1 Training and Inference of Semi-CRFs

In this section, we show more details about the training and inference of Semi-CRFs following the settings we made in the main paper.

1.1 Training of Semi-CRF-based Parameters

Given training data, all the parameters of grSemi-CRFs can be learnt by maximizing log likelihood, i.e., $\mathcal{L} = \log p(s|x)$. To simplify representations, we introduce some auxiliary notations, including $g(h_j, d_j, y_{j-1}, y_j) = F(s_j, x) + A(y_{j-1}, y_j)$ and $G(s, x) = \sum_{j=1}^{|s|} g(h_j, d_j, y_{j-1}, y_j)$. Then the likelihood can be rewritten as $p(s|x) = \frac{1}{Z(x)} \exp(G(s, x))$ where the normalization factor $Z(x) = \sum_y \exp(G(s', x))$.

We further define

$$\alpha_{y,t} = \log \sum_{s' \in \mathcal{C}_{t+1},y} \exp(G(s', x)),$$

where $s_{1:k,y}$ denotes all segmentations for $(x_1, ..., x_T)$ with $y$ being the tag of the ending segment. And we also define

$$\beta_{y,k} = \log \sum_{s' \in \mathcal{C}_{k+1:T},y} \exp(G(s', x)),$$

where $s_{k:T,y}$ denotes all segmentations for $(x_{k+1}, ..., x_T)$ with $y$ being the tag of the segment which contains $x_k$.

Then, by using a Semi-CRF version of forward-backward algorithms, we can compute $\alpha_{y,k}$ and $\beta_{y,k}$ iteratively, i.e.,

$$\alpha_{y,k} = \log \sum_{d=1}^L \sum_{y' \in \mathcal{Y}} \exp \left( \alpha_{y',k-d} + g(k - d + 1, d, y', y) \right),$$

$$\beta_{y,k} = \log \sum_{d=1}^L \sum_{y' \in \mathcal{Y}} \exp \left( \beta_{y',k+d} + g(k + 1, d, y', y') \right),$$

where the boundary conditions are setted as $\alpha_{y,k} = 0$ for $k \leq 0$ and $\beta_{y,k} = 0$ for $k \geq T$.

Then, the normalization factor $Z(x)$ can be denoted as

$$Z(x) = \sum_{y \in \mathcal{Y}} \exp(\alpha_{y,k}),$$

and corresponding partial derivative is

$$\frac{\partial Z(x)}{\partial g(k, d, y', y)} = \frac{1}{Z(x)} \exp(\alpha_{y',k-d} + g(k, d, y', y') + \beta_{y,k}).$$

Thus, the derivative of the objective function is

$$\frac{\partial \mathcal{L}}{\partial g(k, d, y', y)} = \sum_{j=1}^{|s|} \mathbb{I}(s_j = (k, d, y), y_{j-1} = y') - \frac{\partial Z(x)}{\partial g(k, d, y', y)},$$

where $\mathbb{I}(\cdot)$ is the indicator function. Then, we can easily compute gradients for Semi-CRF-based parameters, i.e.,

$$\frac{\partial \mathcal{L}}{\partial A(y', y)} = \sum_{d=1}^L \sum_{k=0}^T \frac{\partial \mathcal{L}}{\partial g(k, d, y', y)},$$

$$\frac{\partial \mathcal{L}}{\partial V(0)_{y,j}} = \sum_{d=1}^T \sum_{k=0}^T \sum_{y' \in \mathcal{Y}} \frac{\partial \mathcal{L}}{\partial g(k, d, y', y')} \mathcal{C}_{k+1},$$

and

$$\frac{\partial \mathcal{L}}{\partial F(s_{(d)}^y)} = \sum_{y' \in \mathcal{Y}} \frac{\partial \mathcal{L}}{\partial g(k, d, y', y')} \mathcal{C}_{k+1},$$

where $\mathcal{C}_{k+1}$ is the $y$th entry of the length-$|\mathcal{Y}|$ vector $\frac{\partial \mathcal{L}}{\partial F(s_{(d)}^y)}$.

1.2 Training of grConv Parameters

Thanks to the recursive structure, the backpropagated gradients follow

$$\frac{\partial \mathcal{L}}{\partial z_k^{(d)}} = \frac{\partial \mathcal{L}}{\partial z_k^{(d+1)}} \frac{\partial \mathcal{L}}{\partial z_k^{(d+1)}} + \frac{\partial \mathcal{L}}{\partial z_k^{(d+1)}} \frac{\partial \mathcal{L}}{\partial z_k^{(d+1)}} + V(0)^T \frac{\partial \mathcal{L}}{\partial F(s_k^{(d)})},$$

where $E = \mathbb{I}(E)$ is 1 when condition $E = \text{true}$ and $\mathbb{I}(E) = 0$ when condition $E = \text{false}$. 

---

* This work was done when J.W.Z was on an internship with Microsoft Research.

† J.Z is the corresponding author.
where

\[
\frac{\partial z_k^{(d+1)}}{\partial \theta_k^{(d)}} = \text{diag}(\theta_L) + \text{diag}(\theta_M \circ g'(\alpha_k^{(d+1)})) W_L,
\]

\[
\frac{\partial z_k^{(d+1)}}{\partial \theta_k^{(d)}} = \text{diag}(\theta_R) + \text{diag}(\theta_M \circ g'(\alpha_k^{(d+1)})) W_R,
\]

and \( \frac{\partial \mathcal{L}}{\partial F(s_k^{(d)}, x)} \) is computed in Eq. (10).

Embeddings can be learnt using \( \frac{\partial \mathcal{L}}{\partial \theta_k^{(0)}} \) as grSemi-CRFs use embeddings as length-1 segment-level features directly.

For \( W_L \), we can compute the local partial derivative first, i.e.,

\[
\left[ \frac{\partial z_k^{(d)}}{\partial W_L} \right]_{i,j} = \theta_{M,i} g'(\alpha_{k,i}^{(d)}) z_{k,j}^{(d-1)}. \tag{13}
\]

Thus we have

\[
\frac{\partial \mathcal{L}}{\partial W_L} = \sum_{d=1}^{L} \sum_{k=1}^{T-d+1} \left[ \theta_M \circ g'(\alpha_k^{(d)}) \circ \frac{\partial \mathcal{L}}{\partial z_k^{(d)}} \right] z_k^{(d-1)T}. \tag{14}
\]

The gradients for \( W_R \) and \( b_W \) can be computed in almost the same ways.

For \( G_L \), the local partial derivative can be denoted as

\[
\left[ \frac{\partial z_k^{(d)}}{\partial G_L} \right]_{D \times t+i,j} = \frac{\partial \theta_L}{\partial G_L} \epsilon_{k,i} z_{k,j}^{(d-1)} + \frac{\partial \theta_R}{\partial G_L} \epsilon_{k,i} z_{k,j}^{(d)} + \frac{\partial \theta_M}{\partial G_L} \epsilon_{k,i} z_{k,j}^{(d-1)} + \frac{\partial \theta_{L,i}}{\partial G_L} \epsilon_{k,i} z_{k,j}^{(d-1)}. \tag{15}
\]

Notice that \( G_L \in \mathbb{R}^{3D \times D} \) has 3D rows where \( \theta_{L,i}, \theta_{R,i}, \) and \( \theta_{M,i} \) corresponds to the \( i \)th, \((D+i)\)th, and \((2D+i)\)th rows of \( G_L \). With a little abuse of notations (i.e., we use \( \ell \) to denote numbers 0, 1, 2 corresponding to rows of \( G_L \), and characters \( L, R, M \) corresponding to the gating coefficients),

\[
\left[ \frac{\partial \theta_L}{\partial G_L} \right]_{D \times t+i,j} = \theta_{L,i} \epsilon_{k,j}^{(d-1)} (\ell = L - \theta_{L,i}),
\]

\[
\left[ \frac{\partial \theta_R}{\partial G_L} \right]_{D \times t+i,j} = \theta_{R,i} \epsilon_{k,j}^{(d-1)} (\ell = R - \theta_{R,i}),
\]

\[
\left[ \frac{\partial \theta_M}{\partial G_L} \right]_{D \times t+i,j} = \theta_{M,i} \epsilon_{k,j}^{(d-1)} (\ell = M - \theta_{M,i}). \tag{16}
\]

Finally, we have,

\[
\left[ \frac{\partial \mathcal{L}}{\partial G_L} \right]_{D \times t+i,j} = \sum_{d=1}^{L} \sum_{k=1}^{T-d+1} \frac{\partial \mathcal{L}}{\partial z_k^{(d)}} \frac{\partial z_k^{(d)}}{\partial G_L} \epsilon_{k,i} z_{k,j}^{(d-1)}. \tag{17}
\]

The gradients for \( G_R \) and \( b_G \) can be computed in a similar way.

1.3 Inference of grSemi-CRFs

The inference problem is, given parameters and \( x \), find the best tag segmentation \( s^* = \arg\max_s \log p(s|x) = \arg\max_s \sum_{j=1}^{|s|} G(s, x). \) This can be solved by using a Semi-Markov version of the Viterbi algorithm. We use \( V_{y,k} \) to denote the maximum value for \( \sum_{s' \in S_{1:k,y}} G(s', x). \) Then the update equation is, for \( i > 0, \)

\[
V_{y,k} = \max_{y' \in Y, d=1,...,L} V_{y',k-d} + g(k-d+1, d, y', y). \tag{18}
\]

For the boundary case, we set \( V_{y,k} = 0 \) for \( k \leq 0. \) Finally, the best segmentation \( s^* \) corresponds to the path traced by \( \max_{y \in Y} V_{y,T}. \)