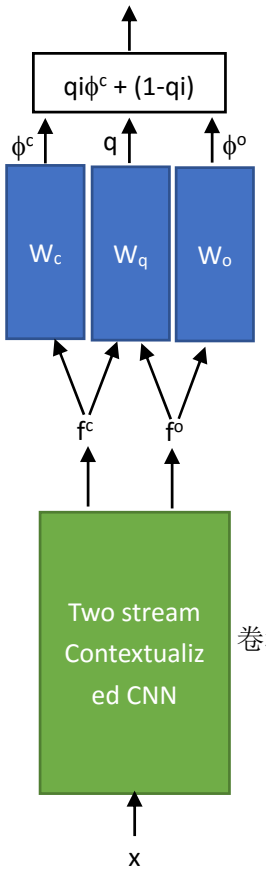
Learning process of two-stream network:



三个独立网络（单全连接层(单个隐层)+softmax 输出层结构）：W_c、W_q、W_o 分别作为 context classifier， pooling parameter regression，object classifier

卷积层

1. 主要变量

x (x_i) : (第 i 个)输入图像

y (y_i) : (第 i 个)输入图像的类别标签(1*n 的 0/1 vector)

$f^c = F(x, M_c; W)$: context feature after SPPnet

$f^o = F(x, M_o; W)$: object feature after SPPnet

$q_i = Q(f^c, f^o, W_q)$ $(0 \le q_i \le 1)$: weight of the context classifier for the i-th image.

$\phi^c = \Phi_c(f^c, W_c)$: context classifier（概率化输出 1*n 的向量）

$\phi^o = \Phi_o(f^o, W_o)$: object classifier（概率化输出 1*n 的向量）

2. 损失函数:

基于 logistic 模型，定义：

$$\pi(x_i) = \frac{1}{1 + \exp(-(q_i\phi_i^c + (1 - q_i)\phi_i^o))}$$

则损失函数可以定义为：

$$L = \sum_i y_i \ln\pi(x_i) + (1 - y_i)\ln(1 - \pi(x_i))$$

我们将上述模型修改为 multi-class 版本，定义第 i 张图片属于第 j 个 class 的 probability:

$$\pi(x_i, j) = \frac{1}{1 + \exp(-(q_i\phi_{ij}^c + (1 - q_i)\phi_{ij}^o))}$$

为后期计算方便，我们令：$u(x_i,j)=q_i\phi_{ij}^c+(1-q_i)\phi_{ij}^o$

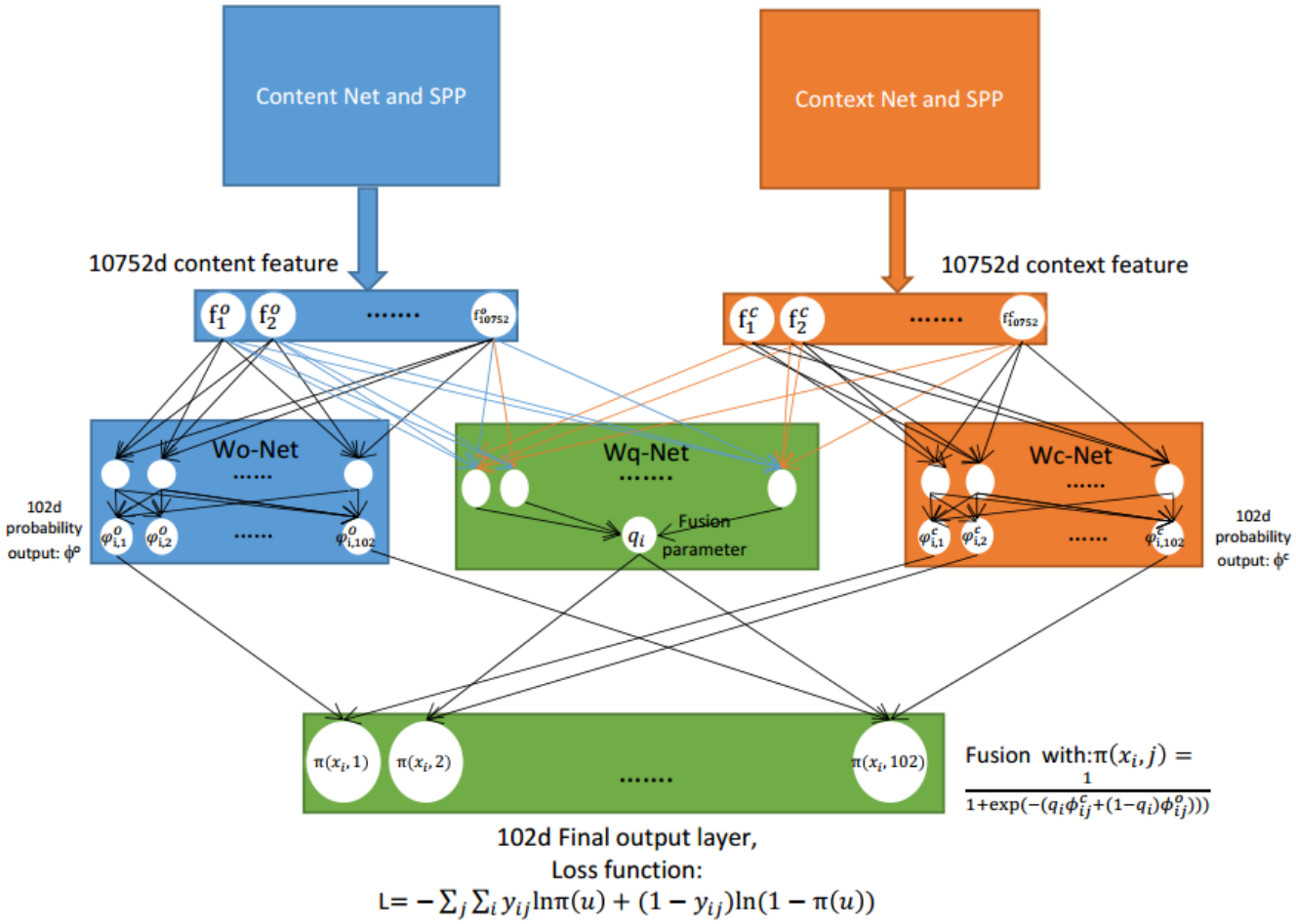
因此，

$$\pi(x_i,j)=\pi(u(x_i,j))=\frac{1}{1+\exp(-u(x_i,j))}$$

则损失函数可以定义为(我们将原来的 cross-entropy loss function 修改为多个 class)

$$L=-\frac{1}{N}\sum_j\sum_i y_{ij}\ln\pi(u(x_i,j))+(1-y_{ij})\ln(1-\pi(u(x_i,j)))$$

SPP Operation 后的融合模型网络示意图如下：f^o和f^c 分别为 content feature 和 context feature，我们分别构造一个包含一个全连接层、一个 softmax 输出层的网络 Wo-Net 和 Wc-Net，作为 content classifier 和 context classifier。同时构造单节点 logistic 函数输出的 Wq-Net 作为权重输出，最终通过$\pi(x_i,j)$，进行 feature 融合，同时 BP 算法反向计算梯度，更新三个网络。



3. 学习方法

STEP 1:初始化： Wc，Wo，Wq，参数随机取值；

STEP 2: Feed-forward: mini-batch：输入 N 张训练集图片的 context 和 content SPP Feature 到 Wc，Wq，Wo，得到：

φ_{ij}^c，φ_{ij}^o，q_i，L

STEP 3: BP 算法，更新 W_c、W_o

为了更新 Wc，Wo，我们可以使用 matconvnet 中封装好的模块计算更新参数，需要输入自定义的 Loss function 对 Wc，Wo 输出参数的梯度，即：$\frac{\partial L}{\partial\varphi_{ij}^c}$，$\frac{\partial L}{\partial\varphi_{ij}^o}$

根据自定义的 loss function 逐层计算梯度：

$$\frac{\partial \mathrm{L}}{\partial \pi}=-\frac{1}{N}\sum_{j}\sum_{i}\frac{y_{ij}}{\pi(x_i,j)}-\frac{1-y_{ij}}{1-\pi(x_i,j)} \qquad \frac{\partial \pi}{\partial \mathrm{u}}=\pi(\mathrm{u}(x_i,j))\big(1-\pi(\mathrm{u}(x_i,j))\big)$$

又因为：

$$\frac{\partial \mathrm{u}(x_i,j)}{\partial \varphi_{ij}^{c}}=q_i \qquad \frac{\partial \mathrm{u}(x_i,j)}{\partial \varphi_{ij}^{o}}=1-q_i$$

所以： $\frac{\partial \mathrm{L}}{\partial \varphi_{ij}^{c}}=\frac{\partial \mathrm{L}}{\partial \pi}\frac{\partial \pi}{\partial \mathrm{u}}\frac{\partial \mathrm{u}}{\partial \varphi_{ij}^{c}}=\left[-\frac{1}{N}\sum_j\sum_i\frac{y_{ij}}{\pi(x_i,j)}-\frac{1-y_{ij}}{1-\pi(x_i,j)}\right]\left[\pi(u(x_i,j))\big(1-\pi(u(x_i,j))\big)\right][q_i]$

Similarly: $\frac{\partial \mathrm{L}}{\partial \varphi_{ij}^{o}}=\frac{\partial \mathrm{L}}{\partial \pi}\frac{\partial \pi}{\partial \mathrm{u}}\frac{\partial \mathrm{u}}{\partial \varphi_{ij}^{o}}=\left[-\frac{1}{N}\sum_j\sum_i\frac{y_{ij}}{\pi(x_i,j)}-\frac{1-y_{ij}}{1-\pi(x_i,j)}\right]\left[\pi(u(x_i,j))\big(1-\pi(u(x_i,j))\big)\right][1-q_i]$

下面用 matconvnet 自带 BP 更新 softmax 和 fully connected layer,:Wc, Wo，输入$\frac{\partial \mathrm{L}}{\partial \varphi_{ij}^{c}}$ $\frac{\partial \mathrm{L}}{\partial \varphi_{ij}^{o}}$，即可。

STEP 4: 固定 W_c、W_o，BP 算法更新 qi(W_q):

同样，Wq 内部网络部分可以使用 matconvnet toolkit 实现，也需要$\frac{\partial \mathrm{L}}{\partial q_i}$ 作为输入：

因为：

$$\frac{\partial \mathrm{u}}{\partial q_i}=\varphi_{ij}^{c}-\varphi_{ij}^{o}$$

所以: $\frac{\partial \mathrm{L}}{\partial q_i}=\frac{\partial \mathrm{L}}{\partial \pi}\frac{\partial \pi}{\partial \mathrm{u}}\frac{\partial \mathrm{u}}{\partial q_i}=\left[-\frac{1}{N}\sum_j\sum_i\frac{y_{ij}}{\pi(x_i,j)}-\frac{1-y_{ij}}{1-\pi(x_i,j)}\right]\left[\pi(u(x_i,j))\big(1-\pi(u(x_i,j))\big)\right]\left[\varphi_{ij}^{c}-\varphi_{ij}^{o}\right]$

BP 更新 softmax 和 fully connected layer:Wq.输入：$\frac{\partial \mathrm{L}}{\partial q_i}$

STEP 5:迭代进行 STEP2-STEP4 直到 Loss function L 收敛。