Gradient for Multi-class Hinge Loss

Given an input feature $\mathbf{x}$ of size 1-by-D, we can compute $\mathbf{y} = \mathbf{x}\mathbf{W} + \mathbf{b}$, where $\mathbf{W}$ is the D-by-C matrix of weights, $\mathbf{b}$ is a C dimensional vector and $\mathbf{y}$ is a vector of $\mathbf{x}$'s class scores . The multi-class Hinge-loss over N training samples can be computed as:

$$L = \frac{1}{N}\sum_{i=1}^{N}\sum_{j\neq y_i}^{C}\max(0, \mathbf{w}_j\mathbf{x}_i + \mathbf{b}_j - (\mathbf{w}_{y_i}\mathbf{x}_i + \mathbf{b}_{y_i}) + 1) \tag{1}$$

where y_i is the correct class index for $\mathbf{x}_i$.
Question: compute $\frac{\partial L}{\partial \mathbf{W}}$ and $\frac{\partial L}{\partial \mathbf{b}}$.
Solution: For simplicity, let's write

$$L_i = \mathbf{w}_j\mathbf{x}_i + \mathbf{b}_j - (\mathbf{w}_{y_i}\mathbf{x}_i + \mathbf{b}_{y_i}) + 1 \tag{2}$$

Now we can simplify equation 1 into

$$L = \frac{1}{N}\sum_{i=1}^{N}\sum_{j\neq y_i}^{C} L_i \tag{3}$$

Notice L_j is a scaler (the same as L), not a vector. We see that

$$\frac{\partial L}{\partial \mathbf{W}} = \frac{1}{N}\sum_{i=1}^{N}\sum_{j\neq y_i}^{C}\frac{\partial L_j}{\partial \mathbf{W}} = \frac{1}{N}\sum_{i=1}^{N}\sum_{j\neq y_i}^{C}(\frac{\partial L_j}{\partial \mathbf{W}_1}, \frac{\partial L_j}{\partial \mathbf{W}_2}, ..., \frac{\partial L_j}{\partial \mathbf{W}_C}) \tag{4}$$

Notice $\frac{\partial L_j}{\partial \mathbf{W}}$ is a D-by-C matrix (the same as $\mathbf{W}$), and $\frac{\partial L_j}{\partial \mathbf{W}_j}$ is D dimentional vector (a colomn of $\frac{\partial L_j}{\partial \mathbf{W}}$).
Let's compute $\frac{\partial L_j}{\partial \mathbf{W}}$ for a single data $\mathbf{x}_i$. There are two cases:

$$\frac{\partial L_j}{\partial \mathbf{W}} = \begin{cases} (\mathbf{0}, ..., \mathbf{0}, \mathbf{x}_i^T, \mathbf{0}, ..., -\mathbf{x}_i^T, \mathbf{0}, ..., \mathbf{0}) & \text{if } L_j > 0, \\ \mathbf{0} & \text{if } L_j = 0. \end{cases} \tag{5}$$

The $\mathbf{x}_i^T$ term in the first case correponses to $\frac{\partial L_j}{\partial \mathbf{W}_j}$, and the $-\mathbf{x}_i^T$ term in the first case correponses to $\frac{\partial L_j}{\partial \mathbf{W}_{y_i}}$. This is from Equation 2. Now we have the fomula of $\frac{\partial L_j}{\partial \mathbf{W}}$. To compute $\frac{\partial L}{\partial \mathbf{W}}$ we just loop through all the N tranining data, and for each of them the C classes (apart from the correct class).
Similarly, we have the:

$$\frac{\partial L}{\partial \mathbf{b}} = \frac{1}{N}\sum_{i=1}^{N}\sum_{j\neq y_i}^{C}\frac{\partial L_j}{\partial \mathbf{b}} = \frac{1}{N}\sum_{i=1}^{N}\sum_{j\neq y_i}^{C}(\frac{\partial L_j}{\partial b_1}, \frac{\partial L_j}{\partial b_2}, ..., \frac{\partial L_j}{\partial b_C}) \tag{6}$$

where

$$\frac{\partial L_j}{\partial \mathbf{b}} = \begin{cases} (0, ..., 0, 1, 0, ..., -1, 0, ..., 0) & \text{if } L_j > 0, \\ \mathbf{0} & \text{if } L_j = 0. \end{cases} \tag{7}$$

See the "svm_loss_bias_forloop" function in "linear_svm.py" for implementation. Notice the regularization term $\frac{1}{2}||\mathbf{W}||^2$ is not included in this note, but used in "svm_loss_bias_forloop". Make sure you understand how does this regularization term influnce the loss and the gradient.

The first task for this week's practice is to implement the same function with bias trick: that is to treat b as part of $\mathbf{W}$, and write $\mathbf{y} = \mathbf{x}\mathbf{W}$ with an additional one appended to $\mathbf{x}$.