HHCV COURSE
of
Convolutional Neural Networks

孟朝晖

河海大学 **HHCV**

Computer Vision Group of Hohai University

Contents

Data Blocks

Abstract

多层卷积神经网络系统（CNN），数据结构建立在基于张量的数据块（tensor）基础上，数据处理过程由几个标准模块（process block）组合而成，主要包括 6 种：Conv, BN, ReLU, Pooling, Full, Softmax，这几个数据处理模块，均是双向运算的，前向激活值 transformation，反向敏感值 propagation，其中 Conv, BN, Full，含有权值学习运算，合计有 15 个主要运算函数。

数据处理模块的输入、输出、权值及辅助数据均以 tensor 形式表达。对于前向激活值运算，其输入输出均为激活值数据块（activation tensor），对于反向敏感值运算，其输入输出均为敏感值数据块（sensitivity tensor），其中 Conv, BN, Full, 几个 process block 需要有权值数据块（weight tensor）的参与，Pooling 有溯源标记数据块（path tensor）的参与。

Tensor

在数据科学中，张量（tensor）就是多维矩阵，用来表示各种维度的数据，比如 0 维（scalar）、1 维（vector）、2 维（matrix）、3 维（2d-spatio & 1d-feature channel）、4 维（2d-spatio & 1d-feature channel & 1d-tempral/1d-sample），等等。

数据块

一个由二维特征图叠合而成的数据块（a stack of feature maps）视为一个三维阵（3d-tensor、cuboid、volume），是深度网络的基本数据单位，表示为

$$X_{\times H,\times W}^{(\times C)} = \left\{ x_{i,j}^{(k)} \right\} \Big|_{\substack{i=0,\cdots,H-1 \\ j=0,\cdots,W-1 \\ k=0\cdots,C-1}}$$

$x_{i,j}^{(k)}$ 为三维阵的数值（scalar）分量，其中，H, W, C 分别表示 feature map 的高 height、宽 width，及 feature maps 的个数即通道数 channels，也是 filter（kernel）的个数。为了方便数据结构设计，游标都从 0 启始。为了与直观相符合，$x_{i,j}^{(k)}$ 的左下标表示高度变化（row），右下标表示宽度变化（col）。

比如一帧输入视频其宽高比为 640×480，具有 3 通道颜色值（bgr in OpenCV），则对应一个 cuboid，可以表示为 $X_{\times 480,\times 640}^{(\times 3)}$。

考虑到视频帧的时间参数及网络的层次，将上述三维阵扩展表示为更一般的形式：

$$X_{\times H_l,\times W_l}^{(l),(t),(\times C_l)} = \left\{x_{i,j}^{(l),(t),(k)}\right\}\Big|_{\substack{i=0,\cdots,H_l-1\\ j=0,\cdots,W_l-1\\ k=0,\cdots,C_l-1}} \qquad \begin{cases} l=1,\cdots,L & layer \\ t=0,\cdots,T-1 & time \end{cases}$$

其中，L 表示最大层次，l 表示当前层次，$T-1$ 表示最大时间，t 表示当前时间，不同层次的数据块 cuboid 其尺寸 $\{H_l, W_l, C_l\}$ 可以不同。以下为了公式表述方便，假设宽高尺寸是相同的而通道数不同，$\{H, W, C_l\}$。

一个特征图 feature map 是一个二维矩阵，表示某一种特征在二维场景（原始帧）中的分布情况，其分量值表示某种特征 (k) 在与原始帧相对应的位置上的显现度 intensity（符合度），可以表示为

$$X_{\times H,\times W}^{(k)} = \left\{x_{i,j}^{(k)}\right\}\Big|_{\substack{i=0,\cdots,H-1\\ j=0,\cdots,W-1}} \qquad k=0,\cdots,C-1 \quad filter$$

或者加上层次和时间

$$X_{\times H,\times W}^{(l),(t),(k)} = \left\{x_{i,j}^{(l),(t),(k)}\right\}\Big|_{\substack{i=0,\cdots,H-1\\ j=0,\cdots,W-1}} \qquad \begin{cases} l=1,\cdots,L & layer \\ t=0,\cdots,T-1 & time \\ k=0,\cdots,C_l-1 & filter \end{cases}$$

特征图不一定与原始帧的尺寸一致，越往后的网络层次，特征图的尺寸越小，精度越低，位置越模糊，其表达的特征越宏观抽象。

mini-batch 数据块

对于以 mini-batch 为训练模式的网络，一次输入数据为若干个图片，用 $t=0,\cdots,T-1$ 对图片进行编码，仍然用 $l=1,\cdots,L$ 对网络的层次进行编码，其它与前述一致，适用于 mini-batch 模式的数据块的一般形式为：

$$X_{\times H_l,\times W_l}^{(l),(\times T),(\times C_l)} = \left\{x_{i,j}^{(l),(t),(k)}\right\}\Big|_{\substack{i=0,\cdots,H_l-1 \quad t=0,\cdots,T-1\\ j=0,\cdots,W_l-1 \quad k=0,\cdots,C_l-1}} \qquad l=1,\cdots,L \quad layer$$

其中，L 表示最大层次，l 表示当前层次，T 表示 mini-batch 容量值，t 表示 mini-batch 游标，i, j, k 分别表示高、宽、通道数，一张图片为一个 3 维阵（3d-tensor），则由 T 张图片组成的 mini-batch 数据块 $X_{\times H_l,\times W_l}^{(l),(\times T),(\times C_l)}$ 为一个 4 维阵（4d-tensor）。不同层次的数据块其尺寸参数 $\{H_l, W_l, C_l, T\}$ 可以不同，T 一般保持不变。

视频片段数据块

与 mini-batch 数据块类似，对于以视频片段（consecutive frames）为训练样本的网络，一次输入数据为时间上连续的若干个帧（图片），用 $t=0,\cdots,T-1$ 对帧进行编码，仍然用 $l=1,\cdots,L$ 对网络的层次进行编码，其它与前述一致，适用于视频片段样本的数据块的一般形式为：

$$X_{\times H_l,\times W_l}^{(l),(\times T),(\times C_l)} = \left\{ x_{i,j}^{(l),(t),(k)} \right\}\Big|_{\substack{i=0,\cdots,H_l-1 \\ j=0,\cdots,W_l-1}} \quad \substack{t=0,\cdots,T-1 \\ k=0,\cdots,C_l-1} \qquad l = 1,\cdots,L \quad layer$$

其中，L 表示最大层次，l 表示当前层次，T 表示一个视频片段样本的总帧数，t 表示帧游标，H_l, W_l 分别表示帧的高、宽，C_l 表示通道数，i, j 分别表示帧的高、宽游标，k 表示通道游标。由 T 帧图片组成的一个视频片段数据块 $X_{\times H_l,\times W_l}^{(l),(\times T),(\times C_l)}$ 为一个 4 维阵（4d-tensor）。不同层次的数据块其尺寸参数 $\{H_l, W_l, C_l, T\}$ 可以不同，T 一般保持不变。

数据块流图（Graph of Tensor Flow）

从 mini-batch 数据块和视频片段数据块的定义可知，依据数据处理模块化原则，数据处理模块的输入、输出数据块应该表达为独立单元的数据块，一般表示为一个 4 维阵（4d-tensor），

$$X_{\times H_l,\times W_l}^{(l),(\times T),(\times C_l)} = \left\{ x_{i,j}^{(l),(t),(k)} \right\}\Big|_{\substack{i=0,\cdots,H_l-1 \\ j=0,\cdots,W_l-1}} \quad \substack{t=0,\cdots,T-1 \\ k=0,\cdots,C_l-1} \qquad l = 1,\cdots,L \quad vertex/node$$

4 维阵数据块 $X_{\times H_l,\times W_l}^{(l),(\times T),(\times C_l)}$ 相当于一个数据结点（vertex/node），而数据处理模块（Conv, BN, ReLU, Pooling, Full, Softmax 等）相当于结点之间的连线（edge），整个神经网络可以看做是一个数据块处理流图（graph of tensor flow）。上式中的 (l) 相当于是 graph 中的结点编号。

Convolutional Operate
(2d conv/asymmetric kernel)

前向卷积输入输出数据

有如下假设：

网络 l 结点（left） t 时刻和 r 结点（right） t 时刻的激活值数据块（activation tensor）分别为，

$$X_{\times H,\times W}^{(l),(t),(\times C_l)}=\left\{x_{i',j'}^{(l),(t),(k')}\right\}\Big|_{\substack{i'=0,\cdots,H-1\\ j'=0,\cdots,W-1\\ k'=0,\cdots,C_l-1}}$$

$$X_{\times H,\times W}^{(r),(t),(\times C_r)}=\left\{x_{i,j}^{(r),(t),(k)}\right\}\Big|_{\substack{i=0,\cdots,H-1\\ j=0,\cdots,W-1\\ k=0,\cdots,C_r-1}}$$

网络 r 结点 t 时刻的第 k 个 feature map 为

$$X_{\times H,\times W}^{(r),(t),(k)}=\left\{x_{i,j}^{(r),(t),(k)}\right\}\Big|_{\substack{i=0,\cdots,H-1\\ j=0,\cdots,W-1}}\qquad k=0,\cdots,C_r-1$$

为了简化表达公式，假设 r 结点 和 l 结点的特征图高宽（$H\times W$）不变，l 结点数据块的通道数为 C_l，r 结点数据块的通道数为 C_r。

注解：目前主流的神经网络，其卷积操作不改变特征图的高宽尺寸。

前向卷积权值阵 3d 阵向量

假设，网络 r 结点第 k 个 feature map 对应的 forward conv 为 $Conv_F{}^{(r),(k)}$，$k=0,\cdots,C_r-1$，对应的权值阵 $W_{\times M,\times N}^{(r),(k),(\times C_l)}$ 为一个权值数据块（3d-weight tensor），可以视为一个由二维卷积窗口组成的**阵向量**（matrices in vector），卷积窗口高宽为（$M\times N$），一般可以取（3×3）、（5×5）等，但也不限定等高宽比，以及奇数尺寸。

$$W_{\times M,\times N}^{(r),(k),(\times C_l)}=\left[W_{\times M,\times N}^{(r),(k),(0)}\cdots W_{\times M,\times N}^{(r),(k),(C_l-1)}\right]_{1\times C_l}=\left\{W_{\times M,\times N}^{(r),(k),(k')}\right\}_{k'=0,\cdots,C_l-1}$$

$$=\left\{\left\{w_{p,q}^{(r),(k),(k')}\right\}\Big|_{\substack{p\in[0,M-1]\\ q\in[0,N-1]}}\right\}_{k'=0,\cdots,C_l-1}=\left\{\begin{bmatrix} w_{0,0}^{(r),(k),(k')} & \cdots & w_{0,N-1}^{(r),(k),(k')}\\ \vdots & \ddots & \vdots\\ w_{M-1,0}^{(r),(k),(k')} & \cdots & w_{M-1,N-1}^{(r),(k),(k')}\end{bmatrix}_{M\times N}\right\}_{k'=0,\cdots,C_l-1}$$

$$k = 0, \cdots, C_r - 1$$

注记：为了与特征图保持一致，卷积窗口 $W_{\times M,\times N}^{(r),(k),(k')} = \left\{ w_{p,q}^{(r),(k),(k')} \right\}\Big|_{\substack{p\in[0,M-1] \\ q\in[0,N-1]}}$ 中的左下标 p 表示高度游标，右下标 q 表示宽度游标，游标变化范围从 0 开始，方便编程。

前向卷积计算公式

第 k 个输出 feature map $X_{\times H,\times W}^{(r),(t),(k)}$ 的计算公式为

$$X_{\times H,\times W}^{(r),(t),(k)} = Conv_F{}^{(r),(k)} \left(X_{\times H,\times W}^{(l),(t),(\times C_l)}, W_{\times M,\times N}^{(r),(k),(\times C_l)} \right) \qquad k = 0, \cdots, C_r - 1$$

及每个分量 $x_{i,j}^{(r),(t),(k)}$ 的前向卷积计算公式为

$$x_{i,j}^{(r),(t),(k)} = \sum_{k'=0,\cdots,C_l-1} \sum_{p\in[0,M-1]} \sum_{q\in[0,N-1]} w_{p,q}^{(r),(k),(k')} \cdot x_{i-m+p,j-n+q}^{(l),(t),(k')}$$

$$\begin{cases} i = 0, \cdots, H-1 \\ j = 0, \cdots, W-1 \\ k = 0, \cdots, C_r - 1 \end{cases}$$

对于超出 l 结点输入 feature map 范围的输入值取 0，称为输入边缘条件 pad condition，

$$x_{i-m+p,j-n+q}^{(l),(t),(k')} = 0 \quad \begin{cases} p \in [0, M-1] \\ q \in [0, N-1] \end{cases} \quad \begin{cases} i-m+p < 0 \quad or \quad i-m+p > H-1 \\ j-n+q < 0 \quad or \quad j-n+q > W-1 \\ k' = 0, \cdots, C_l - 1 \end{cases}$$

前向卷积核锚点

前向卷积计算公式中的参量 (m, n)，为卷积窗口内的**卷积核锚点(base point of conv)**，或称为**卷积核中心点(conv center)**，比如对于 3×3 的卷积核，其中心点为 (1,1)，对于 5×5 的卷积核，其中心点为 (2,2)，对于 7×7 的卷积核，其中心点为 (3,3)。

$$\begin{bmatrix} (0,0) & (0,1) & (0,2) \\ (1,0) & \boxed{(1,1)} & (1,2) \\ (2,0) & (2,1) & (2,2) \end{bmatrix}_{3\times3} \qquad \begin{bmatrix} (0,0) & (0,1) & (0,2) & (0,3) & (0,4) \\ (1,0) & (1,1) & (1,2) & (1,3) & (1,4) \\ (2,0) & (2,1) & \boxed{(2,2)} & (2,3) & (2,4) \\ (3,0) & (3,1) & (3,2) & (3,3) & (3,4) \\ (4,0) & (4,1) & (4,2) & (4,3) & (4,4) \end{bmatrix}_{5\times5}$$

但是这个卷积核锚点也不是一定要在中心位置，这个锚点只要在卷积窗口内，直观上说得通即可。

$$anchor\ (m,n):\quad \begin{cases}(0 \le m \le M-1)\\ and\\ (0 \le n \le N-1)\end{cases}$$

[例]，对于 5×6 的卷积核，非等高宽比，非双奇数尺寸，窗口中没有中心点，可以任选锚点，比如 $(m,n)=(2,1)$，则在前向公式中的权值 $\left\{w_{p,q}^{(r),(k),(k')}\right\}_{\substack{p\in[0,M-1]\\ q\in[0,N-1]\\ k'=0,\cdots,C_l-1}}$ 的 $\{p,q\}_{\substack{p\in[0,5-1]\\ q\in[0,6-1]}}$ 下标阵为

$$\{p,q\}_{\substack{p\in[0,5-1]\\ q\in[0,6-1]}}=\begin{bmatrix}(0,0) & (0,1) & (0,2) & (0,3) & (0,4) & (0,5)\\ (1,0) & (1,1) & (1,2) & (1,3) & (1,4) & (1,5)\\ (2,0) & \boxed{(2,1)} & (2,2) & (2,3) & (2,4) & (2,5)\\ (3,0) & (3,1) & (3,2) & (3,3) & (3,4) & (3,5)\\ (4,0) & (4,1) & (4,2) & (4,3) & (4,4) & (4,5)\end{bmatrix}_{5\times 6}$$

在前向公式中与其对应的激活值 $\left\{x_{i-m+p,j-n+q}^{(l),(t),(k')}\right\}_{\substack{p\in[0,M-1]\\ q\in[0,N-1]\\ k'=0,\cdots,C_l-1}}$ 下标阵 $\{i-m+p,j-n+q\}_{\substack{m=2\\ n=1\\ p\in[0,5-1]\\ q\in[0,6-1]}}$ 为

$$\{i-m+p,j-n+q\}_{\substack{m=2\\ n=1\\ p\in[0,5-1]\\ q\in[0,6-1]}}=\begin{bmatrix}(i-2,j-1) & (i-2,j) & (i-2,j+1) & (i-2,j+2) & (i-2,j+3) & (i-2,j+4)\\ (i-1,j-1) & (i-1,j) & (i-1,j+1) & (i-1,j+2) & (i-1,j+3) & (i-1,j+4)\\ (i\quad,j-1) & \boxed{(i,j)} & (i\quad,j+1) & (i\quad,j+2) & (i\quad,j+3) & (i\quad,j+4)\\ (i+1,j-1) & (i+1,j) & (i+1,j+1) & (i+1,j+2) & (i+1,j+3) & (i+1,j+4)\\ (i+2,j-1) & (i+2,j) & (i+2,j+1) & (i+2,j+2) & (i+2,j+3) & (i+2,j+4)\end{bmatrix}_{5\times 6}$$

$$=\begin{bmatrix}(i-2+0,j-1+0) & (i-2+0,j-1+1) & (i-2+0,j-1+2) & (i-2+0,j-1+3) & (i-2+0,j-1+4) & (i-2+0,j-1+5)\\ (i-2+1,j-1+0) & (i-2+1,j-1+1) & (i-2+1,j-1+2) & (i-2+1,j-1+3) & (i-2+1,j-1+4) & (i-2+1,j-1+5)\\ (i-2+2,j-1+0) & \boxed{(i-2+2,j-1+1)} & (i-2+2,j-1+2) & (i-2+2,j-1+3) & (i-2+2,j-1+4) & (i-2+2,j-1+5)\\ (i-2+3,j-1+0) & (i-2+3,j-1+1) & (i-2+3,j-1+2) & (i-2+3,j-1+3) & (i-2+3,j-1+4) & (i-2+3,j-1+5)\\ (i-2+4,j-1+0) & (i-2+4,j-1+1) & (i-2+4,j-1+2) & (i-2+4,j-1+3) & (i-2+4,j-1+4) & (i-2+4,j-1+5)\end{bmatrix}_{5\times 6}$$

从此例可以看出，卷积核锚点的作用是确定卷积窗口在特征图上进行扫描操作时的定位点以及特征取值范围，即在前向卷积计算公式中，当游标 (p,q) 取锚点值 (m,n) 时，即 $\begin{cases}p=m\\ q=n\end{cases}$ 时，卷积窗口中的锚点权值 $w_{m,n}^{(r),(k),(k')}$ 恰好与输入特征图中的 $x_{i,j}^{(l),(t),(k')}$ 点以及输出特征图中的 $x_{i,j}^{(r),(t),(k)}$ 点相对应。

注解：推广来看，这个锚点也不是一定要在卷积窗口的内部，也可以在卷积窗口之外，

$$outer\ anchor\ (m,n):\quad \begin{cases}(m<0\quad or\quad m>M-1)\\ or\\ (n<0\quad or\quad n>N-1)\end{cases}$$

上述前向卷积公式仍然成立，只要满足前述输入边缘条件即可。

[例]，对于 3×3 的卷积核（中心加框部分），可以选择卷积核窗口之外的 $(-1,-2)$ 为锚点，$(m,n)=(-1,-2)$。则在前向公式中的权值 $\{p,q\}_{\substack{p\in[0,3-1]\\ q\in[0,3-1]}}$ 下标阵为

$$\begin{matrix}
(-3,-3) & (-3,-2) & (-3,-1) & (-3,0) & (-3,1) & (-3,2) & (-3,3) & (-3,4) & (-3,5) \\
(-2,-3) & (-2,-2) & (-2,-1) & (-2,0) & (-2,1) & (-2,2) & (-2,3) & (-2,4) & (-2,5) \\
(-1,-3) & \boxed{(-1,-2)} & (-1,-1) & (-1,0) & (-1,1) & (-1,2) & (-1,3) & (-1,4) & (-1,5) \\
(+0,-3) & (+0,-2) & (+0,-1) & \boxed{(+0,0)} & (+0,1) & (+0,2) & (+0,3) & (+0,4) & (+0,5) \\
(+1,-3) & (+1,-2) & (+1,-1) & (+1,0) & (+1,1) & (+1,2) & (+1,3) & (+1,4) & (+1,5) \\
(+2,-3) & (+2,-2) & (+2,-1) & (+2,0) & (+2,1) & (+2,2) & (+2,3) & (+2,4) & (+2,5) \\
(+3,-3) & (+3,-2) & (+3,-1) & (+3,0) & (+3,1) & (+3,2) & (+3,3) & (+3,4) & (+3,5) \\
(+4,-3) & (+4,-2) & (+4,-1) & (+4,0) & (+4,1) & (+4,2) & (+4,3) & (+4,4) & (+4,5) \\
(+5,-3) & (+5,-2) & (+5,-1) & (+5,0) & (+5,1) & (+5,2) & (+5,3) & (+5,4) & (+5,5)
\end{matrix}$$

在前向公式中与其对应的激活值下标阵 $\{i-m+p, j-n+q\}_{\substack{m=-1 \\ n=-2 \\ p\in[0,3-1] \\ q\in[0,3-1]}}$ 为

$$\begin{matrix}
(i-2,j-1) & (i-2,j) & (i-2,j+1) & (i-2,j+2) & (i-2,j+3) & (i-2,j+4) & (i-2,j+5) & (i-2,j+6) & (i-2,j+7) \\
(i-1,j-1) & (i-1,j) & (i-1,j+1) & (i-1,j+2) & (i-1,j+3) & (i-1,j+4) & (i-1,j+5) & (i-1,j+6) & (i-1,j+7) \\
(i,j-1) & \boxed{(i,j)} & (i,j+1) & (i,j+2) & (i,j+3) & (i,j+4) & (i,j+5) & (i,j+6) & (i,j+7) \\
(i+1,j-1) & (i+1,j) & (i+1,j+1) & \boxed{(i+1,j+2)} & \boxed{(i+1,j+3)} & \boxed{(i+1,j+4)} & (i+1,j+5) & (i+1,j+6) & (i+1,j+7) \\
(i+2,j-1) & (i+2,j) & (i+2,j+1) & \boxed{(i+2,j+2)} & \boxed{(i+2,j+3)} & \boxed{(i+2,j+4)} & (i+2,j+5) & (i+2,j+6) & (i+2,j+7) \\
(i+3,j-1) & (i+3,j) & (i+3,j+1) & \boxed{(i+3,j+2)} & \boxed{(i+3,j+3)} & \boxed{(i+3,j+4)} & (i+3,j+5) & (i+3,j+6) & (i+3,j+7) \\
(i+4,j-1) & (i+4,j) & (i+4,j+1) & (i+4,j+2) & (i+4,j+3) & (i+4,j+4) & (i+4,j+5) & (i+4,j+6) & (i+4,j+7) \\
(i+5,j-1) & (i+5,j) & (i+5,j+1) & (i+5,j+2) & (i+5,j+3) & (i+5,j+4) & (i+5,j+5) & (i+5,j+6) & (i+5,j+7) \\
(i+6,j-1) & (i+6,j) & (i+6,j+1) & (i+6,j+2) & (i+6,j+3) & (i+6,j+4) & (i+6,j+5) & (i+6,j+6) & (i+6,j+7)
\end{matrix}$$

此例的卷积区域在锚点（定位点）之外，称为**飞域卷积**（exclave convolution）。

<u>前向卷积权值阵 4d 阵中阵</u>

网络 r 结点的所有的 forward convs 为一个并行 conv 组（conv bank），记为 $Conv_F{}^{(r)} = \{Conv_F{}^{(r),(k)}\}_{k=0,\cdots,C_r-1}$，则以上前向卷积公式可以整合表示为，

$$X_{\times H,\times W}^{(l),(t),(\times C_l)} \xrightarrow{Conv_F{}^{(r)}} \left\{ \begin{matrix} Conv_F{}^{(r),(0)}\left(X_{\times H,\times W}^{(l),(t),(\times C_l)}, W_{\times M,\times N}^{(r),(0),(\times C_l)}\right) \\ \vdots \\ Conv_F{}^{(r),(C_r-1)}\left(X_{\times H,\times W}^{(l),(t),(\times C_l)}, W_{\times M,\times N}^{(r),(C_r-1),(\times C_l)}\right) \end{matrix} \right\}_{C_r\times 1}$$

$$\rightarrow \left\{ \begin{matrix} X_{\times H,\times W}^{(r),(t),(0)} \\ \vdots \\ X_{\times H,\times W}^{(r),(t),(C_r-1)} \end{matrix} \right\}_{C_r\times 1} \rightarrow X_{\times H,\times W}^{(r),(t),(\times C_r)}$$

即

$$X_{\times H,\times W}^{(r),(t),(\times C_r)} = Conv_F{}^{(r)}\left(X_{\times H,\times W}^{(l),(t),(\times C_l)}, W_{\times M,\times N}^{(r),(\times C_r),(\times C_l)}\right)$$

网络 l 结点的数据块 $X_{\times H,\times W}^{(l),(t),(\times C_l)}$，经过 conv bank $Conv_F{}^{(r)} = \{Conv_F{}^{(r),(k)}\}_{k=0,\cdots,C_r-1}$，得到 r 结点的数据块 $X_{\times H,\times W}^{(r),(t),(\times C_r)}$。

前向卷积公式实际为多组合卷积（multiple combination convolutions），将第 r 结点的所有的权值阵（3d 阵向量）合并表达为一个前向阵中阵（matrices in matrix），

$$W_{\times M,\times N}^{(r),(\times C_r),(\times C_l)}=\left\{W_{\times M,\times N}^{(r),(k),(k')}\right\}_{\substack{k=0,\cdots,C_r-1\\k'=0,\cdots,C_l-1}}=\begin{bmatrix}W_{\times M,\times N}^{(r),(0),(\times C_l)}\\ \vdots\\ W_{\times M,\times N}^{(r),(C_r-1),(\times C_l)}\end{bmatrix}_{C_r}$$

$$W_{\times M,\times N}^{(r),(\times C_r),(\times C_l)}=\begin{bmatrix}W_{\times M,\times N}^{(r),(0),(0)} & \cdots & W_{\times M,\times N}^{(r),(0),(C_l-1)}\\ \vdots & \ddots & \vdots\\ W_{\times M,\times N}^{(r),(C_r-1),(0)} & \cdots & W_{\times M,\times N}^{(r),(C_r-1),(C_l-1)}\end{bmatrix}_{C_r\times C_l}$$

该前向阵中阵为一个权值数据块（4d-weight tensor），实际上是一个 4 维阵（4d-tensor）。每一个 $W_{\times M,\times N}^{(r),(k),(\times C_l)},k=0,\cdots,C_r-1$ 为一个 3 维阵（3d 阵向量），对应于 r 结点的一个（第 k 个）卷积核及对应的 feature map，所以计算次序为行优先。

反向敏感值计算输入输出数据

反向传播计算的核心是敏感值 δ 的计算，事实上，一般而言，r 结点（right）的激活值数据块

$$X_{\times H_r,\times W_r}^{(r),(t),(\times C_r)}=\left\{x_{i,j}^{(r),(t),(k)}\right\}\Big|_{\substack{i=0,\cdots,H_r-1\\j=0,\cdots,W_r-1\\k=0,\cdots,C_r-1}}\qquad t=0,\cdots,T-1$$

的每一个分量 $x_{i,j}^{(r),(t),(k)}$ 都对应有一个敏感值 $\delta_{i,j}^{(r),(t),(k)}$。为了统一表达，仍然采用前述假设，网络 l 结点（left） t 时刻和 r 结点（right） t 时刻的激活值数据块分别为，

$$X_{\times H,\times W}^{(l),(t),(\times C_l)}=\left\{x_{i',j'}^{(l),(t),(k')}\right\}\Big|_{\substack{i'=0,\cdots,H-1\\j'=0,\cdots,W-1\\k'=0,\cdots,C_l-1}}\qquad X_{\times H,\times W}^{(r),(t),(\times C_r)}=\left\{x_{i,j}^{(r),(t),(k)}\right\}\Big|_{\substack{i=0,\cdots,H-1\\j=0,\cdots,W-1\\k=0,\cdots,C_r-1}}$$

则网络 l 结点 t 时刻和 r 结点 t 时刻对应的敏感值数据块（activation tensor）分别为，

$$\Delta_{\times H,\times W}^{(l),(t),(\times C_l)}=\left\{\delta_{i',j'}^{(l),(t),(k')}\right\}\Big|_{\substack{i'=0,\cdots,H-1\\j'=0,\cdots,W-1\\k'=0,\cdots,C_l-1}}\qquad \Delta_{\times H,\times W}^{(r),(t),(\times C_r)}=\left\{\delta_{i,j}^{(r),(t),(k)}\right\}\Big|_{\substack{i=0,\cdots,H-1\\j=0,\cdots,W-1\\k=0,\cdots,C_r-1}}$$

反向敏感值计算问题

假设，r 结点分量 $x_{i,j}^{(r),(t),(k)}$ 对应的敏感值 $\delta_{i,j}^{(r),(t),(k)}$ 已知。求 l 结点分量 $x_{i',j'}^{(l),(t),(k')}$ 对应的敏感

值 $\delta_{i',j'}^{(l),(t),(k')}$，就是说，反向算法的输入为 r 结点的敏感值数据块 $\Delta_{\times H,\times W}^{(r),(t),(\times C_r)}$，输出为 l 结点的 $\Delta_{\times H,\times W}^{(l),(t),(\times C_l)}$。

解法原理 ：（1）从前向激活值计算公式推导前向激活值数据关系图，（2）再从此图中推理反向激活值数据关系图，这个反向激活值关系图与反向敏感值关系图是一致的，（3）由此构造反向敏感值计算公式。

前向激活值数据关系图（多对一）

先从前向公式中推导前向激活值数据关系图，前向卷积公式如下，

$$x_{i,j}^{(r),(t),(k)} = \sum_{k'=0,\cdots,C_l-1} \sum_{p\in[0,M-1]} \sum_{q\in[0,N-1]} w_{p,q}^{(r),(k),(k')} \cdot x_{i-m+p,j-n+q}^{(l),(t),(k')}$$

$$\begin{cases} i = 0,\cdots,H-1 \\ j = 0,\cdots,W-1 \\ k = 0,\cdots,C_r-1 \end{cases}$$

其中 (m,n) 为卷积窗口内的卷积核锚点，则前向传播关系图为从 l 到 r 的多对一 $(N_f\ to\ 1)$ 关系图，

$$\left\{x_{i-m+p,j-n+q}^{(l),(t),(k')}\right\}_{\substack{p\in[0,M-1]\\ q\in[0,N-1]\\ k'=0,\cdots,C_l-1}} \xrightarrow{N_f\ to\ N_f} \left\{w_{p,q}^{(r),(k),(k')}\right\}_{\substack{p\in[0,M-1]\\ q\in[0,N-1]\\ k'=0,\cdots,C_l-1}} \xrightarrow{N_f\ to\ 1} x_{i,j}^{(r),(t),(k)}$$

$$\begin{cases} i = 0,\cdots,H-1 \\ j = 0,\cdots,W-1 \\ k = 0,\cdots,C_r-1 \end{cases} \qquad N_f = C_l \times M \times N$$

前向激活值数据关系图（一对一，右端定位）

进一步，将多对一 $(N_f\ to\ 1)$ 的前向传播关系图分解表达为一对一 $(1\ to\ 1)$ 前向关系图，

$$x_{i-m+p,j-n+q}^{(l),(t),(k')} \xrightarrow{1\ to\ 1} w_{p,q}^{(r),(k),(k')} \xrightarrow{1\ to\ 1} x_{i,j}^{(r),(t),(k)}$$

$$\begin{cases} p \in [0,M-1] \\ q \in [0,N-1] \\ k' = 0,\cdots,C_l-1 \end{cases} \qquad \begin{cases} i = 0,\cdots,H-1 \\ j = 0,\cdots,W-1 \\ k = 0,\cdots,C_r-1 \end{cases}$$

其直观含义为，l 结点 feature map 的输入值 $x_{i-m+p,j-n+q}^{(l),(t),(k')}$ 乘以权值 $w_{p,q}^{(r),(k),(k')}$ 向 r 结点 feature

map 的输出值 $x_{i,j}^{(r),(t),(k)}$ 贡献了部分激活值。

前向输入边缘条件

形式上，这组一对一 $(1\ to\ 1)$ 前向关系图总数有 $(M\times N\times H\times W\times C_l\times C_r)$，但是对于超出 l 结点 feature map 范围的输入值 $x_{i-m+p,j-n+q}^{(l),(t),(k')}$ 取 0，称为前向输入边缘条件 pad condition，

$$x_{i-m+p,j-n+q}^{(l),(t),(k')}=0 \quad \begin{cases} p\in[0,M-1] \\ q\in[0,N-1] \end{cases} \quad \begin{cases} i-m+p<0 \quad or \quad i-m+p>H-1 \\ j-n+q<0 \quad or \quad j-n+q>W-1 \\ k'=0,\cdots,C_l-1 \end{cases}$$

凡是涉及到满足输入边缘条件数据的一对一 $(1\ to\ 1)$ 前向关系图均视为无效关系图。

准确计算所有的无效关系图比较繁琐，下面只计算当 i 和 j 分别取边界值时的无效关系图总数，而且限定卷积核锚点位于卷积窗口内部。

（1）上边缘：假设 $0\le m\le M-1$，取 $i=0$，则由 $i-m+p<0$ 得到 $p<m$，结合 $p\in[0,M-1]$，可知 $p=0,\cdots,m-1$，共有 m 行 l 结点 feature map 的上边缘为无效数据。

（2）下边缘：假设 $0\le m\le M-1$，取 $i=H-1$，则由 $i-m+p>H-1$ 得到 $p>m$，结合 $p\in[0,M-1]$，可知 $p=m+1,\cdots,M-1$，共有 $(M-1-m)$ 行 l 结点 feature map 的下边缘为无效数据。

（3）左边缘：假设 $0\le n\le N-1$，取 $j=0$，则由 $j-n+q<0$ 得到 $q<n$，结合 $q\in[0,N-1]$，可知 $q=0,\cdots,n-1$，共有 n 行 l 结点 feature map 的左边缘为无效数据。

（4）右边缘：假设 $0\le n\le N-1$，取 $j=W-1$，则由 $j-n+q>W-1$ 得到 $q>n$，结合 $q\in[0,N-1]$，可知 $q=n+1,\cdots,N-1$，共有 $(N-1-n)$ 行 l 结点 feature map 的右边缘为无效数据。

综合 4 种边缘部分尺寸，以及中间有效数据部分尺寸 $M\times N$，一张输入特征图的边缘无效数据总数为 $(3MN-2M-2N+1)$，

$$N\times m+N\times(M-1-m)+M\times n+M\times(N-1-n)+$$
$$n\times m+n\times(M-1-m)+(N-1-n)\times(M-1-m)+m\times(N-1-n)$$

$$=N\times(M-1)+M\times(N-1)+n\times(M-1)+(N-1-n)\times(M-1)$$
$$=N\times(M-1)+M\times(N-1)+(N-1)\times(M-1)$$

$$=3MN-2M-2N+1$$

再考虑到多对多特征图关系 $\begin{cases} k'=0,\cdots,C_l-1 \\ k=0,\cdots,C_r-1 \end{cases}$，边缘无效数据总数为

$$C_l \times C_r \times (3MN - 2M - 2N + 1)$$

前向激活值数据关系图（一对一，左端定位）

为使 l 结点特征图下标为主下标，做位置下标变换

$$\begin{cases} i' = i - m + p \\ j' = j - n + q \end{cases}$$

则前向传播一对一 $(1\ to\ 1)$ 关系图变为，

$$x_{i',j'}^{(l),(t),(k')} \xrightarrow{1\ to\ 1} w_{p,q}^{(r),(k),(k')} \xrightarrow{1\ to\ 1} x_{i'+m-p,j'+n-q}^{(r),(t),(k)}$$

$$\begin{cases} i' = 0,\cdots,(H-1) \\ j' = 0,\cdots,(W-1) \\ k' = 0,\cdots,C_l - 1 \end{cases} \quad \begin{cases} p \in [0, M-1] \\ q \in [0, N-1] \\ k = 0,\cdots,C_r - 1 \end{cases}$$

注记：6 个游标取值范围是一致的，但分组方式已经改变了。

$$\begin{cases} p \in [0, M-1] \\ q \in [0, N-1] \\ k' = 0,\cdots,C_l - 1 \end{cases} \begin{cases} i' = 0,\cdots,(H-1) \\ j' = 0,\cdots,(W-1) \\ k = 0,\cdots,C_r - 1 \end{cases} \Rightarrow \begin{cases} i' = 0,\cdots,(H-1) \\ j' = 0,\cdots,(W-1) \\ k' = 0,\cdots,C_l - 1 \end{cases} \begin{cases} p \in [0, M-1] \\ q \in [0, N-1] \\ k = 0,\cdots,C_r - 1 \end{cases}$$

与前述一样，形式上，这组一对一 $(1\ to\ 1)$ 关系图总数也有 $(M \times N \times H \times W \times C_l \times C_r)$，但是对于超出 r 结点 feature map 范围的输出值 $x_{i'+m-p,j'+n-q}^{(r),(t),(k)}$ 不做前向计算，直接取 0，称为输出边缘条件 pad condition，

$$x_{i'+m-p,j'+n-q}^{(r),(t),(k)} = 0 \quad \begin{cases} p \in [0, M-1] \\ q \in [0, N-1] \end{cases} \quad \begin{cases} i' + m - p < 0 \quad or \quad i' + m - p > H - 1 \\ j' + n - q < 0 \quad or \quad j' + n - q > W - 1 \\ k = 0,\cdots,C_r - 1 \end{cases}$$

凡是涉及到满足输出边缘条件数据的一对一 $(1\ to\ 1)$ 前向关系图均视为无效关系图。

同样地，准确计算所有的无效关系图也比较繁琐，下面只计算当 i' 和 j' 分别取边界值时的无效关系图总数，同样限定卷积核锚点位于卷积窗口内部。

（1）假设 $0 \le m \le M-1$，取 $i' = 0$，则由 $i' + m - p < 0$ 得到 $p > m$，结合 $p \in [0, M-1]$，可知 $p = m+1,\cdots,M-1$，共有 $(M-1-m)$ 行 r 结点 feature map 的上边缘为无效数据。

（2）假设 $0 \le m \le M-1$，取 $i' = H-1$，则由 $i' + m - p > H - 1$ 得到 $p < m$，结合 $p \in [0, M-1]$，可知 $p = 0,\cdots,m-1$，共有 m 行 r 结点 feature map 的下边缘为无效数据。

（3）假设 $0 \le n \le N-1$，取 $j'=0$，则由 $j'+n-q<0$ 得到 $q>n$，结合 $q \in [0,N-1]$，可知 $q=n+1,\cdots,N-1$，共有 $(N-1-n)$ 行 r 结点 feature map 的左边缘为无效数据。

（4）假设 $0 \le n \le N-1$，取 $j'=W-1$，则由 $j'+n-q>W-1$ 得到 $q<n$，结合 $q \in [0,N-1]$，可知 $q=0,\cdots,n-1$，共有 n 行 r 结点 feature map 的右边缘为无效数据。

综合 4 种边缘部分尺寸，以及中间有效数据部分尺寸 $M \times N$，一对一特征图的边缘无效数据总数为

$$N \times (M-1-m)+N \times m+M \times (N-1-n)+M \times n+$$
$$(N-1-n) \times (M-1-m)+n \times (M-1-m)+m \times (N-1-n)+n \times m$$

$$= N \times (M-1)+M \times (N-1)+(M-1-m) \times (N-1)+m \times (N-1)$$
$$= N \times (M-1)+M \times (N-1)+(M-1) \times (N-1)$$

$$= 3MN-2M-2N+1$$

再考虑到多对多特征图关系 $\begin{cases} k'=0,\cdots,C_l-1 \\ k=0,\cdots,C_r-1 \end{cases}$，边缘无效数据总数为

$$C_l \times C_r \times (3MN-2M-2N+1)$$

结论：两种前向传播一对一 $(1\ to\ 1)$ 关系图的总无效关系图个数是一致的（在指定条件下）。

<u>前向激活值数据关系图（一对多）</u>

进一步，将左端定位的前向传播一对一 $(1\ to\ 1)$ 关系图，

$$x_{i',j'}^{(l),(t),(k')} \xrightarrow{1\ to\ 1} w_{p,q}^{(r),(k),(k')} \xrightarrow{1\ to\ 1} x_{i'+m-p,j'+n-q}^{(r),(t),(k)}$$

$$\begin{cases} i'=0,\cdots,(H-1) \\ j'=0,\cdots,(W-1) \\ k'=0,\cdots,C_l-1 \end{cases} \quad \begin{cases} p \in [0,M-1] \\ q \in [0,N-1] \\ k=0,\cdots,C_r-1 \end{cases}$$

合并表达为一对多 $(1\ to\ N_b)$ 的关系图，

$$x_{i',j'}^{(l),(t),(k')} \xrightarrow{1\ to\ N_b} \left\{ w_{p,q}^{(r),(k),(k')} \right\}_{\substack{p\in[0,M-1] \\ q\in[0,N-1] \\ k=0,\cdots,C_r-1}} \xrightarrow{N_b\ to\ N_b} \left\{ x_{i'+m-p,j'+n-q}^{(r),(t),(k)} \right\}_{\substack{p\in[0,M-1] \\ q\in[0,N-1] \\ k=0,\cdots,C_r-1}}$$

$$\begin{cases} i'=0,\cdots,H-1 \\ j'=0,\cdots,W-1 \\ k'=0,\cdots,C_l-1 \end{cases} \qquad N_b = C_r \times M \times N$$

其直观含义为，l 结点 feature map 的输入值 $x_{i',j'}^{(l),(t),(k')}$ 分别乘以权值 $w_{p,q}^{(r),(k),(k')}$ 并分别向对应的

r 结点 feature map 的输出值 $x_{i'+m-p,j'+n-q}^{(r),(t),(k)}$ 贡献了部分激活值。

<u>激活值反向数据关系图（多对一）</u>

将前向激活值数据关系图（一对多）左右翻转，则推导出激活值反向传播关系图（多对一）为，

$$\left\{x_{i'+m-p,j'+n-q}^{(r),(t),(k)}\right\}_{\substack{p\in[0,M-1]\\ q\in[0,N-1]\\ k=0,\cdots,C_r-1}} \xrightarrow{N_b\ to\ N_b} \left\{w_{p,q}^{(r),(k),(k')}\right\}_{\substack{p\in[0,M-1]\\ q\in[0,N-1]\\ k=0,\cdots,C_r-1}} \xrightarrow{N_b\ to\ 1} x_{i',j'}^{(l),(t),(k')}$$

$$\begin{cases} i'=0,\cdots,H-1 \\ j'=0,\cdots,W-1 \\ k'=0,\cdots,C_l-1 \end{cases}$$

进一步，做卷积核下标变换

$$\begin{cases} m-p=-m'+p' \\ n-q=-n'+q' \end{cases} \quad \Rightarrow \quad \begin{cases} p=m+m'-p' \\ q=n+n'-q' \end{cases}$$

则有

$$\begin{cases} 0\le p\le M-1 \\ 0\le q\le N-1 \end{cases} \Rightarrow \begin{cases} 0\le m+m'-p'\le M-1 \\ 0\le n+n'-q'\le N-1 \end{cases} \Rightarrow \begin{cases} m+m'-M+1\le p'\le m+m' \\ n+n'-N+1\le q'\le n+n' \end{cases}$$

事实上，(p',q') 仍然为卷积核下标，即应当满足条件 $\begin{cases} 0\le p'\le M-1 \\ 0\le q'\le N-1 \end{cases}$，则有 $\begin{cases} m+m'=M-1 \\ n+n'=N-1 \end{cases}$，卷积核下标变换实际为

$$\begin{cases} p=M-1-p' \\ q=N-1-q' \end{cases}$$

激活值反向传播关系图（多对一）变为，

$$\left\{x_{i'-m'+p',j'-n'+q'}^{(r),(t),(k)}\right\}_{\substack{p'\in[0,M-1]\\ q'\in[0,N-1]\\ k=0,\cdots,C_r-1}} \xrightarrow{N_b\ to\ N_b} \left\{w_{M-1-p',N-1-q'}^{(r),(k),(k')}\right\}_{\substack{p'\in[0,M-1]\\ q'\in[0,N-1]\\ k=0,\cdots,C_r-1}} \xrightarrow{N_b\ to\ 1} x_{i',j'}^{(l),(t),(k')}$$

$$\begin{cases} i'=0,\cdots,H-1 \\ j'=0,\cdots,W-1 \\ k'=0,\cdots,C_l-1 \end{cases}$$

<u>敏感值反向数据关系图（多对一）</u>

前面解法原理中指出 ：这个激活值反向传播关系图（多对一）与反向敏感值关系图（多对一）是一致的，得到新图，

$$\left\{\delta_{i'-m'+p',j'-n'+q'}^{(r),(t),(k)}\right\}_{\substack{p'\in[0,M-1]\\ q'\in[0,N-1]\\ k=0,\cdots,C_r-1}} \xrightarrow{N_b\ to\ N_b} \left\{w_{M-1-p',N-1-q'}^{(r),(k),(k')}\right\}_{\substack{p'\in[0,M-1]\\ q'\in[0,N-1]\\ k=0,\cdots,C_r-1}} \xrightarrow{N_b\ to\ 1} \delta_{i',j'}^{(l),(t),(k')}$$

$$\begin{cases} i'=0,\cdots,H-1 \\ j'=0,\cdots,W-1 \\ k'=0,\cdots,C_l-1 \end{cases}$$

敏感值反向传播计算公式

由此图直接得到 l 结点分量 $x_{i',j'}^{(l),(t),(k')}$ 对应的敏感值 $\delta_{i',j'}^{(l),(t),(k')}$ 的反向传播计算公式如下（仍然是卷积公式），

$$\delta_{i',j'}^{(l),(t),(k')} = \sum_{k=0,\cdots,C_r-1} \sum_{p'\in[0,M-1]} \sum_{q'\in[0,N-1]} w_{M-1-p',N-1-q'}^{(r),(k),(k')} \cdot \delta_{i'-m'+p',j'-n'+q'}^{(r),(t),(k)}$$

$$\begin{cases} i'=0,\cdots,H-1 \\ j'=0,\cdots,W-1 \\ k'=0,\cdots,C_l-1 \end{cases}$$

以及 r 结点敏感值边缘条件 pad condition，

$$\delta_{i,j}^{(r),(t),(k)} = 0 \qquad \begin{cases} i<0 \ \ or \ \ i>H-1 \\ j<0 \ \ or \ \ j>W-1 \\ k=0,\cdots,C_r-1 \end{cases}$$

反向求敏感值公式的主要部分也为多组合卷积（multiple combination convolutions），设置新的权值参量，

$$v_{p',q'}^{(r),(k'),(k)} = w_{M-1-p',N-1-q'}^{(r),(k),(k')} \qquad \begin{cases} p'\in[0,M-1] \\ q'\in[0,N-1] \end{cases} \qquad \begin{cases} k=0,\cdots,C_r-1 \\ k'=0,\cdots,C_l-1 \end{cases}$$

敏感值 $\delta_{i',j'}^{(l),(t),(k')}$ 的反向传播计算公式修改为

$$\delta_{i',j'}^{(l),(t),(k')} = \sum_{k=0,\cdots,C_r-1} \sum_{p'\in[0,M-1]} \sum_{q'\in[0,N-1]} v_{p',q'}^{(r),(k'),(k)} \cdot \delta_{i'-m'+p',j'-n'+q'}^{(r),(t),(k)}$$

$$\begin{cases} i'=0,\cdots,H-1 \\ j'=0,\cdots,W-1 \\ k'=0,\cdots,C_l-1 \end{cases}$$

反向卷积输入输出数据

反向传播公式的主体部分也为一个多组合卷积计算，卷积的输入为第 r 结点的所有敏感值图（sensitivity maps）$\Delta_{\times H,\times W}^{(r),(t),(\times C_r)}$，

$$\Delta_{\times H,\times W}^{(r),(t),(\times C_r)}=\left\{\delta_{i,j}^{(r),(t),(k)}\right\}\Big|_{\substack{i=0,\cdots,H-1\\ j=0,\cdots,W-1\\ k=0,\cdots,C_r-1}} \qquad \begin{cases} l=1,\cdots,L & layer \\ t=0,\cdots,T-1 & time \end{cases}$$

输出为第 l 结点的一个敏感值图 $\Delta_{\times H,\times W}^{(l),(t),(k')}$，

$$\Delta_{\times H,\times W}^{(l),(t),(k')}=\left\{\delta_{i',j'}^{(l),(t),(k')}\right\}\Big|_{\substack{i'=0,\cdots,H-1\\ j'=0,\cdots,W-1}} \qquad k'=0,\cdots,C_l-1$$

反向卷积权值阵 3d 阵向量

假设，网络 l 结点第 k' 个 sensitivity map 对应的 backward filter 为 $Conv_{BF}{}^{(l),(k')}$，$k'=0,\cdots,C_l-1$，对应的的权值阵 $V_{\times M,\times N}^{(r),(k'),(\times C_r)}$ 为一个权值数据块（3d-weight tensor），可以视为一个由二维卷积窗口组成的**阵向量**（matrices in vector），卷积窗口高宽为（$M\times N$），一般可以取（3×3）、（5×5）等，但也不限定等高宽比，以及奇数尺寸，backward filter 多组合卷积的权值阵为，

$$V_{\times M,\times N}^{(r),(k'),(\times C_r)}=\left[V_{\times M,\times N}^{(r),(k'),(0)}\cdots V_{\times M,\times N}^{(r),(k'),(C_r-1)}\right]=\left\{V_{\times M,\times N}^{(r),(k'),(k)}\right\}_{k=0,\cdots,C_r-1}$$

$$=\left\{\left\{v_{p',q'}^{(r),(k'),(k)}\right\}\Big|_{\substack{p'\in[0,M-1]\\ q'\in[0,N-1]}}\right\}_{k=0,\cdots,C_r-1}=\left\{\begin{bmatrix} v_{0,0}^{(r),(k'),(k)} & \cdots & v_{0,N-1}^{(r),(k'),(k)} \\ \vdots & \ddots & \vdots \\ v_{M-1,0}^{(r),(k'),(k)} & \cdots & v_{M-1,N-1}^{(r),(k'),(k)} \end{bmatrix}_{M\times N}\right\}_{k=0,\cdots,C_r-1}$$

$$k'=0,\cdots,C_l-1$$

注记：卷积窗口 $V_{\times M,\times N}^{(r),(k'),(k)}=\left\{v_{p',q'}^{(r),(k'),(k)}\right\}\Big|_{\substack{p'\in[0,M-1]\\ q'\in[0,N-1]}}$ 中的左下标 p 表示高度游标，右下标 q 表示宽度游标，游标变化范围从 0 开始。

则有，网络 l 结点第 k' 个 sensitivity map $\Delta_{\times H,\times W}^{(l),(t),(k')}$ 的计算公式为

$$\Delta_{\times H,\times W}^{(l),(t),(k')}=Conv_{BF}{}^{(l),(k')}\left(\Delta_{\times H,\times W}^{(r),(t),(\times C_r)},V_{\times M,\times N}^{(r),(k'),(\times C_r)}\right)$$

$$k' = 0, \cdots, C_l - 1$$

反向卷积计算公式中的锚点

再回看反向敏感值卷积公式

$$\delta_{i',j'}^{(l),(t),(k')} = \sum_{k=0,\cdots,C_r-1} \sum_{p' \in [0,M-1]} \sum_{q' \in [0,N-1]} v_{p',q'}^{(r),(k'),(k)} \cdot \delta_{i'-m'+p',j'-n'+q'}^{(r),(t),(k)}$$

可知，其中的 (m', n') 为反向敏感值卷积窗口 $V_{\times M,\times N}^{(r),(k'),(k)}$ 的锚点，又有 $\begin{cases} m + m' = M - 1 \\ n + n' = N - 1 \end{cases}$，并且 (m, n) 为正向特征值卷积窗口 $W_{\times M,\times N}^{(r),(k),(k')}$ 的锚点，再考虑到参数代换 $v_{p',q'}^{(r),(k'),(k)} = w_{M-1-p',N-1-q'}^{(r),(k),(k')}$，则有 $v_{m',n'}^{(r),(k'),(k)} = w_{M-1-m',N-1-n'}^{(r),(k),(k')} = w_{m,n}^{(r),(k),(k')}$，可知，这两个锚点实际上对应的是同一个权值。

反向卷积权值阵 4d 阵中阵

网络 r 结点到 l 结点的所有的 backward filters 也为一个并行 conv 组（conv bank），记为

$$Conv_{BF}{}^{(l)} = \left\{ Conv_{BF}{}^{(l),(k')} \right\}_{k'=0,\cdots,C_l-1}$$

则以上反向传播公式也可以整合表示为，

$$\Delta_{\times H,\times W}^{(r),(t),(\times C_r)} \xrightarrow{Conv_{BF}{}^{(l)}} \begin{Bmatrix} Conv_{BF}{}^{(l),(0)} \left(\Delta_{\times H,\times W}^{(r),(t),(\times C_r)}, V_{\times M,\times N}^{(r),(0),(\times C_r)} \right) \\ \vdots \\ Conv_{BF}{}^{(l),(C_l-1)} \left(\Delta_{\times H,\times W}^{(r),(t),(\times C_r)}, V_{\times M,\times N}^{(r),(C_l-1),(\times C_r)} \right) \end{Bmatrix}_{C_l \times 1}$$

$$\rightarrow \begin{Bmatrix} \Delta_{\times H,\times W}^{(l),(t),(0)} \\ \vdots \\ \Delta_{\times H,\times W}^{(l),(t),(C_l-1)} \end{Bmatrix}_{C_l \times 1} \rightarrow \Delta_{\times H,\times W}^{(l),(t),(\times C_l)}$$

即

$$\Delta_{\times H,\times W}^{(l),(t),(\times C_l)} = Conv_{BF}{}^{(l)} \left(\Delta_{\times H,\times W}^{(r),(t),(\times C_r)}, V_{\times M,\times N}^{(r),(\times C_l),(\times C_r)} \right)$$

网络 r 结点的数据块 $\Delta_{\times H,\times W}^{(r),(t),(\times C_r)}$，经过 conv bank $Conv_{BF}{}^{(l)} = \left\{ Conv_{BF}{}^{(l),(k')} \right\}_{k'=0,\cdots,C_l-1}$，得到 l 结点的数据块 $\Delta_{\times H,\times W}^{(l),(t),(\times C_l)}$。

考虑到参数代换 $v_{p',q'}^{(r),(k'),(k)}=w_{M-1-p',N-1-q'}^{(r),(k),(k')}$，实际为平面旋转$180^{\circ}$的关系，

$$V_{\times M,\times N}^{(r),(k'),(k)}=\left[W_{\times M,\times N}^{(r),(k),(k')}\right]^{rot180^{\circ}}\qquad\begin{cases}k=0,\cdots,C_r-1\\k'=0,\cdots,C_l-1\end{cases}$$

对比前向多组合卷积阵，

$$\left\{W_{\times M,\times N}^{(r),(k),(k')}\right\}_{k'=0,\cdots,C_l-1}=\left\{\begin{bmatrix}w_{0,0}^{(r),(k),(k')}&\cdots&w_{0,N-1}^{(r),(k),(k')}\\\vdots&\ddots&\vdots\\w_{M-1,0}^{(r),(k),(k')}&\cdots&w_{M-1,N-1}^{(r),(k),(k')}\end{bmatrix}_{M\times N}\right\}_{k'=0,\cdots,C_l-1}\qquad k=0,\cdots,C_r-1$$

得到

$$\left\{V_{\times M,\times N}^{(r),(k'),(k)}\right\}_{k=0,\cdots,C_r-1}=\left\{\left[W_{\times M,\times N}^{(r),(k),(k')}\right]^{rot180^{\circ}}\right\}_{k=0,\cdots,C_r-1}\qquad k'=0,\cdots,C_l-1$$

如前所述，前向阵中阵（matrices in matrix for forward）为，

$$W_{\times M,\times N}^{(r),(\times C_r),(\times C_l)}=\left\{W_{\times M,\times N}^{(r),(k),(k')}\right\}_{\substack{k=0,\cdots,C_r-1\\k'=0,\cdots,C_l-1}}=\begin{bmatrix}W_{\times M,\times N}^{(r),(0),(\times C_l)}\\\vdots\\W_{\times M,\times N}^{(r),(C_r-1),(\times C_l)}\end{bmatrix}_{C_r}$$

$$W_{\times M,\times N}^{(r),(\times C_r),(\times C_l)}=\begin{bmatrix}W_{\times M,\times N}^{(r),(0),(0)}&\cdots&W_{\times M,\times N}^{(r),(0),(C_l-1)}\\\vdots&\ddots&\vdots\\W_{\times M,\times N}^{(r),(C_r-1),(0)}&\cdots&W_{\times M,\times N}^{(r),(C_r-1),(C_l-1)}\end{bmatrix}_{C_r\times C_l}$$

则反向阵中阵（matrices in matrix for backward）为，

$$V_{\times M,\times N}^{(r),(\times C_l),(\times C_r)}=\left\{V_{\times M,\times N}^{(r),(k'),(k)}\right\}_{\substack{k'=0,\cdots,C_l-1\\k=0,\cdots,C_r-1}}=\begin{bmatrix}V_{\times M,\times N}^{(r),(0),(\times C_r)}\\\vdots\\V_{\times M,\times N}^{(r),(C_l-1),(\times C_r)}\end{bmatrix}_{C_l}$$

$$V_{\times M,\times N}^{(r),(\times C_l),(\times C_r)}=\begin{bmatrix}V_{\times M,\times N}^{(r),(0),(0)}&\cdots&V_{\times M,\times N}^{(r),(0),(C_r-1)}\\\vdots&\ddots&\vdots\\V_{\times M,\times N}^{(r),(C_l-1),(0)}&\cdots&V_{\times M,\times N}^{(r),(C_l-1)(C_r-1)}\end{bmatrix}_{C_l\times C_r}$$

$$V_{\times M,\times N}^{(r),(\times C_l),(\times C_r)}=\left\{\left[W_{\times M,\times N}^{(r),(k),(k')}\right]^{rot180^\circ}\right\}_{\substack{k'=0,\cdots,C_l-1\\ k=0,\cdots,C_r-1}}$$

$$=\begin{bmatrix}\left[W_{\times M,\times N}^{(r),(0),(0)}\right]^{rot180^\circ} & \cdots & \left[W_{\times M,\times N}^{(r),(C_r-1),(0)}\right]^{rot180^\circ}\\ \vdots & \ddots & \vdots\\ \left[W_{\times M,\times N}^{(r),(0),(C_l-1)}\right]^{rot180^\circ} & \cdots & \left[W_{\times M,\times N}^{(r),(C_r-1),(C_l-1)}\right]^{rot180^\circ}\end{bmatrix}_{C_l\times C_r}$$

该反向阵中阵为一个权值数据块（4d-weight tensor），实际上为一个 4 维阵（4d-tensor）。每一个 $V_{\times m,\times m}^{(r),(\times C_r),(k')},k'=0,\cdots,C_l-1$ 为一个 3 维阵，对应于求解第 l 结点的一个（第 k' 个）sensitivity map，所以计算次序仍然为行优先。

注解：前向阵与反向阵恰好是大阵转置、子阵平面旋转180°（大阵转置同时各个子阵做平面旋转180°操作）。

<u>平面旋转180°举例</u>

该操作适用任何二维矩阵。

$$\begin{bmatrix}(0,0) & (0,1) & (0,2)\\ (1,0) & (1,1) & (1,2)\\ (2,0) & (2,1) & (2,2)\end{bmatrix}^{rot180^\circ}=\begin{bmatrix}(2,2) & (2,1) & (2,0)\\ (1,2) & (1,1) & (1,0)\\ (0,2) & (0,1) & (0,0)\end{bmatrix}$$

$$\begin{bmatrix}1 & 2 & 3\\ 4 & 5 & 6\\ 7 & 8 & 9\end{bmatrix}^{rot180^\circ}=\begin{bmatrix}9 & 8 & 7\\ 6 & 5 & 4\\ 3 & 2 & 1\end{bmatrix}$$

$$\begin{bmatrix}1 & 2 & 3 & 4 & 5 & 6\\ 7 & 8 & 9 & 10 & 11 & 12\\ 13 & 14 & 15 & 16 & 17 & 18\\ 19 & 20 & 21 & 22 & 23 & 24\end{bmatrix}^{rot180^\circ}=\begin{bmatrix}24 & 23 & 22 & 21 & 20 & 19\\ 18 & 17 & 16 & 15 & 14 & 13\\ 12 & 11 & 10 & 9 & 8 & 7\\ 6 & 5 & 4 & 3 & 2 & 1\end{bmatrix}$$

注解：平面旋转180°操作，可以以任何一个位置为旋转轴，其旋转结果都是一样的。所以，卷积核的锚点位置，可以认为是旋转轴的位置，这样，正向阵与反向阵的锚点权值就是一样的。并且锚点也可以是矩阵中的任何一个位置（甚至在矩阵之外），并不影响前向阵与反向阵的关系。

<u>权值梯度计算</u>

由一对一 $(1\ to\ 1)$前向传播关系图，

$$x_{i-m+p,j-n+q}^{(l),(t),(k')}\ \xrightarrow{1\ to\ 1}\ w_{p,q}^{(r),(k),(k')}\ \xrightarrow{1\ to\ 1}\ x_{i,j}^{(r),(t),(k)}$$

$$\begin{cases} p \in [0, M-1] \\ q \in [0, N-1] \end{cases} \qquad \begin{cases} k = 0, \cdots, C_r - 1 \\ k' = 0, \cdots, C_l - 1 \end{cases} \qquad \begin{cases} i = 0, \cdots, H-1 \\ j = 0, \cdots, W-1 \end{cases}$$

其直观含义为，l 结点 feature map 的输入值 $x_{i-m+p,j-n+q}^{(l),(t),(k')}$ 乘以权值 $w_{p,q}^{(r),(k),(k')}$ 向 r 结点 feature map 的输出值 $x_{i,j}^{(r),(t),(k)}$ 贡献了部分激活值。则由每一个一对一 $(1\ to\ 1)$前向传播关系图，可以得到针对权值 $\left(w_{p,q}^{(r),(k),(k')}\right)$ 的梯度 $\left.\frac{\partial J}{\partial\left(w_{p,q}^{(r),(k),(k')}\right)}\right|_{\substack{i=0,\cdots,H-1 \\ j=0,\cdots,W-1}}$，

$$\left.\frac{\partial J}{\partial\left(w_{p,q}^{(r),(k),(k')}\right)}\right|_{\substack{i=0,\cdots,H-1 \\ j=0,\cdots,W-1}} = \frac{\partial J}{\partial\left(x_{i,j}^{(r),(t),(k)}\right)} \cdot \frac{\partial\left(x_{i,j}^{(r),(t),(k)}\right)}{\partial\left(w_{p,q}^{(r),(k),(k')}\right)} = \delta_{i,j}^{(r),(t),(k)} \cdot \left(x_{i-m+p,j-n+q}^{(l),(t),(k')}\right)$$

$$\begin{cases} p \in [0, M-1] \\ q \in [0, N-1] \end{cases} \qquad \begin{cases} k = 0, \cdots, C_r - 1 \\ k' = 0, \cdots, C_l - 1 \end{cases}$$

而且，这个梯度是共用的，共用次数为 $H \times W$ 次，得到阵中阵的每一个权值 $\left(w_{p,q}^{(r),(k),(k')}\right)$ 的综合修正量，

$$\frac{\partial J}{\partial\left(w_{p,q}^{(r),(k),(k')}\right)} = \sum_{i=0,\cdots,H-1} \sum_{j=0,\cdots,W-1} \delta_{i,j}^{(r),(t),(k)} \cdot \left(x_{i-m+p,j-n+q}^{(l),(t),(k')}\right)$$

$$\begin{cases} p \in [0, M-1] \\ q \in [0, N-1] \end{cases} \qquad \begin{cases} k = 0, \cdots, C_r - 1 \\ k' = 0, \cdots, C_l - 1 \end{cases}$$

即每个权值都使用了 $H \times W$ 次。

<u>Conv 前向计算</u>

Conv 前向计算只做卷积运算，其输入数据块为网络 l 结点的激活值数据块（activation tensor）：

$$X_{\times H_l,\times W_l}^{(l),(\times T),(\times C_l)} = \left\{x_{i',j'}^{(l),(t),(k')}\right\}\Big|_{\substack{i'=0,\cdots,H_l-1 \quad t=0,\cdots,T-1 \\ j'=0,\cdots,W_l-1 \quad k'=0,\cdots,C_l-1}}$$

输出数据块为网络 r 结点的激活值数据块（activation tensor）：

$$X_{\times H_r,\times W_r}^{(r),(\times T),(\times C_r)} = \left\{x_{i,j}^{(r),(t),(k)}\right\}\Big|_{\substack{i=0,\cdots,H_r-1 \quad t=0,\cdots,T-1 \\ j=0,\cdots,W_r-1 \quad k=0,\cdots,C_r-1}}$$

参数为前向权值数据块（weight tensor）：

$$W_{\times M,\times N}^{(r),(\times C_r),(\times C_l)}=\left\{W_{\times M,\times N}^{(r),(k),(k')}\right\}_{\substack{k=0,\cdots,C_r-1\\k'=0,\cdots,C_l-1}}=\left\{w_{p,q}^{(r),(k),(k')}\right\}\Big|_{\substack{p\in[0,M-1]\\q\in[0,N-1]}}^{\substack{k=0,\cdots,C_r-1\\k'=0,\cdots,C_l-1}}$$

前向 Conv 函数表达为 $Conv_F$：

$$X_{\times H_r,\times W_r}^{(r),(\times T),(\times C_r)}=Conv_F\left(X_{\times H_l,\times W_l}^{(l),(\times T),(\times C_l)},W_{\times M,\times N}^{(r),(\times C_r),(\times C_l)}\right)$$

其中每个分量值（elements in 4d-tensor）的（并行）计算公式为：

$$x_{i,j}^{(r),(t),(k)}=\sum_{k'=0,\cdots,C_l-1}\ \sum_{p\in[0,M-1]}\ \sum_{q\in[0,N-1]}w_{p,q}^{(r),(k),(k')}\cdot x_{i-m+p,j-n+q}^{(l),(t),(k')}$$

$$\begin{cases}i=0,\cdots,H_r-1 & t=0,\cdots,T-1\\ j=0,\cdots,W_r-1 & k=0,\cdots,C_r-1\end{cases}$$

<u>Conv 权值梯度计算</u>

代价函数对前向权值数据块（weight tensor）

$$W_{\times M,\times N}^{(r),(\times C_r),(\times C_l)}=\left\{W_{\times M,\times N}^{(r),(k),(k')}\right\}_{\substack{k=0,\cdots,C_r-1\\k'=0,\cdots,C_l-1}}=\left\{w_{p,q}^{(r),(k),(k')}\right\}\Big|_{\substack{p\in[0,M-1]\\q\in[0,N-1]}}^{\substack{k=0,\cdots,C_r-1\\k'=0,\cdots,C_l-1}}$$

的每个分量求导，得到权值梯度数据块（weight gradient tensor）

$$\Delta W_{\times M,\times N}^{(r),(\times C_r),(\times C_l)}=\left\{\Delta w_{p,q}^{(r),(k),(k')}=\frac{\partial J}{\partial\left(w_{p,q}^{(r),(k),(k')}\right)}\right\}\Bigg|_{\substack{p\in[0,M-1]\\q\in[0,N-1]}}^{\substack{k=0,\cdots,C_r-1\\k'=0,\cdots,C_l-1}}$$

由前向 Conv 公式

$$x_{i,j}^{(r),(t),(k)}=\sum_{k'=0,\cdots,C_l-1}\ \sum_{p\in[0,M-1]}\ \sum_{q\in[0,N-1]}w_{p,q}^{(r),(k),(k')}\cdot x_{i-m+p,j-n+q}^{(l),(t),(k')}$$

$$\begin{cases}i=0,\cdots,H_r-1 & t=0,\cdots,T-1\\ j=0,\cdots,W_r-1 & k=0,\cdots,C_r-1\end{cases}$$

可知，如果固定 (i,j,t)，则权值梯度由单一式子确定，

$$\Delta w_{p,q}^{(r),(k),(k')}=\frac{\partial J}{\partial\left(w_{p,q}^{(r),(k),(k')}\right)}=\frac{\partial J}{\partial\left(x_{i,j}^{(r),(t),(k)}\right)}\cdot\frac{\partial\left(x_{i,j}^{(r),(t),(k)}\right)}{\partial\left(w_{p,q}^{(r),(k),(k')}\right)}=\delta_{i,j}^{(r),(t),(k)}\cdot\left(x_{i-m+p,j-n+q}^{(l),(t),(k')}\right)$$

$$\begin{cases} p \in [0, M-1] \\ q \in [0, N-1] \end{cases} \qquad \begin{cases} k = 0, \cdots, C_r - 1 \\ k' = 0, \cdots, C_l - 1 \end{cases}$$

再考虑到卷积核在 $\begin{cases} i = 0, \cdots, H_r - 1 \\ j = 0, \cdots, W_r - 1 \end{cases}$ 范围共用，及 mini-batch 批量 T，权值梯度合成如下，

$$\Delta w_{p,q}^{(r),(k),(k')} = \sum_{t=0,\cdots,T-1} \sum_{i=0,\cdots,H_r-1} \sum_{j=0,\cdots,W_r-1} \delta_{i,j}^{(r),(t),(k)} \cdot \left(x_{i-m+p,j-n+q}^{(l),(t),(k')} \right)$$

$$\begin{cases} p \in [0, M-1] \\ q \in [0, N-1] \end{cases} \qquad \begin{cases} k = 0, \cdots, C_r - 1 \\ k' = 0, \cdots, C_l - 1 \end{cases}$$

Conv 权值梯度计算函数表达为 $Conv_{grad}$：

$$\Delta W_{\times M,\times N}^{(r),(\times C_r),(\times C_l)} = Conv_{grad} \left(\Delta_{\times H_r,\times W_r}^{(r),(\times T),(\times C_r)}, X_{\times H_l,\times W_l}^{(l),(\times T),(\times C_l)} \right)$$

一部分输入数据块为网络 r 结点的敏感值数据块（sensitivity tensor）：

$$\Delta_{\times H_r,\times W_r}^{(r),(\times T),(\times C_r)} = \left\{ \delta_{i,j}^{(r),(t),(k)} \right\} \Big|_{\substack{i=0,\cdots,H_r-1 \quad t=0,\cdots,T-1 \\ j=0,\cdots,W_r-1 \quad k=0,\cdots,C_r-1}}$$

另一部分输入数据块为网络 l 结点的激活值数据块（activation tensor）：

$$X_{\times H_l,\times W_l}^{(l),(\times T),(\times C_l)} = \left\{ x_{i',j'}^{(l),(t),(k')} \right\} \Big|_{\substack{i'=0,\cdots,H_l-1 \quad t=0,\cdots,T-1 \\ j'=0,\cdots,W_l-1 \quad k'=0,\cdots,C_l-1}}$$

输出为权值梯度数据块（weight gradient tensor）

$$\Delta W_{\times M,\times N}^{(r),(\times C_r),(\times C_l)} = \left\{ \Delta w_{p,q}^{(r),(k),(k')} \right\} \Big|_{\substack{p\in[0,M-1] \quad k=0,\cdots,C_r-1 \\ q\in[0,N-1] \quad k'=0,\cdots,C_l-1}}$$

Conv 反向计算

如果激活值数据块 Conv 前向计算只做卷积运算，则其敏感值数据块反向传播的计算也只做卷积计算。事实上，只做卷积的前向运算相比较于一般化的 $f(\sum xw + b)$ 神经元运算，相当于 $\begin{cases} b = 0 \\ f(x) = x \end{cases}$，则敏感值反向传播计算公式中就只有卷积部分。

Conv 反向计算也只做卷积运算，其输入数据块为网络 r 结点的敏感值数据块（sensitivity tensor）：

$$\Delta_{\times H_r,\times W_r}^{(r),(\times T),(\times C_r)} = \left\{ \delta_{i,j}^{(r),(t),(k)} \right\} \Big|_{\substack{i=0,\cdots,H_r-1 \quad t=0,\cdots,T-1 \\ j=0,\cdots,W_r-1 \quad k=0,\cdots,C_r-1}}$$

输出数据块为网络 l 结点的敏感值数据块（sensitivity tensor）：

$$\Delta_{\times H_l,\times W_l}^{(l),(\times T),(\times C_l)}=\left\{\delta_{i',j'}^{(l),(t),(k')}\right\}\Big|_{\substack{i'=0,\cdots,H_l-1 \quad t=0,\cdots,T-1\\ j'=0,\cdots,W_l-1 \quad k'=0,\cdots,C_l-1}}$$

参数为反向权值数据块（weight tensor）：

$$V_{\times M,\times N}^{(r),(\times C_l),(\times C_r)}=\left\{V_{\times M,\times N}^{(r),(k'),(k)}\right\}_{\substack{k'=0,\cdots,C_l-1\\ k=0,\cdots,C_r-1}}=\left\{v_{p',q'}^{(r),(k'),(k)}\right\}\Big|_{\substack{p'\in[0,M-1] \quad k'=0,\cdots,C_l-1\\ q'\in[0,N-1] \quad k=0,\cdots,C_r-1}}$$

反向 Conv 函数表达为 $Conv_{BF}$：

$$\Delta_{\times H_l,\times W_l}^{(l),(\times T),(\times C_l)}=Conv_{BF}\left(\Delta_{\times H_r,\times W_r}^{(r),(\times T),(\times C_r)},V_{\times M,\times N}^{(r),(\times C_l),(\times C_r)}\right)$$

其中每个分量值（elements in 4d-tensor）的（并行）计算公式为：

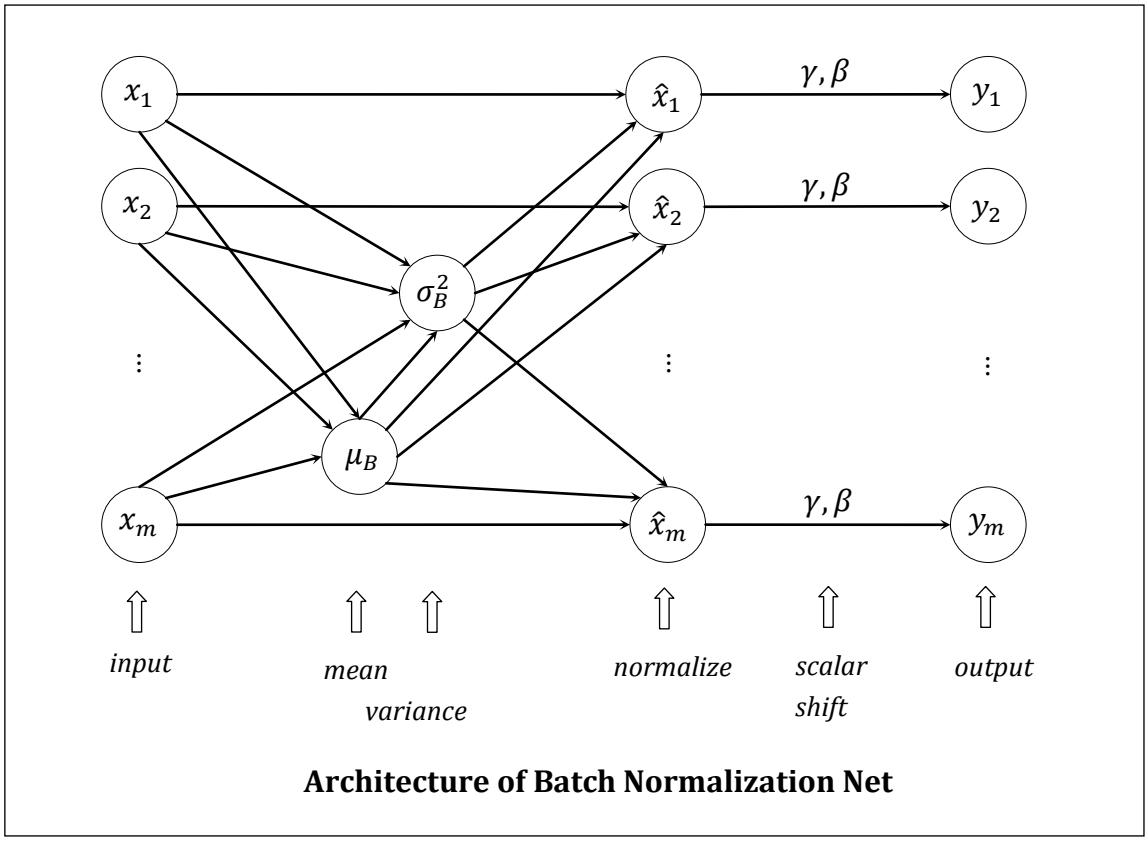
$$\delta_{i',j'}^{(l),(t),(k')}=\sum_{k=0,\cdots,C_r-1}\ \sum_{p'\in[0,M-1]}\ \sum_{q'\in[0,N-1]}v_{p',q'}^{(r),(k'),(k)}\cdot\delta_{i'-m'+p',j'-n'+q'}^{(r),(t),(k)}$$

$$\begin{cases}i'=0,\cdots,H_l-1 & t=0,\cdots,T-1\\ j'=0,\cdots,W_l-1 & k'=0,\cdots,C_l-1\end{cases}$$

Batch Normalization

BN 模块的网络结构

将 BN（Batch Normalization）算法看做一个网络模块（BN Block），其输入为某个神经元的 mini-batch 输出，有 m 个值，作为 BN 输入，记为 $\{x_1, \cdots, x_m\}$，BN Block 对此输入做 4 步（层）操作，分别为：求均值 μ_B、求方差 σ_B^2、归一化得到 $\{\hat{x}_1, \cdots, \hat{x}_m\}$、再伸缩平移，得到 BN 输出 $\{y_1, \cdots, y_m\}$，同样有 m 个值。



Architecture of Batch Normalization Net

1 mini-batch mean

$$\mu_B \leftarrow \frac{1}{m} \cdot \sum_{i=1}^{m} x_i$$

2 mini-batch variance

$$\sigma_B^2 \leftarrow \frac{1}{m} \cdot \sum_{i=1}^{m} (x_i - \mu_B)^2$$

3 normalize

$$\hat{x}_i \leftarrow \frac{x_i - \mu_B}{\sqrt{\sigma_B^2 + \epsilon}}$$

4 scale and shift

$$y_i \leftarrow \gamma \hat{x}_i + \beta \equiv BN_{\gamma,\beta}(x_i)$$

其中 (γ, β) 为可学习参数，ϵ 为方差修正量。BN Block 的网络结构图如下。

BN forward formula (copy from original paper)

$$\begin{cases} y_i = \gamma \hat{x}_i + \beta & \hat{x}_i = \dfrac{x_i - \mu_B}{\sqrt{\sigma_B^2 + \epsilon}} \quad i = 1, \cdots, m \\ \sigma_B^2 = \dfrac{1}{m} \cdot \sum_{i=1}^{m} (x_i - \mu_B)^2 & \mu_B = \dfrac{1}{m} \cdot \sum_{i=1}^{m} x_i \end{cases}$$

BN backward formula

反向梯度计算的输入为 $\{\frac{\partial \ell}{\partial y_1}, \cdots, \frac{\partial \ell}{\partial y_m}\}$，依据 BN 子网络结构图、前向公式及反向 chain rules，可以直接写出各个梯度公式：

(1) γ 权值在前向公式 $y_i = \gamma \hat{x}_i + \beta,\ i = 1, \cdots, m$ 中使用了 m 次，则梯度 $\frac{\partial \ell}{\partial \gamma}$ 的计算为和式，

$$\frac{\partial \ell}{\partial \gamma} = \sum_{i=1}^{m} \frac{\partial \ell}{\partial y_i} \cdot \hat{x}_i$$

(2) β 权值也在前向公式 $y_i = \gamma \hat{x}_i + \beta,\ i = 1, \cdots, m$ 中使用了 m 次，则梯度 $\frac{\partial \ell}{\partial \beta}$ 的计算也为和式，

$$\frac{\partial \ell}{\partial \beta} = \sum_{i=1}^{m} \frac{\partial \ell}{\partial y_i}$$

(3) 在前向公式 $y_i = \gamma \hat{x}_i + \beta,\ i = 1, \cdots, m$ 中 $\hat{x}_i$ 与 y_i 直接对应，则梯度 $\frac{\partial \ell}{\partial \hat{x}_i}$ 的计算为，

$$\frac{\partial \ell}{\partial \hat{x}_i} = \frac{\partial \ell}{\partial y_i} \cdot \gamma \qquad i = 1, \cdots, m$$

(4) 在前向公式 $\hat{x}_i = \frac{x_i - \mu_B}{\sqrt{\sigma_B^2 + \epsilon}},\ i = 1, \cdots, m$ 中 σ_B^2 与 $\{\hat{x}_1, \cdots, \hat{x}_m\}$ 对应，即 σ_B^2 也使用了 m 次，则梯度 $\frac{\partial \ell}{\partial \sigma_B^2}$ 的计算也为和式，

$$\frac{\partial \ell}{\partial \sigma_B^2} = \sum_{i=1}^{m} \frac{\partial \ell}{\partial \hat{x}_i} \cdot \frac{\partial \hat{x}_i}{\partial \sigma_B^2} = \sum_{i=1}^{m} \frac{\partial \ell}{\partial \hat{x}_i} \cdot (x_i - \mu_B) \cdot \left(-\frac{1}{2}\right) \cdot (\sigma_B^2 + \epsilon)^{-\frac{3}{2}}$$

(5) 在前向公式 $\hat{x}_i=\frac{x_i-\mu_B}{\sqrt{\sigma_B^2+\epsilon}},\ i=1,\cdots,m$ 中 μ_B 与 $\{\hat{x}_1,\cdots,\hat{x}_m\}$ 对应，即 μ_B 也使用了 m 次，同时，在前向公式 $\sigma_B^2=\frac{1}{m}\cdot\sum_{i=1}^{m}(x_i-\mu_B)^2$ 中 μ_B 还与 σ_B^2 对应，则梯度 $\frac{\partial\ell}{\partial\mu_B}$ 的计算分为 2 部分，

$$\frac{\partial\ell}{\partial\mu_B}=\left(\sum_{i=1}^{m}\frac{\partial\ell}{\partial\hat{x}_i}\cdot\frac{\partial\hat{x}_i}{\partial\mu_B}\right)+\left(\frac{\partial\ell}{\partial\sigma_B^2}\cdot\frac{\partial\sigma_B^2}{\partial\mu_B}\right)=\left(\sum_{i=1}^{m}\frac{\partial\ell}{\partial\hat{x}_i}\cdot\frac{-1}{\sqrt{\sigma_B^2+\epsilon}}\right)+\frac{\partial\ell}{\partial\sigma_B^2}\cdot\left(\frac{-2}{m}\cdot\sum_{j=1}^{m}(x_j-\mu_B)\right)$$

(6) 对于某个特定的 i，在前向公式 $\hat{x}_i=\frac{x_i-\mu_B}{\sqrt{\sigma_B^2+\epsilon}},\ i=1,\cdots,m$ 中 x_i 与 $\hat{x}_i$ 一一对应，又在前向公式 $\sigma_B^2=\frac{1}{m}\cdot\sum_{i=1}^{m}(x_i-\mu_B)^2$ 中 x_i 还与 σ_B^2 对应，还有在前向公式 $\mu_B=\frac{1}{m}\cdot\sum_{i=1}^{m}x_i$ 中 x_i 还与 μ_B 对应，则梯度 $\frac{\partial\ell}{\partial x_i}$ 的计算分为 3 部分，

$$\frac{\partial\ell}{\partial x_i}=\left(\frac{\partial\ell}{\partial\hat{x}_i}\cdot\frac{\partial\hat{x}_i}{\partial x_i}\right)+\left(\frac{\partial\ell}{\partial\sigma_B^2}\cdot\frac{\partial\sigma_B^2}{\partial x_i}\right)+\left(\frac{\partial\ell}{\partial\mu_B}\cdot\frac{\partial\mu_B}{\partial x_i}\right)\qquad i=1,\cdots,m$$

$$\frac{\partial\ell}{\partial x_i}=\left(\frac{\partial\ell}{\partial\hat{x}_i}\cdot\frac{1}{\sqrt{\sigma_B^2+\epsilon}}\right)+\left(\frac{\partial\ell}{\partial\sigma_B^2}\cdot\frac{2(x_i-\mu_B)}{m}\right)+\left(\frac{1}{m}\cdot\frac{\partial\ell}{\partial\mu_B}\right)\qquad i=1,\cdots,m$$

<u>BN 前向算法</u>

BN 算法的基本思想是，在 (t) 维度上以 mini-batch 为统计单位，对数据块中 (i,j,k) 维度的每个分量值，独立地进行归一化，使其均值为 0，方差为 1。可以形象地称为“一条线”算法。

BN 算法的基本操作针对的数据单位是一个 T 维向量（1d-tensor），T 个数据组成一个 mini-batch 向量，

$$X_{i,j}^{(l),(\times T),(k)}=\left\{x_{i,j}^{(l),(t),(k)}\right\}\Big|_{t=0,\cdots,T-1}\qquad\begin{cases}l=1,\cdots,L & layer\\ i=0,\cdots,H_l-1 & height\\ j=0,\cdots,W_l-1 & width\\ k=0,\cdots,C_l-1 & channel\end{cases}$$

BN 算法可以分为两个步骤，（1）归一化 normalize 层，（2）缩放平移 scale-shift 层，对 1d-tensor 而言，

$$X_{i,j}^{(l),(\times T),(k)}\ \xrightarrow{norm}\ \hat{X}_{i,j}^{(l),(\times T),(k)}\ \xrightarrow{scale-shift}\ Y_{i,j}^{(l),(\times T),(k)}$$

$$\left\{x_{i,j}^{(l),(t),(k)}\right\}\Big|_{t=0,\cdots,T-1}\ \xrightarrow{norm}\ \left\{\hat{x}_{i,j}^{(l),(t),(k)}\right\}\Big|_{t=0,\cdots,T-1}\ \xrightarrow{scale-shift}\ \left\{y_{i,j}^{(l),(t),(k)}\right\}\Big|_{t=0,\cdots,T-1}$$

对 4d-tensor 而言，

$$X_{\times H_l,\times W_l}^{(l),(\times T),(\times C_l)} \xrightarrow{norm} \hat{X}_{\times H_l,\times W_l}^{(l),(\times T),(\times C_l)} \xrightarrow{scale-shift} Y_{\times H_l,\times W_l}^{(l),(\times T),(\times C_l)}$$

$$\left\{x_{i,j}^{(l),(t),(k)}\right\}\Big|_{\substack{i=0,\cdots,H_l-1\\ j=0,\cdots,W_l-1\\ k=0,\cdots,C_l-1\\ t=0,\cdots,T-1}} \xrightarrow{norm} \left\{\hat{x}_{i,j}^{(l),(t),(k)}\right\}\Big|_{\substack{i=0,\cdots,H_l-1\\ j=0,\cdots,W_l-1\\ k=0,\cdots,C_l-1\\ t=0,\cdots,T-1}} \xrightarrow{scale-shift} \left\{y_{i,j}^{(l),(t),(k)}\right\}\Big|_{\substack{i=0,\cdots,H_l-1\\ j=0,\cdots,W_l-1\\ k=0,\cdots,C_l-1\\ t=0,\cdots,T-1}}$$

（1）归一化 normalize 层，

1 mini-batch mean

$$\mu_{i,j}^{(l),(*),(k)} = \frac{1}{T}\cdot\sum_{t=0}^{T-1} x_{i,j}^{(l),(t),(k)}$$

2 mini-batch variance

$$\left(\sigma_{i,j}^{(l),(*),(k)}\right)^2 = \frac{1}{T}\cdot\sum_{t=0}^{T-1}\left(x_{i,j}^{(l),(t),(k)} - \mu_{i,j}^{(l),(*),(k)}\right)^2$$

3 normalize

$$\hat{x}_{i,j}^{(l),(t),(k)} = \frac{x_{i,j}^{(l),(t),(k)} - \mu_{i,j}^{(l),(*),(k)}}{\sqrt{\left(\sigma_{i,j}^{(l),(*),(k)}\right)^2 + \epsilon}} \qquad t = 0,\cdots,T-1$$

输出仍是一个 T 维向量，这一部分不含有可学习参数。

（2）缩放平移 scale-shift 层，

$$y_{i,j}^{(l),(t),(k)} = \gamma_{i,j}^{(l),(*),(k)}\cdot\hat{x}_{i,j}^{(l),(t),(k)} + \beta_{i,j}^{(l),(*),(k)} \qquad t = 0,\cdots,T-1$$

输出仍是一个 T 维向量，这一部分含有可学习参数 $\left(\gamma_{i,j}^{(l),(*),(k)}, \beta_{i,j}^{(l),(*),(k)}\right)$。

BN 反向算法

设 ℓ 为 loss function，

$$\frac{\partial \ell}{\partial \left(y_{i,j}^{(l),(t),(k)}\right)} \qquad t = 0, \cdots, T-1$$

为已知，是 BN 反向传播输入量，从后层反向传播而来。对缩放平移 scale-shift 层，则有，

(1)

$$\frac{\partial \ell}{\partial \left(\gamma_{i,j}^{(l),(*),(k)}\right)} = \sum_{t=0}^{T-1} \frac{\partial \ell}{\partial \left(y_{i,j}^{(l),(t),(k)}\right)} \cdot \hat{x}_{i,j}^{(l),(t),(k)}$$

(2)

$$\frac{\partial \ell}{\partial \left(\beta_{i,j}^{(l),(*),(k)}\right)} = \sum_{t=0}^{T-1} \frac{\partial \ell}{\partial \left(y_{i,j}^{(l),(t),(k)}\right)}$$

(1) (2) 为可学习参数 $\left(\gamma_{i,j}^{(l),(*),(k)}, \beta_{i,j}^{(l),(*),(k)}\right)$ 的修正量。

对归一化 normalize 层，

(3)

$$\frac{\partial \ell}{\partial \left(\hat{x}_{i,j}^{(l),(t),(k)}\right)} = \frac{\partial \ell}{\partial \left(y_{i,j}^{(l),(t),(k)}\right)} \cdot \gamma_{i,j}^{(l),(*),(k)} \qquad t = 0, \cdots, T-1$$

(4)

$$\frac{\partial \ell}{\partial \left(\sigma_{i,j}^{(l),(*),(k)}\right)^2} = \sum_{t=0}^{T-1} \frac{\partial \ell}{\partial \left(\hat{x}_{i,j}^{(l),(t),(k)}\right)} \cdot \frac{\partial \left(\hat{x}_{i,j}^{(l),(t),(k)}\right)}{\partial \left(\sigma_{i,j}^{(l),(*),(k)}\right)^2}$$

$$= \sum_{t=0}^{T-1} \frac{\partial \ell}{\partial \left(\hat{x}_{i,j}^{(l),(t),(k)}\right)} \cdot \left(x_{i,j}^{(l),(t),(k)} - \mu_{i,j}^{(l),(*),(k)}\right) \cdot \left(-\frac{1}{2}\right) \cdot \left(\left(\sigma_{i,j}^{(l),(*),(k)}\right)^2 + \epsilon\right)^{-\frac{3}{2}}$$

(5)

$$\frac{\partial \ell}{\partial \left(\mu_{i,j}^{(l),(*),(k)}\right)} = \sum_{t=0}^{T-1} \frac{\partial \ell}{\partial \left(\hat{x}_{i,j}^{(l),(t),(k)}\right)} \cdot \frac{\partial \left(\hat{x}_{i,j}^{(l),(t),(k)}\right)}{\partial \left(\mu_{i,j}^{(l),(*),(k)}\right)}$$

$$= \left(\sum_{t=0}^{T-1} \frac{\partial \ell}{\partial \left(\hat{x}_{i,j}^{(l),(t),(k)}\right)} \cdot \frac{-1}{\sqrt{\left(\sigma_{i,j}^{(l),(*),(k)}\right)^2 + \epsilon}} \right) +$$

$$+ \frac{\partial \ell}{\partial \left(\sigma_{i,j}^{(l),(*),(k)}\right)^2} \cdot \left(\frac{-2}{T} \cdot \sum_{t=0}^{T-1} \left(x_{i,j}^{(l),(t),(k)} - \mu_{i,j}^{(l),(*),(k)}\right) \right)$$

(6)

$$\frac{\partial \ell}{\partial \left(x_{i,j}^{(l),(t),(k)}\right)} = \frac{\partial \ell}{\partial \left(\hat{x}_{i,j}^{(l),(t),(k)}\right)} \cdot \frac{1}{\sqrt{\left(\sigma_{i,j}^{(l),(*),(k)}\right)^2 + \epsilon}} +$$

$$+ \frac{\partial \ell}{\partial \left(\sigma_{i,j}^{(l),(*),(k)}\right)^2} \cdot \frac{2\left(x_{i,j}^{(l),(t),(k)} - \mu_{i,j}^{(l),(*),(k)}\right)}{T} + \frac{1}{T} \cdot \frac{\partial \ell}{\partial \left(\mu_{i,j}^{(l),(*),(k)}\right)}$$

$$t = 0, \cdots, T-1$$

(3)(4)(5)为中间量，(6) $\frac{\partial \ell}{\partial \left(x_{i,j}^{(l),(t),(k)}\right)}$ 为归一化 normalize 层的反向传播输出量，传递给前面层次。

Rectified Linear Unit

ReLU 前向计算

ReLU（Rectified Linear Unit）函数定义如下：

$$y = f(x) = \max(0, x)$$

假设输入数据块为网络 l 结点的激活值数据块（activation tensor）：

$$X_{\times H_l,\times W_l}^{(l),(\times T),(\times C_l)} = \left\{x_{i,j}^{(l),(t),(k)}\right\}\Big|_{\substack{i=0,\cdots,H_l-1 \quad t=0,\cdots,T-1 \\ j=0,\cdots,W_l-1 \quad k=0,\cdots,C_l-1}}$$

则输出数据块仍为网络 l 结点的激活值数据块（activation tensor）：

$$Y_{\times H_l,\times W_l}^{(l),(\times T),(\times C_l)} = \left\{y_{i,j}^{(l),(t),(k)}\right\}\Big|_{\substack{i=0,\cdots,H_l-1 \quad t=0,\cdots,T-1 \\ j=0,\cdots,W_l-1 \quad k=0,\cdots,C_l-1}}$$

前向 ReLU 函数表达为 $ReLU_F$：

$$Y_{\times H_l,\times W_l}^{(l),(\times T),(\times C_l)} = ReLU_F\left(X_{\times H_l,\times W_l}^{(l),(\times T),(\times C_l)}\right)$$

其中每个分量值（elements in 4d-tensor）的（并行）计算公式为：

$$y_{i,j}^{(l),(t),(k)} = max\left(0, x_{i,j}^{(l),(t),(k)}\right) \qquad \begin{cases} i = 0,\cdots,H_l-1 & t = 0,\cdots,T-1 \\ j = 0,\cdots,W_l-1 & k = 0,\cdots,C_l-1 \end{cases}$$

ReLU 反向计算

ReLU（Rectified Linear Unit）函数为 $y = f(x) = \max(0, x)$，则敏感值公式如下：

$$\frac{\partial J}{\partial x} = \frac{\partial J}{\partial y}\cdot\frac{\partial y}{\partial x} = \frac{\partial J}{\partial y}\cdot\frac{\partial y}{\partial x} = \begin{cases} \dfrac{\partial J}{\partial y} & x > 0 \\ 0 & x \le 0 \end{cases}$$

假设输入数据块为网络 l 结点的敏感值数据块（sensitivity tensor）：

$$\Delta Y_{\times H_l,\times W_l}^{(l),(\times T),(\times C_l)} = \left\{\delta y_{i,j}^{(l),(t),(k)}\right\}\Big|_{\substack{i=0,\cdots,H_l-1 \quad t=0,\cdots,T-1 \\ j=0,\cdots,W_l-1 \quad k=0,\cdots,C_l-1}}$$

则输出数据块仍为网络 l 结点的敏感值数据块（sensitivity tensor）：

$$\Delta X_{\times H_l,\times W_l}^{(l),(\times T),(\times C_l)} = \left\{\delta x_{i,j}^{(l),(t),(k)}\right\}\Big|_{\substack{i=0,\cdots,H_l-1 \quad t=0,\cdots,T-1 \\ j=0,\cdots,W_l-1 \quad k=0,\cdots,C_l-1}}$$

反向 ReLU 函数表达为 $ReLU_{BF}$：

$$\Delta X_{\times H_l,\times W_l}^{(l),(\times T),(\times C_l)} = ReLU_{BF}\left(\Delta Y_{\times H_l,\times W_l}^{(l),(\times T),(\times C_l)}\right)$$

其中每个分量值（elements in 4d-tensor）的（并行）计算公式为：

$$\delta x_{i,j}^{(l),(t),(k)} = \begin{cases} \delta y_{i,j}^{(l),(t),(k)} & x > 0 \\ 0 & x \le 0 \end{cases} \qquad \begin{cases} i = 0,\cdots,H_l-1 & t = 0,\cdots,T-1 \\ j = 0,\cdots,W_l-1 & k = 0,\cdots,C_l-1 \end{cases}$$

Pooling Operate

Pooling 前向计算

Pooling 前向计算，对激活值做局部竞争，胜出者其值赋给输出数据块的相应位置，其特点是特征图尺寸缩减，并且特征点移位，其效果是解像度降低，次要特征被忽略。

Pooling 前向计算，其输入数据块为网络 l 结点的激活值数据块（activation tensor）：

$$X_{\times H_l,\times W_l}^{(l),(\times T),(\times C_l)} = \left\{ x_{i',j'}^{(l),(t),(k')} \right\} \Big|_{\substack{i'=0,\cdots,H_l-1 \quad t=0,\cdots,T-1 \\ j'=0,\cdots,W_l-1 \quad k'=0,\cdots,C_l-1}}$$

对输入数据块在 (i',j') 维度执行池化操作，输出数据块为网络 r 结点的激活值数据块（activation tensor）：

$$X_{\times H_r,\times W_r}^{(r),(\times T),(\times C_r)} = \left\{ x_{i,j}^{(r),(t),(k)} \right\} \Big|_{\substack{i=0,\cdots,H_r-1 \quad t=0,\cdots,T-1 \\ j=0,\cdots,W_r-1 \quad k=0,\cdots,C_r-1}}$$

注意到，两个数据块的通道数应该是一样的，$C_l = C_r$。

前向 Pooling 函数表达为 $Pooling_F$：

$$X_{\times H_r,\times W_r}^{(r),(\times T),(\times C_r)} = Pooling_F\left(X_{\times H_l,\times W_l}^{(l),(\times T),(\times C_l)} \right)$$

池化操作从输入图的一个池内的各个分量中选出一个最大值（max pooling），对应输出到输出图中的一个分量，定标方式以输入图中每个池的左上角在输入图中的坐标 (i',j') 为基准点（base point），池内游标为 (a,b)，

$$\left\{ \ (a,b) \ \middle| \ \begin{cases} a = 0,\cdots,(p_h-1) \\ b = 0,\cdots,(p_w-1) \end{cases} \right\}$$

其中 $(p_h \times p_w)$ 为池的尺寸（height & width），池的移动步长（stride）为 $\{s_h, s_w\}$，输出数据块 $X_{\times H_r,\times W_r}^{(r),(\times T),(\times C_r)}$ 中每个分量值（elements in 4d-tensor）的（并行）计算公式为：

$$x_{i,j}^{(r),(t),(k)} = max\left\{ x_{i\times s_h+a,j\times s_w+b}^{(l),(t),(k)} \right\}_{\substack{a=0,\cdots,(p_h-1) \\ b=0,\cdots,(p_w-1)}} \qquad \begin{cases} i = 0,\cdots,H_r-1 & t = 0,\cdots,T-1 \\ j = 0,\cdots,W_r-1 & k = 0,\cdots,C_r-1 \end{cases}$$

$$\begin{cases} 0 \le (i \times s_h + a) \le H_l - 1 \\ 0 \le (j \times s_w + b) \le W_l - 1 \end{cases}$$

这是个一般形式的表达式，池之间可以交叠（overlap），也可以有间隔。通常取 $(p_h \times p_w) = (2 \times 2)$，$\{s_h, s_w\} = \{2,2\}$，池之间不交叠且无间隔，则（并行）计算公式为：

$$x_{i,j}^{(r),(t),(k)} = max\left\{x_{i\times 2+a,j\times 2+b}^{(l),(t),(k)}\right\}_{\substack{a=0,1\\b=0,1}} \qquad \begin{cases} i=0,\cdots,H_r-1 & t=0,\cdots,T-1 \\ j=0,\cdots,W_r-1 & k=0,\cdots,C_r-1 \end{cases}$$

$$\begin{cases} 0 \le (i\times 2+a) \le H_l-1 \\ 0 \le (j\times 2+b) \le W_l-1 \end{cases}$$

在做池化的同时，还需要求出一个与输出数据块 $X_{\times H_r,\times W_r}^{(r),(\times T),(\times C_r)}$ 同维度同大小的溯源标记数据块（path tensor），

$$P_{\times H_r,\times W_r}^{(r),(\times T),(\times C_r)} = \left\{p_{i,j}^{(r),(t),(k)}\right\}\Big|_{\substack{i=0,\cdots,H_r-1 \quad t=0,\cdots,T-1\\ j=0,\cdots,W_r-1 \quad k=0,\cdots,C_r-1}}$$

其中，$p_{i,j}^{(r),(t),(k)}$ 为池化胜出值 $x_{i',j'}^{(l),(t),(k)}$ 在输入数据块 $X_{\times H_l,\times W_l}^{(l),(\times T),(\times C_l)}$ 中的坐标 (i',j')，

$$p_{i,j}^{(r),(t),(k)} = (i',j') \mid \left\{x_{i',j'}^{(l),(t),(k)} = max\left\{x_{i\times s_h+a,j\times s_w+b}^{(l),(t),(k)}\right\}_{\substack{a=0,\cdots,(p_h-1)\\b=0,\cdots,(p_w-1)}}\right\}$$

$$\begin{cases} i=0,\cdots,H_r-1 & t=0,\cdots,T-1 \\ j=0,\cdots,W_r-1 & k=0,\cdots,C_r-1 \end{cases}$$

此函数记为前向 PoolingPath 函数，表达为 $PoolingPath_F$：

$$P_{\times H_r,\times W_r}^{(r),(\times T),(\times C_r)} = PoolingPath_F\left(X_{\times H_l,\times W_l}^{(l),(\times T),(\times C_l)}, X_{\times H_r,\times W_r}^{(r),(\times T),(\times C_r)}\right)$$

Pooling 反向计算

Pooling 反向计算，其输入数据块为网络 r 结点的敏感值数据块（sensitivity tensor）：

$$\Delta_{\times H_r,\times W_r}^{(r),(\times T),(\times C_r)} = \left\{\delta_{i,j}^{(r),(t),(k)}\right\}\Big|_{\substack{i=0,\cdots,H_r-1 \quad t=0,\cdots,T-1\\ j=0,\cdots,W_r-1 \quad k=0,\cdots,C_r-1}}$$

输出数据块为网络 l 结点的敏感值数据块（sensitivity tensor）：

$$\Delta_{\times H_l,\times W_l}^{(l),(\times T),(\times C_l)} = \left\{\delta_{i',j'}^{(l),(t),(k')}\right\}\Big|_{\substack{i'=0,\cdots,H_l-1 \quad t=0,\cdots,T-1\\ j'=0,\cdots,W_l-1 \quad k'=0,\cdots,C_l-1}}$$

注意到，两个数据块的通道数应该是一样的，$C_l = C_r$。

需要的参数为溯源标记数据块（path tensor）：

$$P_{\times H_r,\times W_r}^{(r),(\times T),(\times C_r)} = \left\{p_{i,j}^{(r),(t),(k)}\right\}\Big|_{\substack{i=0,\cdots,H_r-1 \quad t=0,\cdots,T-1 \\ j=0,\cdots,W_r-1 \quad k=0,\cdots,C_r-1}}$$

在 Pooling 操作模块进行反向敏感值投送的函数记为 $Pooling_{BF}$，

$$\Delta_{\times H_l,\times W_l}^{(l),(\times T),(\times C_l)} = Pooling_{BF}\left(\Delta_{\times H_r,\times W_r}^{(r),(\times T),(\times C_r)}, P_{\times H_r,\times W_r}^{(r),(\times T),(\times C_r)}\right)$$

反向计算的步骤分为两步：

（1）先将输出数据块 $\Delta_{\times H_l,\times W_l}^{(l),(\times T),(\times C_l)}$ 的每个分量清零，

$$\delta_{i',j'}^{(l),(t),(k')} = 0 \qquad \begin{cases} i' = 0,\cdots,H_l-1 & t = 0,\cdots,T-1 \\ j' = 0,\cdots,W_l-1 & k' = 0,\cdots,C_l-1 \end{cases}$$

（2）将 r 结点输入数据块的每个分量 $\delta_{i,j}^{(r),(t),(k)}$ 依据溯源标记 $\left\{p_{i,j}^{(r),(t),(k)} = (i',j')\right\}$ 投射给 l 结点输出数据块的对应分量 $\delta_{i',j'}^{(l),(t),(k')}$，

$$\delta_{i',j'}^{(l),(t),(k')} \Leftarrow \delta_{i,j}^{(r),(t),(k)}$$

$$where \quad \begin{cases} (i',j') = p_{i,j}^{(r),(t),(k)} \\ k' = k \end{cases} \qquad for \quad \begin{cases} i' = 0,\cdots,H_l-1 & t = 0,\cdots,T-1 \\ j' = 0,\cdots,W_l-1 & k' = 0,\cdots,C_l-1 \end{cases}$$

如果前向池化运算的池之间有交叠，则前向池化时就有可能有 l 结点的某个局部最大激活值被传送给 r 结点中多于一个的位置及激活值，那么，反向敏感值则有可能出现重叠投射的情况，即在溯源标记式中 $\left\{(i',j') = p_{i,j}^{(r),(t),(k)}\right\}$ 会出现多个 (i,j) 对应同一个 (i',j') 的情况。对于这种情境，串行编程不会有什么问题，直接覆盖或比较大小后覆盖均可，如果是并行编程则可能出现写入操作冲突，造成不可预测的结果，需要注意这个问题。池非交叠的前向池化则不会有这个问题。

Full Connected Layer

Full 前向计算

Full 前向计算为全连接（full connected layer）网络计算，其输入数据块为网络 l 结点的激活值数据块（activation tensor）：

$$X_{\times H_l,\times W_l}^{(l),(\times T),(\times C_l)}=\left\{x_{i',j'}^{(l),(t),(k')}\right\}\Big|_{\substack{i'=0,\cdots,H_l-1 \quad t=0,\cdots,T-1\\ j'=0,\cdots,W_l-1 \quad k'=0,\cdots,C_l-1}}$$

输出数据块为网络 r 结点的激活值数据块（activation tensor）：

$$X_{\times H_r,\times W_r}^{(r),(\times T),(\times C_r)}=\left\{x_{i,j}^{(r),(t),(k)}\right\}\Big|_{\substack{i=0,\cdots,H_r-1 \quad t=0,\cdots,T-1\\ j=0,\cdots,W_r-1 \quad k=0,\cdots,C_r-1}}$$

通常输入输出数据块的宽高均为 1，$W_l=H_l=W_r=H_r=1$，相当于特征图 feature map 的尺寸缩为 1×1，这是 LeNet5 中 C5 层的观点，而通道数仍为 C_l 和 C_r，则输入输出数据块简化为：

$$X_{\times 1,\times 1}^{(l),(\times T),(\times C_l)}=\left\{x_{1,1}^{(l),(t),(k')}\right\}\Big|_{\substack{t=0,\cdots,T-1\\ k'=0,\cdots,C_l-1}}=\left[X_{\times 1,\times 1}^{(l),(0),(\times C_l)} \quad \cdots \quad X_{\times 1,\times 1}^{(l),(T-1),(\times C_l)}\right]_{1\times T}$$

$$X_{\times 1,\times 1}^{(r),(\times T),(\times C_r)}=\left\{x_{1,1}^{(r),(t),(k)}\right\}\Big|_{\substack{t=0,\cdots,T-1\\ k=0,\cdots,C_r-1}}=\left[X_{\times 1,\times 1}^{(r),(0),(\times C_r)} \quad \cdots \quad X_{\times 1,\times 1}^{(r),(T-1),(\times C_r)}\right]_{1\times T}$$

这两个数据块形式上为二维阵（2d-tensor），实际前向计算时对 t $(t=0,\cdots,T-1)$ 维度的每个 t 是逐个分开计算的，如下图所示，

$$X_{\times 1,\times 1}^{(l),(t),(\times C_l)} \xrightarrow{full\ connected} X_{\times 1,\times 1}^{(r),(t),(\times C_r)} \qquad t=0,\cdots,T-1$$

其中的 $X_{\times 1,\times 1}^{(l),(t),(\times C_l)}$ 和 $X_{\times 1,\times 1}^{(r),(t),(\times C_r)}$ 均为向量（1d-tensor），

$$X_{\times 1,\times 1}^{(l),(t),(\times C_l)}=\begin{bmatrix}x_{1,1}^{(l),(t),(0)}\\ \vdots\\ x_{1,1}^{(l),(t),(C_l-1)}\end{bmatrix}_{C_l\times 1} \qquad X_{\times 1,\times 1}^{(r),(t),(\times C_r)}=\begin{bmatrix}x_{1,1}^{(r),(t),(0)}\\ \vdots\\ x_{1,1}^{(r),(t),(C_r-1)}\end{bmatrix}_{C_r\times 1} \qquad t=0,\cdots,T-1$$

参数为全连接前向权值数据块（2d-weight tensor）：

$$W_{\times 1,\times 1}^{(r),(\times C_r),(\times C_l)}=\left\{w_{1,1}^{(r),(k),(k')}\right\}\Big|_{\substack{k=0,\cdots,C_r-1\\ k'=0,\cdots,C_l-1}}=\begin{bmatrix}w_{1,1}^{(r),(0),(0)} & \cdots & w_{1,1}^{(r),(0),(C_l-1)}\\ \vdots & \ddots & \vdots\\ w_{1,1}^{(r),(C_r-1),(0)} & \cdots & w_{1,1}^{(r),(C_r-1),(C_l-1)}\end{bmatrix}_{C_r\times C_l}$$

对比 $M \times N$ 小窗口卷积的前向卷积权值数据块（4d-weight tensor）：

$$W_{\times M,\times N}^{(r),(\times C_r),(\times C_l)} = \left\{W_{\times M,\times N}^{(r),(k),(k')}\right\}_{\substack{k=0,\cdots,C_r-1 \\ k'=0,\cdots,C_l-1}} = \begin{bmatrix} W_{\times M,\times N}^{(r),(0),(0)} & \cdots & W_{\times M,\times N}^{(r),(0),(C_l-1)} \\ \vdots & \ddots & \vdots \\ W_{\times M,\times N}^{(r),(C_r-1),(0)} & \cdots & W_{\times M,\times N}^{(r),(C_r-1),(C_l-1)} \end{bmatrix}_{C_r \times C_l}$$

如前所述，输入输出数据块的 feature map 的尺寸缩为 1×1，则全连接前向权值数据块相当于将上式中的每个 $M \times N$ 卷积小窗口缩为一个点，

$$\left\{W_{\times M,\times N}^{(r),(k),(k')}\right\}_{\substack{k=0,\cdots,C_r-1 \\ k'=0,\cdots,C_l-1}} \xrightarrow{window\ to\ dot} \left\{w_{1,1}^{(r),(k),(k')}\right\}\Big|_{\substack{k=0,\cdots,C_r-1 \\ k'=0,\cdots,C_l-1}}$$

全连接前向计算，可以看作是卷积计算的特殊形式。矩阵形式的全连接前向计算公式为，

$$X_{\times 1,\times 1}^{(r),(t),(\times C_r)} = W_{\times 1,\times 1}^{(r),(\times C_r),(\times C_l)} \times X_{\times 1,\times 1}^{(l),(t),(\times C_l)} \qquad t = 0,\cdots,T-1$$

$$\begin{bmatrix} x_{1,1}^{(r),(t),(0)} \\ \vdots \\ x_{1,1}^{(r),(t),(C_r-1)} \end{bmatrix}_{C_r \times 1} = \begin{bmatrix} w_{1,1}^{(r),(0),(0)} & \cdots & w_{1,1}^{(r),(0),(C_l-1)} \\ \vdots & \ddots & \vdots \\ w_{1,1}^{(r),(C_r-1),(0)} & \cdots & w_{1,1}^{(r),(C_r-1),(C_l-1)} \end{bmatrix}_{C_r \times C_l} \times \begin{bmatrix} x_{1,1}^{(l),(t),(0)} \\ \vdots \\ x_{1,1}^{(l),(t),(C_l-1)} \end{bmatrix}_{C_l \times 1}$$

$$t = 0,\cdots,T-1$$

进一步整合为，

$$X_{\times 1,\times 1}^{(r),(\times T),(\times C_r)}\Big|_{C_r \times T} = W_{\times 1,\times 1}^{(r),(\times C_r),(\times C_l)}\Big|_{C_r \times C_l} \times X_{\times 1,\times 1}^{(l),(\times T),(\times C_l)}\Big|_{C_l \times T}$$

前向 Full 函数表达为 $Full_F$：

$$X_{\times 1,\times 1}^{(r),(\times T),(\times C_r)} = Full_F\left(X_{\times 1,\times 1}^{(l),(\times T),(\times C_l)}, W_{\times 1,\times 1}^{(r),(\times C_r),(\times C_l)}\right)$$

其中每个分量值的计算公式为：

$$x_{1,1}^{(r),(t),(k)} = \sum_{k'=0}^{C_l-1} w_{1,1}^{(r),(k),(k')} \cdot x_{1,1}^{(l),(t),(k')} \qquad \begin{cases} t = 0,\cdots,T-1 \\ k = 0,\cdots,C_r-1 \end{cases}$$

Full 权值梯度计算

代价函数对全连接前向权值数据块（2d-weight tensor）

$$W_{\times 1,\times 1}^{(r),(\times C_r),(\times C_l)} = \left\{ w_{1,1}^{(r),(k),(k')} \right\}\Big|_{\substack{k=0,\cdots,C_r-1 \\ k'=0,\cdots,C_l-1}}$$

的每个分量求导，得到权值梯度数据块（weight gradient tensor）

$$\Delta W_{\times 1,\times 1}^{(r),(\times C_r),(\times C_l)} = \left\{ \Delta w_{1,1}^{(r),(k),(k')} = \frac{\partial J}{\partial \left(w_{1,1}^{(r),(k),(k')} \right)} \right\}\Bigg|_{\substack{k=0,\cdots,C_r-1 \\ k'=0,\cdots,C_l-1}}$$

由前向 Full 公式

$$x_{1,1}^{(r),(t),(k)} = \sum_{k'=0}^{C_l-1} w_{1,1}^{(r),(k),(k')} \cdot x_{1,1}^{(l),(t),(k')} \qquad \begin{cases} t = 0,\cdots,T-1 \\ k = 0,\cdots,C_r-1 \end{cases}$$

可知，如果固定 (t)，则权值梯度由单一式子确定，

$$\Delta w_{1,1}^{(r),(k),(k')} = \frac{\partial J}{\partial \left(w_{1,1}^{(r),(k),(k')} \right)} = \frac{\partial J}{\partial \left(x_{1,1}^{(r),(t),(k)} \right)} \cdot \frac{\partial \left(x_{1,1}^{(r),(t),(k)} \right)}{\partial \left(w_{1,1}^{(r),(k),(k')} \right)} = \delta_{1,1}^{(r),(t),(k)} \cdot x_{1,1}^{(l),(t),(k')}$$

$$\begin{cases} k = 0,\cdots,C_r-1 \\ k' = 0,\cdots,C_l-1 \end{cases}$$

再考虑到 mini-batch 批量 T，权值梯度合成如下，

$$\Delta w_{1,1}^{(r),(k),(k')} = \sum_{t=0,\cdots,T-1} \delta_{1,1}^{(r),(t),(k)} \cdot x_{1,1}^{(l),(t),(k')} \qquad \begin{cases} k = 0,\cdots,C_r-1 \\ k' = 0,\cdots,C_l-1 \end{cases}$$

Full 权值梯度计算函数表达为 $Full_{grad}$：

$$\Delta W_{\times 1,\times 1}^{(r),(\times C_r),(\times C_l)} = Full_{grad}\left(\Delta_{\times 1,\times 1}^{(r),(\times T),(\times C_r)}, X_{\times 1,\times 1}^{(l),(\times T),(\times C_l)} \right)$$

其中，一部分输入数据块为网络 r 结点的敏感值数据块（sensitivity tensor）：

$$\Delta_{\times 1,\times 1}^{(r),(\times T),(\times C_r)} = \left\{ \delta_{1,1}^{(r),(t),(k)} \right\}\Big|_{\substack{t=0,\cdots,T-1 \\ k=0,\cdots,C_r-1}}$$

另一部分输入数据块为网络 l 结点的激活值数据块（activation tensor）：

$$X_{\times 1,\times 1}^{(l),(\times T),(\times C_l)} = \left\{ x_{1,1}^{(l),(t),(k')} \right\}\Big|_{\substack{t=0,\cdots,T-1 \\ k'=0,\cdots,C_l-1}}$$

输出为权值梯度数据块（weight gradient tensor）

$$\Delta W_{\times 1,\times 1}^{(r),(\times C_r),(\times C_l)} = \left\{ \Delta w_{1,1}^{(r),(k),(k')} \right\}\Big|_{\substack{k=0,\cdots,C_r-1 \\ k'=0,\cdots,C_l-1}}$$

Full 反向计算

Full 反向计算也为全连接（full connected layer）网络计算，其输入数据块为网络 r 结点的敏感值数据块（sensitivity tensor）：

$$\Delta_{\times H_r,\times W_r}^{(r),(\times T),(\times C_r)} = \left\{ \delta_{i,j}^{(r),(t),(k)} \right\}\Big|_{\substack{i=0,\cdots,H_r-1 \quad t=0,\cdots,T-1 \\ j=0,\cdots,W_r-1 \quad k=0,\cdots,C_r-1}}$$

输出数据块为网络 l 结点的敏感值数据块（sensitivity tensor）：

$$\Delta_{\times H_l,\times W_l}^{(l),(\times T),(\times C_l)} = \left\{ \delta_{i',j'}^{(l),(t),(k')} \right\}\Big|_{\substack{i'=0,\cdots,H_l-1 \quad t=0,\cdots,T-1 \\ j'=0,\cdots,W_l-1 \quad k'=0,\cdots,C_l-1}}$$

与前向一样，通常输入输出数据块的宽高均为 1，$W_l = H_l = W_r = H_r = 1$，相当于特征图 feature map 的尺寸缩为 1×1，而通道数仍为 C_l 和 C_r，则输入输出数据块简化为：

$$\Delta_{\times H_r,\times W_r}^{(r),(\times T),(\times C_r)} = \left\{ \delta_{1,1}^{(r),(t),(k)} \right\}\Big|_{\substack{t=0,\cdots,T-1 \\ k=0,\cdots,C_r-1}} = \left[\Delta_{\times 1,\times 1}^{(r),(0),(\times C_r)} \quad \cdots \quad \Delta_{\times 1,\times 1}^{(r),(T-1),(\times C_r)} \right]_{1\times T}$$

$$\Delta_{\times H_l,\times W_l}^{(l),(\times T),(\times C_l)} = \left\{ \delta_{1,1}^{(l),(t),(k')} \right\}\Big|_{\substack{t=0,\cdots,T-1 \\ k'=0,\cdots,C_l-1}} = \left[\Delta_{\times 1,\times 1}^{(l),(0),(\times C_l)} \quad \cdots \quad \Delta_{\times 1,\times 1}^{(l),(T-1),(\times C_l)} \right]_{1\times T}$$

这两个数据块形式上为二维阵（2d-tensor），实际反向计算时对 t $(t = 0,\cdots,T-1)$ 维度的每个 t 是逐个分开计算的，如下图所示，

$$\Delta_{\times 1,\times 1}^{(r),(t),(\times C_r)} \xrightarrow{full\ connected} \Delta_{\times 1,\times 1}^{(l),(t),(\times C_l)} \qquad t = 0,\cdots,T-1$$

其中的 $\Delta_{\times 1,\times 1}^{(r),(t),(\times C_r)}$ 和 $\Delta_{\times 1,\times 1}^{(l),(t),(\times C_l)}$ 均为向量（1d-tensor），

$$\Delta_{\times 1,\times 1}^{(r),(t),(\times C_r)} = \begin{bmatrix} \delta_{1,1}^{(r),(t),(0)} \\ \vdots \\ \delta_{1,1}^{(r),(t),(C_r-1)} \end{bmatrix}_{C_r\times 1} \quad \Delta_{\times 1,\times 1}^{(l),(t),(\times C_l)} = \begin{bmatrix} \delta_{1,1}^{(l),(t),(0)} \\ \vdots \\ \delta_{1,1}^{(l),(t),(C_l-1)} \end{bmatrix}_{C_l\times 1} \qquad t = 0,\cdots,T-1$$

考虑到全连接的前向计算公式为，

$$x_{1,1}^{(r),(t),(k)}=\sum_{k'=0}^{C_l-1} w_{1,1}^{(r),(k),(k')}\cdot x_{1,1}^{(l),(t),(k')} \qquad \begin{cases} t=0,\cdots,T-1 \\ k=0,\cdots,C_r-1 \end{cases}$$

则前向传播关系图为，此图为从 l 结点到 r 结点的多对一 $(N_f\ to\ 1)$ 关系图，

$$\left\{x_{1,1}^{(l),(t),(k')}\right\}_{k'=0,\cdots,C_l-1} \xrightarrow{N_f\ to\ N_f} \left\{w_{1,1}^{(r),(k),(k')}\right\}_{k'=0,\cdots,C_l-1} \xrightarrow{N_f\ to\ 1} x_{1,1}^{(r),(t),(k)}$$

$$\begin{cases} t=0,\cdots,T-1 \\ k=0,\cdots,C_r-1 \end{cases} \qquad N_f=C_l$$

在上图中，左端上标 (k') 为游标，现将其固定，使右端上标 (k) 变为游标，则上图变为，

$$x_{1,1}^{(l),(t),(k')} \xrightarrow{1\ to\ N_b} \left\{w_{1,1}^{(r),(k),(k')}\right\}_{k=0,\cdots,C_r-1} \xrightarrow{N_b\ to\ N_b} \left\{x_{1,1}^{(r),(t),(k)}\right\}_{k=0,\cdots,C_r-1}$$

$$\begin{cases} t=0,\cdots,T-1 \\ k'=0,\cdots,C_l-1 \end{cases} \qquad N_b=C_r$$

将上图左右翻转，改为敏感值的对应关系，并且将权值改名，

$$\left\{\delta_{1,1}^{(r),(t),(k)}\right\}_{k=0,\cdots,C_r-1} \xrightarrow{N_b\ to\ N_b} \left\{v_{1,1}^{(r),(k'),(k)}\right\}_{k=0,\cdots,C_r-1} \xrightarrow{N_b\ to\ 1} \delta_{1,1}^{(l),(t),(k')}$$

$$\begin{cases} t=0,\cdots,T-1 \\ k'=0,\cdots,C_l-1 \end{cases}$$

$$v_{1,1}^{(r),(k'),(k)}=w_{1,1}^{(r),(k),(k')} \qquad \begin{cases} k'=0,\cdots,C_l-1 \\ k=0,\cdots,C_r-1 \end{cases}$$

全连接反向权值数据块（2d-weight tensor）：

$$V_{\times 1,\times 1}^{(r),(\times C_l),(\times C_r)}=\left\{v_{1,1}^{(r),(k'),(k)}\right\}\Big|_{\substack{k'=0,\cdots,C_l-1 \\ k=0,\cdots,C_r-1}}=\begin{bmatrix} v_{1,1}^{(r),(0),(0)} & \cdots & v_{1,1}^{(r),(0),(C_r-1)} \\ \vdots & \ddots & \vdots \\ v_{1,1}^{(r),(C_l-1),(0)} & \cdots & v_{1,1}^{(r),(C_l-1),(C_r-1)} \end{bmatrix}_{C_l\times C_r}$$

对比全连接前向权值数据块（2d-weight tensor），考虑到 $v_{1,1}^{(r),(k'),(k)}=w_{1,1}^{(r),(k),(k')}$，

$$W_{\times 1,\times 1}^{(r),(\times C_r),(\times C_l)}=\left\{w_{1,1}^{(r),(k),(k')}\right\}\Big|_{\substack{k=0,\cdots,C_r-1 \\ k'=0,\cdots,C_l-1}}=\begin{bmatrix} w_{1,1}^{(r),(0),(0)} & \cdots & w_{1,1}^{(r),(0),(C_l-1)} \\ \vdots & \ddots & \vdots \\ w_{1,1}^{(r),(C_r-1),(0)} & \cdots & w_{1,1}^{(r),(C_r-1),(C_l-1)} \end{bmatrix}_{C_r\times C_l}$$

可知，$V_{\times 1,\times 1}^{(r),(\times C_l),(\times C_r)} = \left[W_{\times 1,\times 1}^{(r),(\times C_r),(\times C_l)}\right]'$，即反向权值阵与前向权值阵互为转置关系，再对比 $M \times N$ 小窗口卷积的反向敏感值卷积权值数据块（4d-weight tensor）：

$$V_{\times M,\times N}^{(r),(\times C_l),(\times C_r)} = \left\{V_{\times M,\times N}^{(r),(k'),(k)}\right\}_{\substack{k'=0,\cdots,C_l-1\\ k=0,\cdots,C_r-1}} = \begin{bmatrix} V_{\times M,\times N}^{(r),(0),(0)} & \cdots & V_{\times M,\times N}^{(r),(0),(C_r-1)} \\ \vdots & \ddots & \vdots \\ V_{\times M,\times N}^{(r),(C_l-1),(0)} & \cdots & V_{\times M,\times N}^{(r),(C_l-1)(C_r-1)} \end{bmatrix}_{C_l \times C_r}$$

如前所述，输入输出数据块的 feature map 的尺寸缩为 1×1，则全连接反向权值数据块相当于将上式中的每个 $M \times N$ 卷积小窗口缩为一个点，

$$\left\{V_{\times M,\times N}^{(r),(k'),(k)}\right\}_{\substack{k'=0,\cdots,C_l-1\\ k=0,\cdots,C_r-1}} \xrightarrow{window\ to\ dot} \left\{v_{1,1}^{(r),(k'),(k)}\right\}\Big|_{\substack{k'=0,\cdots,C_l-1\\ k=0,\cdots,C_r-1}}$$

全连接反向敏感值计算，也可以看作是卷积计算的特殊形式。矩阵形式的全连接反向计算公式为，

$$\Delta_{\times 1,\times 1}^{(l),(t),(\times C_l)} = V_{\times 1,\times 1}^{(r),(\times C_l),(\times C_r)} \times \Delta_{\times 1,\times 1}^{(r),(t),(\times C_r)} \qquad t = 0,\cdots,T-1$$

$$\begin{bmatrix} \delta_{1,1}^{(l),(t),(0)} \\ \vdots \\ \delta_{1,1}^{(l),(t),(C_l-1)} \end{bmatrix}_{C_l \times 1} = \begin{bmatrix} v_{1,1}^{(r),(0),(0)} & \cdots & v_{1,1}^{(r),(0),(C_r-1)} \\ \vdots & \ddots & \vdots \\ v_{1,1}^{(r),(C_l-1),(0)} & \cdots & v_{1,1}^{(r),(C_l-1),(C_r-1)} \end{bmatrix}_{C_l \times C_r} \times \begin{bmatrix} \delta_{1,1}^{(r),(t),(0)} \\ \vdots \\ \delta_{1,1}^{(r),(t),(C_r-1)} \end{bmatrix}_{C_r \times 1}$$

$$t = 0,\cdots,T-1$$

进一步整合为，

$$\Delta_{\times 1,\times 1}^{(l),(\times T),(\times C_l)}\Big|_{C_l \times T} = V_{\times 1,\times 1}^{(r),(\times C_l),(\times C_r)}\Big|_{C_l \times C_r} \times \Delta_{\times 1,\times 1}^{(r),(\times T),(\times C_r)}\Big|_{C_r \times T}$$

反向 Full 函数表达为 $Full_{BF}$：

$$\Delta_{\times 1,\times 1}^{(l),(\times T),(\times C_l)} = Full_{BF}\left(\Delta_{\times 1,\times 1}^{(r),(\times T),(\times C_r)}, V_{\times 1,\times 1}^{(r),(\times C_l),(\times C_r)}\right)$$

其中每个分量值的计算公式为：

$$\delta_{1,1}^{(l),(t),(k')} = \sum_{k=0}^{C_r-1} v_{1,1}^{(r),(k'),(k)} \cdot \delta_{1,1}^{(r),(t),(k)} \qquad \begin{cases} t = 0,\cdots,T-1 \\ k' = 0,\cdots,C_l-1 \end{cases}$$

Softmax Net

Softmax function

定义 softmax 函数

$$x=\begin{bmatrix}x_1\\ \vdots\\ x_n\end{bmatrix}\qquad y=\text{softmax}(x)=\begin{bmatrix}y_1\\ \vdots\\ y_n\end{bmatrix}=\frac{1}{\sum_{j=1}^{n}e^{x_j}}\cdot\begin{bmatrix}e^{x_1}\\ \vdots\\ e^{x_n}\end{bmatrix}$$

令 $\sigma(x_1,\cdots,x_n)=\frac{1}{\sum_{j=1}^{n}e^{x_j}}$，则有

$$y_i=\sigma(x_1,\cdots,x_n)\cdot e^{x_i}=\frac{e^{x_i}}{\sum_{j=1}^{n}e^{x_j}}\qquad i=1,\cdots,n$$

并且 y_i 满足归一化

$$\sum_{i=1}^{n}y_i=1$$

softmax 函数是一个自变量和函数值均为向量的函数，其函数值向量的每个分量均为正值，且满足归一化条件。

从函数变换的过程上看，softmax 函数先将一组输入数据做指数变换，都映射到正值，但不改变各数值的排列次序，然后再做归一化，输出值都在(0,1) 区间，并且其和为 1。

将指数函数 e^x 与线性恒等函数 x 的变化率进行比较，

$$\begin{cases}(e^x)'>(x)' & \Rightarrow & e^x>1 & \Rightarrow & x>0\\ (e^x)'<(x)' & \Rightarrow & e^x<1 & \Rightarrow & x<0\end{cases}$$

如果输入数据是某种多指标评判结果的话，可以看出，取指数操作将较大的数（$x>0$）进一步增大，较小的数（$x<0$）进一步压缩。

由指数函数特性，输入的正值越大，输出增大效果越明显，而负值都压缩进 (0,1) 区间，负值离 0 点越远，输出值越钝化。也就是说，softmax 有优势放大、劣势压缩的效果，并且对正值的敏感性远大于负值。

softmax 函数具有超正方体顶点对平移不变性，将自变量向量的每个分量平移相同的值，其 softmax 函数值向量保持不变。

$$y = \begin{bmatrix} y_1 \\ \vdots \\ y_n \end{bmatrix} = x - \begin{bmatrix} \min(x_i) \\ \vdots \\ \min(x_i) \end{bmatrix} = \begin{bmatrix} x_1 - \min(x_i) \\ \vdots \\ x_n - \min(x_i) \end{bmatrix}$$

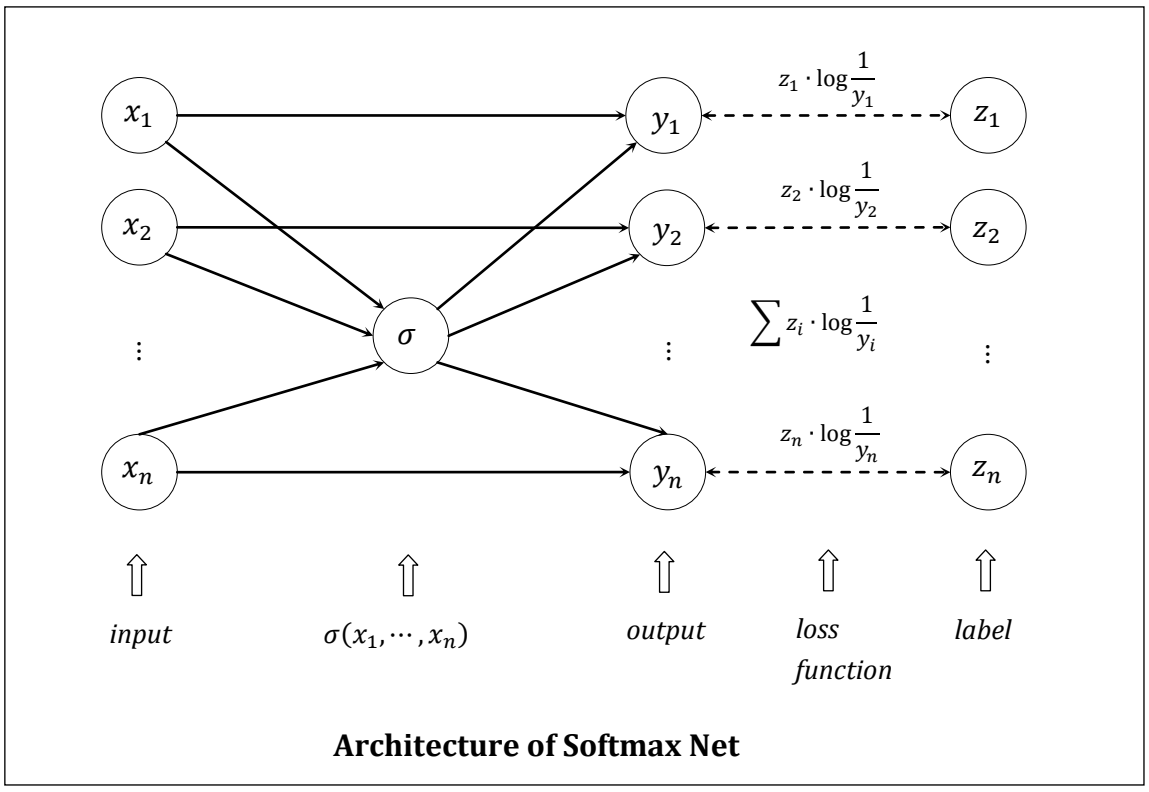
则有

$$\text{softmax}(y) = \text{softmax}(x)$$

Softmax Net

将 softmax 函数看做是一个子网络（Softmax Net），输入为 $\{x_1, \cdots, x_n\}$，输出为 $\{y_1, \cdots, y_n\}$，附加监督标签为 $\{z_1, \cdots, z_n\}$，一般为类似单位向量的 one-hot vector，也可以泛化为满足归一化条件，$\sum_{i=1}^{n} z_i = 1$。

Softmax Net 的运算关系如图所示。



Architecture of Softmax Net

Softmax 网络并不含有可学习权值，Softmax 算法本质为一个**归一化多指标输出与标签比对模型**，其输出为一组归一化的数据（向量），这组数据可以用来与既有标签（同维度向量）进行比对（有监督）。

Likelihood function

将归一化的网络输出（softmax 输出 $\{y_1, \cdots, y_n\}$ 的各个分量）看做是概率值，并将各个输出分量以连乘的形式表达，得到似然函数。

对于有监督网络，监督标签 $\{z_1, \cdots, z_n\}$ 为 one-hot vector 形式。对全网络而言，如果某个训练样本 s，其最终输出为 $\{y_1, \cdots, y_n\}$，表示网络经过计算认为，样本 s 的分类是第 i 类的概率为 y_i；又有，

其监督标签为 $\{z_1,\cdots,z_n\}$，其中只有一个 $z_{i_0}=1$，其它都为 0，则表示样本 s 的真实分类（ground truth）为第 i_0 类，记样本 s 的真实分类值函数为 $d(s)$，比如 $d(s)=i_0$。

在以上设定下，一个样本 s 对应的似然函数 $L(s;\theta)$ 表示为

$$L(s;\theta)=\prod_{i=1}^{n}(y_i)^{z_i}=\prod_{i=1}^{n}(P\{d(s)=i|s;\theta\})^{z_i}$$

形式上，似然函数 $L(s;\theta)$ 是一个概率值连乘函数。而事实上，对每个特定样本 s，连乘分项虽然为 n 式相乘式（下式），但其有效概率值只会取其中之一，其它项都取 1，因为样本 s 的真实分类值 $d(s)$ 只能取 $1,\cdots,n$ 中的某一个值，比如 $d(s)=i_0$，而该值早已经由监督标签 $\{z_1,\cdots,z_n\}$ 给出，其中只有一个 $z_{i_0}=1$，其它都为 0，

$$\prod_{i=1}^{n}(y_i)^{z_i}=y_{i_0}$$

$$\prod_{i=1}^{n}(P\{d(s)=i|s;\theta\})^{z_i}=P\{d(s)=i_0|s;\theta\}$$

指数部分的监督标签分量值 z_i，实际上起到一个挑选的作用，用有监督标签对所有的输出分量进行挑选，挑出与标签相对应的输出分量，通过反向传播学习算法，反复修正网络参数 θ，使这个输出分量最大化，这就是**最大似然原理**（principal of maximum likelihood）的学习算法。

Negative log likelihood function

以上似然函数的含义比较直观，但连乘式不好操作，主要是不方便求导。对似然函数进行负对数复合，得到常用的代价函数，就是**交叉熵代价函数**（cross entropy loss function）

$$J(s;\theta)=-logL(s;\theta)=-\log\left(\prod_{i=1}^{n}(y_i)^{z_i}\right)$$

$$J(s;\theta)=-\sum_{i=1}^{n}z_i\cdot\log y_i=\sum_{i=1}^{n}z_i\cdot\log\frac{1}{y_i}$$

同样的道理，代价函数 $J(\theta)$ 形式上是一个连加函数。而事实上，对每个特定样本 s，求和式中只会有一项取非 0 值，其它项都取 0，因为样本 s 的真实分类值 $d(s)$ 只能取 $1,\cdots,n$ 中的某一个值，比如 $d(s)=i_0$，而该值由标签 $\{z_1,\cdots,z_n\}$ 给出，其中只有一个 $z_{i_0}=1$，其它都为 0，

$$J(s;\theta)=-\log\left(\prod_{i=1}^{n}(y_i)^{z_i}\right)=-\sum_{i=1}^{n}z_i\cdot\log y_i=-\log y_{i_0}\qquad d(s)=i_0$$

基于信息熵的解释

将网络输出记为随机变量 Y 的分布 $\{y_1,\cdots,y_n\}$，将监督标签记为随机变量 Z 的分布 $\{z_1,\cdots,z_n\}$。根据交叉熵的定义和性质，代价函数 $J(s;\theta)$ 相当于两项之和，即网络输出 Y 与监督标签 Z 的差异度信息 $H_{KL}(Z||Y)$（KL 散度）加上监督标签 Z 的信息 $H(Z)$。

$$J(s;\theta)=\sum_{i=1}^{n}z_i\cdot\log\frac{1}{y_i}=H_{cross}(Z||Y)=H_{KL}(Z||Y)+H(Z)$$

网络的学习过程就是不断地减小代价函数 $J(s;\theta)$ 即减小差异度信息 $H_{KL}(Z||Y)$ 的过程。

对于 one-hot 型监督标签，其信息 $H(Z)=0$，表示确定性的信息，则网络的学习过程就是不断地减小网络输出 Y 的信息量 $H(Y)$，并将其分布 $\{y_1,\cdots,y_n\}$ 逐步收敛到 one-hot 型监督标签 Z 的分布 $\{z_1,\cdots,z_n\}$ 的过程。

Generalized Likelihood function

将标签向量推广为更一般的形式，如下，

$$z=\begin{bmatrix}z_1\\ \vdots\\ z_n\end{bmatrix}\qquad\begin{cases}i=1,\cdots,n\\ z_i\in[0,1]\end{cases}\qquad\sum_{i=1}^{n}z_i=1$$

其中的分量 z_i 可以取 0~1 之间的实数值，并且满足归一化条件 $\sum_{i=1}^{n}z_i=1$。

将似然函数进行推广，形式不变，但内涵已变，成为实实在在的连乘函数，

$$L(s;\theta)=\prod_{i=1}^{n}(y_i)^{z_i}$$

Generalized Negative log likelihood function

对推广的似然函数进行负对数复合，得到推广的代价函数，

$$J(s;\theta)=-logL(s;\theta)=-\log\left(\prod_{i=1}^{n}(y_i)^{z_i}\right)$$

$$J(s;\theta)=-\sum_{i=1}^{n}z_i\cdot\log y_i$$

同样，形式不变，内涵已变。

Sensitivity $\boldsymbol{\delta_{x_i}}$

求输入值 x_i 对应的**敏感值** δ_{x_i}。该值实为代价函数 $J(\theta)$ 相对于输入值 x_i 的导数。

$$\delta_{x_i} = \frac{\partial J(\theta)}{\partial (x_i)} = -(z_i - y_i) \qquad i = 1, \cdots, n$$

以下求导用推广的代价函数。

$$J(\theta) = -\sum_{j=1}^{n} z_j \cdot \log y_j$$

根据 Softmax Net 的网络图结构，应用反向 chain rules，依次求出梯度值 $\frac{\partial J(\theta)}{\partial (y_i)}$, $\frac{\partial J(\theta)}{\partial \sigma}$, $\frac{\partial J(\theta)}{\partial (x_i)}$,

$$\frac{\partial J(\theta)}{\partial (y_i)} = -\frac{z_i}{y_i} \qquad i = 1, \cdots, n$$

$$\frac{\partial J(\theta)}{\partial \sigma} = \sum_{j=1}^{n} \frac{\partial J(\theta)}{\partial (y_j)} \cdot \frac{\partial (y_j)}{\partial \sigma} = \sum_{j=1}^{n} \left(-\frac{z_j}{y_j}\right) \cdot e^{x_j} = \frac{-1}{\sigma} \sum_{j=1}^{n} z_j$$

$$note: \qquad y_i = \sigma(x_1, \cdots, x_n) \cdot e^{x_i} = \frac{e^{x_i}}{\sum_{j=1}^{n} e^{x_j}} \qquad i = 1, \cdots, n$$

$$\frac{\partial J(\theta)}{\partial (x_i)} = \frac{\partial J(\theta)}{\partial (y_i)} \cdot \frac{\partial (y_i)}{\partial (x_i)} + \frac{\partial J(\theta)}{\partial \sigma} \cdot \frac{\partial \sigma}{\partial (x_i)} \qquad i = 1, \cdots, n$$

$$\frac{\partial J(\theta)}{\partial (x_i)} = \left(-\frac{z_i}{y_i}\right) \cdot (\sigma \cdot e^{x_i}) + \left(\frac{-1}{\sigma} \sum_{j=1}^{n} z_j\right) \cdot \left(\frac{-e^{x_i}}{\left(\sum_{j=1}^{n} e^{x_j}\right)^2}\right) \qquad i = 1, \cdots, n$$

$$note: \qquad \sigma = \frac{1}{\sum_{j=1}^{n} e^{x_j}} \quad \Rightarrow \quad \frac{\partial \sigma}{\partial (x_i)} = \frac{-e^{x_i}}{\left(\sum_{j=1}^{n} e^{x_j}\right)^2} \qquad i = 1, \cdots, n$$

$$\frac{\partial J(\theta)}{\partial (x_i)} = (-z_i) + \left(\frac{1}{\sigma} \sum_{j=1}^{n} z_j\right) \cdot (\sigma^2 \cdot e^{x_i}) \qquad i = 1, \cdots, n$$

$$note: \qquad y_i = \sigma \cdot e^{x_i} \qquad i = 1, \cdots, n$$

$$\frac{\partial J(\theta)}{\partial(x_i)} = (-z_i) + \left(\sum_{j=1}^{n} z_j\right) \cdot (y_i) \qquad i = 1, \cdots, n$$

$$note: \qquad \sum_{j=1}^{n} z_j = 1$$

$$\frac{\partial J(\theta)}{\partial(x_i)} = -(z_i - y_i) \qquad i = 1, \cdots, n$$

Softmax 前向计算

函数原型：softmax 函数将 n 维向量 x 进行压缩和归一化操作，输出 n 维向量 y，

$$x = \begin{bmatrix} x_1 \\ \vdots \\ x_n \end{bmatrix} \qquad y = \text{softmax}(x) = \begin{bmatrix} y_1 \\ \vdots \\ y_n \end{bmatrix} = \frac{1}{\sum_{j=1}^{n} e^{x_j}} \cdot \begin{bmatrix} e^{x_1} \\ \vdots \\ e^{x_n} \end{bmatrix}$$

注释：按照此公式，每个分量不能取到 0 或 1 值，所以有 $y_i \in (0,1)$。并且向量 y 满足归一化条件，

$$\sum_{i=1}^{n} y_i = 1$$

Softmax 函数的输入输出均为向量，与 Full 类似，Softmax 前向计算的输入为网络 l 结点的激活值数据块（activation tensor）：

$$X_{\times 1,\times 1}^{(l),(\times T),(\times C_l)} = \left\{ x_{1,1}^{(l),(t),(k')} \right\}\Big|_{\substack{t=0,\cdots,T-1 \\ k'=0,\cdots,C_l-1}} = \left[X_{\times 1,\times 1}^{(l),(0),(\times C_l)} \quad \cdots \quad X_{\times 1,\times 1}^{(l),(T-1),(\times C_l)} \right]_{1\times T}$$

注意到 T 表达的是 mini-batch 输入样本的个数，该数据块实际为 T 个 C_l 维向量，其中第 t 个 C_l 维向量已经是对 Input 层相应第 t 个输入样本的最后的综合特征的向量表达了，只是这个向量 $X_{\times 1,\times 1}^{(l),(t),(\times C_l)}$ 还不满足归一化条件，不方便与最终的监督数据（one-hot vector）相比较，需要将其经过 Softmax 函数处理后，得到归一化的对应向量 $X_{\times 1,\times 1}^{(r),(t),(\times C_r)}$。

输出为网络 r 结点的激活值数据块（activation tensor）：

$$X_{\times 1,\times 1}^{(r),(\times T),(\times C_r)} = \left\{ x_{1,1}^{(r),(t),(k)} \right\}\Big|_{\substack{t=0,\cdots,T-1 \\ k=0,\cdots,C_r-1}} = \left[X_{\times 1,\times 1}^{(r),(0),(\times C_r)} \quad \cdots \quad X_{\times 1,\times 1}^{(r),(T-1),(\times C_r)} \right]_{1\times T}$$

输出数据块同样为 T 个 C_r 维向量，每个向量都已经满足归一化条件了。注意，此时的通道数与输入数据块的是一样的，为 $C_r = C_l$，结点标记为 l 结点和 r 结点。

前向 Softmax 函数表达为 $Softmax_F$：

$$X_{\times 1,\times 1}^{(r),(\times T),(\times C_r)} = Softmax_F\left(X_{\times 1,\times 1}^{(l),(\times T),(\times C_l)}\right)$$

其中每个分量值的计算公式为：

$$x_{1,1}^{(r),(t),(k)} = \frac{exp\left(x_{1,1}^{(l),(t),(k')}\right)}{\sum_{k'=0}^{C_l-1} exp\left(x_{1,1}^{(l),(t),(k')}\right)} \qquad \begin{cases} t = 0,\cdots,T-1 \\ k = 0,\cdots,C_r-1 \end{cases}$$

或者写为向量形式，

$$\begin{bmatrix} x_{1,1}^{(r),(t),(0)} \\ x_{1,1}^{(r),(t),(1)} \\ \vdots \\ x_{1,1}^{(r),(t),(C_r-1)} \end{bmatrix}_{C_r\times 1} = \frac{1}{\sum_{k'=0}^{C_l-1} exp\left(x_{1,1}^{(l),(t),(k')}\right)} \cdot \begin{bmatrix} exp\left(x_{1,1}^{(l),(t),(0)}\right) \\ exp\left(x_{1,1}^{(l),(t),(1)}\right) \\ \vdots \\ exp\left(x_{1,1}^{(l),(t),(C_l-1)}\right) \end{bmatrix}_{C_l\times 1} \quad t = 0,\cdots,T-1$$

Softmax 反向计算

softmax 反向计算的目的，是得到与 Softmax 前向计算的输入，即网络 l 结点的激活值数据块相对应的敏感值数据块（sensitivity tensor），

$$\Delta_{\times 1,\times 1}^{(l),(\times T),(\times C_l)} = \left\{\delta_{1,1}^{(l),(t),(k')}\right\}\Big|_{\substack{t=0,\cdots,T-1 \\ k'=0,\cdots,C_l-1}} = \left[\delta_{\times 1,\times 1}^{(l),(0),(\times C_l)} \quad \cdots \quad \delta_{\times 1,\times 1}^{(l),(T-1),(\times C_l)}\right]_{1\times T}$$

需要的输入数据有两部分，其一是前面已经得到的 Softmax 前向计算的输出，即网络 r 结点的激活值数据块（activation tensor）：

$$X_{\times 1,\times 1}^{(r),(\times T),(\times C_r)} = \left\{x_{1,1}^{(r),(t),(k)}\right\}\Big|_{\substack{t=0,\cdots,T-1 \\ k=0,\cdots,C_r-1}} = \left[X_{\times 1,\times 1}^{(r),(0),(\times C_r)} \quad \cdots \quad X_{\times 1,\times 1}^{(r),(T-1),(\times C_r)}\right]_{1\times T}$$

该数据块为 T 个 C_r 维向量，仍然有 $C_r = C_l$，每个向量都已经满足归一化条件了。

其二是目标监督数据块（target tensor），

$$Z_{\times 1,\times 1}^{(r),(\times T),(\times C_r)} = \left\{z_{1,1}^{(r),(t),(k)}\right\}\Big|_{\substack{t=0,\cdots,T-1 \\ k=0,\cdots,C_r-1}} = \left[Z_{\times 1,\times 1}^{(r),(0),(\times C_r)} \quad \cdots \quad Z_{\times 1,\times 1}^{(r),(T-1),(\times C_r)}\right]_{1\times T}$$

与前面一样， T 为 mini-batch 输入样本的个数，该数据块实际为 T 个 C_r 维向量，其中第 t 个 C_r 维向量是与 Input 层相应第 t 个输入样本相对应的标签向量，这个向量 $Z_{\times 1,\times 1}^{(r),(t),(\times C_r)}$ 一般为 one fire vector 模式，就是向量中只有一个分量值为 1，其它分量值都为 0 的向量。

反向 Softmax 函数表达为 $Softmax_{BF}$：

$$\Delta_{\times 1,\times 1}^{(l),(\times T),(\times C_l)} = Softmax_{BF}\left(X_{\times 1,\times 1}^{(r),(\times T),(\times C_r)}, Z_{\times 1,\times 1}^{(r),(\times T),(\times C_r)}\right)$$

其中每个分量值的计算公式为（代价函数为交叉熵形式）：

$$\delta_{1,1}^{(l),(t),(k')} = -\left(z_{1,1}^{(r),(t),(k)} - x_{1,1}^{(r),(t),(k)}\right)_{k=k'} \qquad \begin{cases} t = 0,\cdots,T-1 \\ k' = 0,\cdots,C_l - 1 \\ C_r = C_l \end{cases}$$

或者写为向量形式，

$$\begin{bmatrix} \delta_{1,1}^{(l),(t),(0)} \\ \vdots \\ \delta_{1,1}^{(l),(t),(C_l-1)} \end{bmatrix}_{C_l\times 1} = -\left(\begin{bmatrix} z_{1,1}^{(r),(t),(0)} \\ \vdots \\ z_{1,1}^{(r),(t),(C_r-1)} \end{bmatrix}_{C_r\times 1} - \begin{bmatrix} x_{1,1}^{(r),(t),(0)} \\ \vdots \\ x_{1,1}^{(r),(t),(C_r-1)} \end{bmatrix}_{C_r\times 1}\right)_{C_r=C_l} \qquad t = 0,\cdots,T-1$$

这是全网络敏感值计算的源头。已经证明，在交叉熵代价函数的条件下，标签向量 $Z_{\times 1,\times 1}^{(r),(t),(\times C_r)}$ 只要满足归一化条件，

$$\sum_{k=0}^{C_r-1} z_{1,1}^{(r),(t),(k)} = 1 \qquad t = 0,\cdots,T-1$$

则敏感值计算的公式是一样的。

Entropy

离散随机变量

设有离散随机变量 X，其事件空间为 $\{X_1,\cdots,X_n\}$，其分布表达为 $x=\{x_1,\cdots,x_n\}'$，满足归一化条件，

$$x=\begin{bmatrix}x_1\\ \vdots\\ x_n\end{bmatrix} \qquad \begin{cases} i=1,\cdots,n \\ x_i\in[0,1]\end{cases} \qquad \sum_{i=1}^{n}x_i=1$$

信息增益量

对于某具体事件 X_i，定义其信息增益量为 $I(X_i)$，

$$I(X_i)=log\left(\frac{1}{x_i}\right)=-log(x_i) \qquad i=1,\cdots,n$$

信息增益量表达的是观察到具有概率 x_i 的事件 X_i 后，所获得的信息量的度量。如果对数函数以 2 为底，则信息量的单位为 bit。定性来说，小概率事件发生时给出的信息量更大。

熵

离散随机变量 X 的熵 $H(X)$ 定义为信息增益量的期望，

$$H(X)=E[I(X_i)]=\sum_{i=1}^{n}x_i\cdot log\left(\frac{1}{x_i}\right)=-\sum_{i=1}^{n}x_i\cdot log(x_i)$$

熵 $H(X)$ 表示每个事件（消息）所携带的信息的平均量。满足下式，

$$0\le H(X)\le log(n)$$

$H(X)=0$ 当且仅当某个事件 X_i 的概率 $x_i=1$ 时，而其它的概率均为 0，对应确定性的 X。$H(X)=1$ 当且仅当 X 的分布为均匀分布 $\begin{cases} i=1,\cdots,n \\ x_i=\frac{1}{n}\end{cases}$。

相对熵 KL 散度

设有两个离散随机变量 X 和 Y，具有相同的事件空间为 $\{X_1,\cdots,X_n\}=\{Y_1,\cdots,Y_n\}$，其分布分别为 $x=\{x_1,\cdots,x_n\}'$ 和 $y=\{y_1,\cdots,y_n\}'$，均满足归一化条件，

$$x=\begin{bmatrix}x_1\\ \vdots\\ x_n\end{bmatrix}\quad \begin{cases}i=1,\cdots,n\\ x_i\in[0,1]\end{cases}\quad \sum_{i=1}^{n}x_i=1 \qquad y=\begin{bmatrix}y_1\\ \vdots\\ y_n\end{bmatrix}\quad \begin{cases}i=1,\cdots,n\\ y_i\in[0,1]\end{cases}\quad \sum_{i=1}^{n}y_i=1$$

定义相对熵（Kullback-Leibler 散度）为

$$H_{KL}(X||Y)=\sum_{i=1}^{n}x_i\cdot log\left(\frac{x_i}{y_i}\right)$$

相对熵比较两个分布不同的程度，相对熵 $H_{KL}(X||Y)\geq 0$，当两个分布完全一致时，$x_i=y_i, i=1,\cdots,n$，有 $H_{KL}(X||Y)=0$。

交叉熵

设有两个离散随机变量 X 和 Y，具有相同的事件空间为 $\{X_1,\cdots,X_n\}=\{Y_1,\cdots,Y_n\}$，其分布分别为 $x=\{x_1,\cdots,x_n\}'$ 和 $y=\{y_1,\cdots,y_n\}'$，均满足归一化条件（同上），定义交叉熵为

$$H_{cross}(X||Y)=\sum_{i=1}^{n}x_i\cdot log\left(\frac{1}{y_i}\right)$$

结合前面的公式，可以得到关系式，

$$H_{cross}(X||Y)=H_{KL}(X||Y)+H(X)$$

事实上，

$$H_{KL}(X||Y)=\sum_{i=1}^{n}x_i\cdot log\left(\frac{x_i}{y_i}\right)=\sum_{i=1}^{n}x_i\cdot log(x_i)+\sum_{i=1}^{n}x_i\cdot log\left(\frac{1}{y_i}\right)$$

$$H_{KL}(X||Y)+\sum_{i=1}^{n}x_i\cdot log\left(\frac{1}{x_i}\right)=\sum_{i=1}^{n}x_i\cdot log\left(\frac{1}{y_i}\right)$$

$$H_{KL}(X||Y)+H(X)=H_{cross}(X||Y)$$

联合熵

设有两个离散随机变量 X 和 Y，具有不相同的事件空间,分别为 $\{X_1,\cdots,X_n\}$ 和 $\{Y_1,\cdots,Y_m\}$，其分布分别为 $x=\{x_1,\cdots,x_n\}'$ 和 $y=\{y_1,\cdots,y_m\}'$，均满足归一化条件，

$$x=\begin{bmatrix}x_1\\ \vdots\\ x_n\end{bmatrix}\quad \begin{cases}i=1,\cdots,n\\ x_i\in[0,1]\end{cases}\quad \sum_{i=1}^{n}x_i=1 \qquad y=\begin{bmatrix}y_1\\ \vdots\\ y_m\end{bmatrix}\quad \begin{cases}j=1,\cdots,m\\ y_j\in[0,1]\end{cases}\quad \sum_{j=1}^{m}y_j=1$$

其联合分布为，$p=\{p_{ij}\}_{\substack{i=1,\cdots,n\\ j=1,\cdots,m}}$，满足归一化条件，

$$\sum_{i=1}^{n}\sum_{j=1}^{m}p_{ij}=1$$

定义联合熵为

$$H(X,Y)=\sum_{i=1}^{n}\sum_{j=1}^{m}p_{ij}\,log\left(\frac{1}{p_{ij}}\right)$$

条件熵

设有两个离散随机变量 X 和 Y，具有不相同的事件空间,分别为 $\{X_1,\cdots,X_n\}$ 和 $\{Y_1,\cdots,Y_m\}$，其分布分别为 $x=\{x_1,\cdots,x_n\}'$ 和 $y=\{y_1,\cdots,y_m\}'$，均满足归一化条件，其联合分布为，$p=\{p_{ij}\}_{\substack{i=1,\cdots,n\\ j=1,\cdots,m}}$，也满足归一化条件，定义条件熵为

$$H(X|Y)=H(X,Y)-H(Y)$$

假设系统的输入为 X，输出为 Y，则条件熵表示观测到系统输出 Y 后，对 X 剩余不确定性的度量。

互信息

两个离散随机变量 X 和 Y，假设如上。熵 $H(X)$ 表示观测前的系统输入的不确定性，条件熵 $H(X|Y)$ 表示观测到系统输出 Y 后对系统输入的不确定性的度量，则差 $[H(X)-H(X|Y)]$ 表示观测前后系统输入不确定性的减少，称这个量为随机变量 X 和 Y 之间的互信息，

$$I(X;Y)=H(X)-H(X|Y)$$

$$I(X;Y)=H(X)-H(X|Y)=H(X)+H(Y)-H(X,Y)=\sum_{i=1}^{n}\sum_{j=1}^{m}p_{ij}\,log\left(\frac{p_{ij}}{x_i\cdot y_j}\right)$$

证明：

$$I(X;Y)=\sum_{i=1}^{n}x_i\cdot log\left(\frac{1}{x_i}\right)+\sum_{j=1}^{m}y_j\cdot log\left(\frac{1}{y_j}\right)-\sum_{i=1}^{n}\sum_{j=1}^{m}p_{ij}\,log\left(\frac{1}{p_{ij}}\right)$$

$$note: \qquad \sum_{j=1}^{m} p_{ij} = x_i \quad (i = 1, \cdots, n) \qquad \sum_{i=1}^{n} p_{ij} = y_j \quad (j = 1, \cdots, m)$$

$$I(X;Y) = \sum_{i=1}^{n} \left(log\left(\frac{1}{x_i}\right) \cdot \sum_{j=1}^{m} p_{ij} \right) + \sum_{j=1}^{m} \left(log\left(\frac{1}{y_j}\right) \cdot \sum_{i=1}^{n} p_{ij} \right) + \sum_{i=1}^{n} \sum_{j=1}^{m} p_{ij}\, log(p_{ij})$$

$$I(X;Y) = \sum_{i=1}^{n} \sum_{j=1}^{m} p_{ij}\, log\left(\frac{1}{x_i}\right) + \sum_{i=1}^{n} \sum_{j=1}^{m} p_{ij}\, log\left(\frac{1}{y_j}\right) + \sum_{i=1}^{n} \sum_{j=1}^{m} p_{ij}\, log(p_{ij})$$

$$I(X;Y) = \sum_{i=1}^{n} \sum_{j=1}^{m} p_{ij}\, log\left(\frac{p_{ij}}{x_i \cdot y_j}\right)$$