloor\frac{j+1}{2}\rfloor$.

Therefore,

$$\delta_1^1(i,j)=f_1'[s_1^1(i,j)]\times\delta_1^2(i_c,j_c)\times w_1^2,\quad \delta_2^1(i,j)=f_1'[s_2^1(i,j)]\times\delta_2^2(i_c,j_c)\times w_2^2$$

where $i_c=\lfloor\frac{i+1}{2}\rfloor$ and $j_c=\lfloor\frac{j+1}{2}\rfloor$.

$$\frac{\partial E}{\partial w_{1,1}^1(i,j)}=\sum_{i'=1}^{H_1}\sum_{j'=1}^{W_1}\delta_1^1(i',j')\bar{\bar{y}}_1^0(i-1+i',j-1+j'),\quad \frac{\partial E}{\partial w_{1,2}^1(i,j)}=\sum_{i'=1}^{H_1}\sum_{j'=1}^{W_1}\delta_2^1(i',j')\bar{\bar{y}}_1^0(i-1+i',j-1+j')$$

$$\frac{\partial E}{\partial b_1^1} = \sum_{i=1}^{H_1}\sum_{j=1}^{W_1}\delta_1^1(i,j), \quad \frac{\partial E}{\partial b_2^1} = \sum_{i=1}^{H_1}\sum_{j=1}^{W_1}\delta_2^1(i,j)$$

REFERENCES

[1] Y. LeCun, L. Bottou, Y. Bengio, P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol.86, issue 11, pp.2278-2324, 1998.

[2] S. L. Phung and A. Bouzerdoum, A pyramidal neural network for visual pattern recognition, *IEEE Transactions on Neural Networks*, vol. 27, no. 1, pp. 329V343, 2007.

[3] M. Riedmiller and H. Braun, A direct adaptive method of faster backpropagation learning: The rprop algorithm, in IEEE International Conference on Neural Networks, San Francisco, 1993, pp. 586V591.