Homework1 Backprop

Rakesh Kumar

December 21, 2022

---

1 Theory

1.1 Two-Layer Neural Nets

$Linear_1$ → f → $Linear_2$ → g where $Linear_i(x) = \mathbf{W^{(i)}x} + \mathbf{b^{(i)}}$ is the i-th tranformation, and f, g are elementwise nonlinear activation functions. where input $\mathbf{x} \epsilon \mathbb{R}^n$ and $\hat{y} \epsilon \mathbb{R}^n$.

1.2 Regression Task

$f(.) = (.)^+ = ReLU(.)$ and g is to be identity function.
MSE loss function $l_{MSE}(y, \hat{y}) = ||\hat{y} - y||^2$ where y is the target output.

1. Name and mathematically define the 5 programming steps to train the above model architecture with PyTorch using SGD on a single batch of data.
   Solution:
   step 1. Feed Forward to get the logits.

$$y_{pred} = model(X) \tag{1}$$

   $\mathbf{x} \, \epsilon \, \mathbb{R}^n \rightarrow \mathbf{h} \epsilon \, \mathbb{R}^d \rightarrow \mathbf{\hat{y}} \, \epsilon \, \mathbb{R}^k$
   Where:

$$\mathbf{h} = f(\mathbf{W_h x} + \mathbf{b_h}) \tag{2}$$
$$\mathbf{\hat{y}} = g(\mathbf{W_y h} + \mathbf{b_y}) \tag{3}$$

   $\mathbf{W_h} \, \epsilon \, \mathbb{R}^{dxn}$, $\mathbf{b_h} \, \epsilon \, \mathbb{R}^d$, $\mathbf{W_y} \, \epsilon \, \mathbb{R}^{Kxd}$, $\mathbf{b_y} \, \epsilon \, \mathbb{R}^K$

step 2. Compute the loss using different criterion.

$$loss = criterion(y_{pred}, y) \tag{4}$$

Example of loss function: $l_{MSE}(y, \hat{y}) = \frac{1}{n} \sum ||y_{pred} - y||^2$

Cross entropy or neagtive likelihood:

$$\mathrm{L}(\hat{Y}, c) = \frac{1}{m} \sum_{i=1}^{m} (l(\hat{y^(i)}, c_i)) \tag{5}$$

Where: $l(\hat{y^(i)}, c_i) = -log(\hat{y}[c])$

step 3. Zeroing the gradients before going through the backward pass. $optimizer.zero\_grad()$

step 4. Fourth step after zeroing the gradients, is Bakward pass to compute the gradient of loss w.r.t our learnable parameters.

$$\Theta = (\mathbf{W_h}\,, \mathbf{b_h}\,, \mathbf{W_y}\,, \mathbf{b_y}) \tag{6}$$

$$\mathrm{J}(\Theta) = \mathrm{L}(\hat{Y}(\Theta), c) \,\epsilon\, \mathbb{R}^+ \tag{7}$$

$$\frac{\partial \mathrm{J}(\Theta)}{\partial \mathbf{W_y}} = \frac{\partial \mathrm{J}(\Theta)}{\partial \hat{\mathbf{y}}} \frac{\partial \hat{\mathbf{y}}}{\partial \mathbf{W_y}} \tag{8}$$

$$\frac{\partial \mathrm{J}(\Theta)}{\partial \mathbf{W_h}} = \frac{\partial \mathrm{J}(\Theta)}{\partial \hat{\mathbf{y}}} \frac{\partial \hat{\mathbf{y}}}{\partial \mathbf{h}} \frac{\partial \mathbf{h}}{\mathbf{W_h}} \tag{9}$$

step 5. Updating the parameters.
Based on the SGD optimization using backprop we update the learnable parameters.

2. Write doen the forward pass of each layer:

Layer	Input	Output
$Linear_1$	$\mathbf{x}, \mathbf{W^1}, \mathbf{b^1}$	$\mathbf{z}_1 = \mathbf{W^1x} + \mathbf{b^1}$
f	$\mathbf{W^1x} + \mathbf{b^1}$	$\mathbf{z}_2 = f(\mathbf{W^1x} + \mathbf{b^1})$
$Linear_2$	$\mathbf{z}_2, \mathbf{W^2}, \mathbf{b^2}$	$\mathbf{z}_3 = \mathbf{W^2z}_2 + \mathbf{b^2}$
g	$\mathbf{W^2h} + \mathbf{b_y}$	$\hat{\mathbf{Y}} = g(\mathbf{W^2h} + \mathbf{b^2})$
Loss	$\hat{\mathbf{Y}}, \mathbf{c}$	$\mathrm{L}(\hat{Y}(\Theta), c)$

3. Gradient calulated from the backward pass:

Parameters	Gradient
$\mathbf{W}^1$	
$\mathbf{b}^1$	
$\mathbf{W}^2$	
$\mathbf{b}^2$	