CPSC 340: Machine Learning and Data Mining

Deep Learning

BONUS SLIDES

Backpropagation

- Consider the loss for a single example:

$$f(w, W) = \frac{1}{2}\left(\sum_{c=1}^{k} w_c\, h(W_c\, x_i) - y_i\right)^2$$

Element 'c' of 'w'

Row 'c' of W

- Derivative with respect to 'w_c':

From squared loss

$$\frac{\partial}{\partial w_c}\left[f(w,W)\right] = \left(\sum_{c=1}^{k} w_c\, h(W_c x_i) - y_i\right) h(W_c x_i)$$

derivative with respect to w_c

- Derivative with respect to 'W_{cj}'

derivative with respect to W_{cj}

$$\frac{\partial}{\partial W_{cj}}\left[f(w,W)\right] = \left(\sum_{c=1}^{k} w_c\, h(W_c x_i) - y_i\right) \underbrace{w_c\, h'(W_c x_i)}\, x_{ij}$$

derivative with respect to $W_c x_i$

Backpropagation

- Notice repeated calculations in gradients:

$$\frac{\partial}{\partial w_c}\left[f(w,W)\right] = \underbrace{\left(\sum_{c=1}^{k} w_c h(W_c x_i) - y_i\right)}_{r_i} h(W_c x_i)$$

$$= \underline{r_i}\, h(W_c x_i)$$

same r_i for all 'c'

$$\frac{\partial}{\partial W_{cj}}\left[f(w,W)\right] = \underbrace{\left(\sum_{c=1}^{k} w_c h(W_c x_i) - y_i\right)}_{r_i} \underbrace{w_c h'(W_c x_i)}_{v_c} x_{ij}$$

$$= \underline{r_i}\, \underline{v_c}\, x_{ij}$$

Same v_c for all 'j'

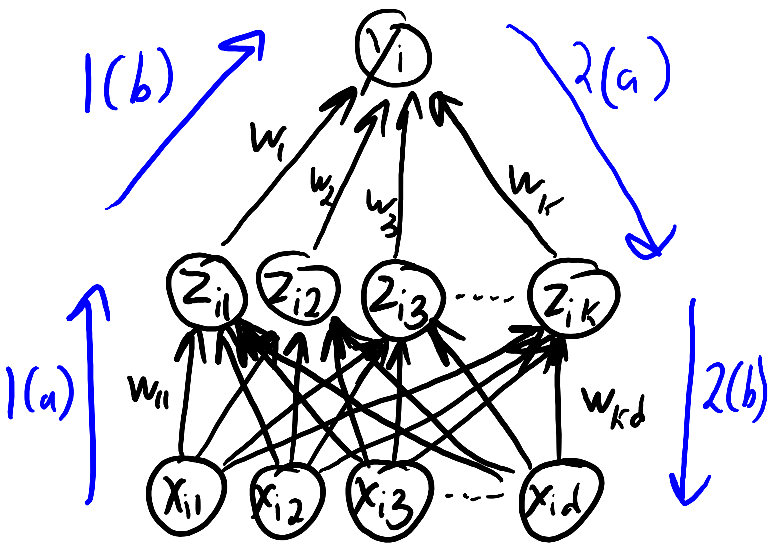
same r_i for all 'c')

Backpropagation

- Calculation of gradient is split into two phases:

1. "Forward" pass
   (a) Compute $h(W_c x_i)$ for all 'c'
   (b) Compute residual $r_i = (\sum_{c=1}^{k} w_c h(W_c x_i) - y_i)$

2. "Backprogation"
   (a) Compute $\frac{\partial f}{\partial w_c} = r_i h(W_c x_i)$ for all 'c'
   (b) Compute $v_c = w_c h'(W_c x_i)$ for all 'c'
   (c) Compute $\frac{\partial f}{\partial W_{cj}} = r_i v_c x_{ij}$ for all 'c' and 'j'



Last Time: Backpropagation with 1 Hidden Layer

- Squared loss our objective function with 1 layer and 1 example:

$$f(w, W) = \frac{1}{2}\left(\sum_{c=1}^{k} w_c h(W_c x_i) - y_i\right)^2$$

- Gradient with respect to element of vector 'w'.

$$\frac{\partial}{\partial w_c}\left[f(w, W)\right] = \underbrace{\left(\sum_{c=1}^{k} w_c h(W_c x_i) - y_i\right)}_{r_i} h(W_c x_i) = r_i h(W_c x_i)$$

- Gradient with respect to element of matrix 'W'.

$$\frac{\partial}{\partial W_{cj}}\left[f(w, W)\right] = \underbrace{\left(\sum_{c=1}^{k} w_c h(W_c x_i) - y_i\right)}_{r_i} \underbrace{w_c h'(W_c x_i)}_{v_c} x_{ij} = r_i v_c x_{ij}$$

- Only r_i changes if you aren't using squared error.

Last Time: Backpropagation with 1 Hidden Layer

- Squared loss our objective function with 1 layer and 1 example:

$$f(w, W) = \frac{1}{2}\left(\sum_{c=1}^{k} w_c h(W_c x_i) - y_i\right)^2$$

- Gradient with respect to elements of vector 'w' and 'W':

$$\frac{\partial}{\partial w_c}\left[f(w,W)\right] = r_i h(W_c x_i) \qquad \frac{\partial}{\partial W_{cj}}\left[f(w,W)\right] = r_i v_c x_{ij}$$

with $r_i = \sum_{c=1}^{k} w_c h(W_c x_i) - y_i$

and $v_c = w_c h'(W_c x_i)$

- Backpropagation algorithm:
  - Forward propagation computes $z_i = h(W_c x_i)$ then $w^T z_i$.
  - Backpropagation step 1: use r_i to get gradient of 'w'.
  - Backpropagation step 2: use r_i and v_c to get gradient of 'W'.

Backpropagation with 2 Hidden Layer

- General objective function with 2 layers and 1 example:

$$f(w, W^{(2)}, W^{(1)}) = f_i(w^T h(W^{(2)} h(W^{(1)} x_i)))$$

- Gradient with respect to element of vector 'w':

$$\frac{\partial}{\partial w_c}[f(w, W^{(2)}, W^{(1)})] = \underbrace{f_i'(w^T h(W^{(2)} h(W^{(1)} x_i)))}_{r} \underbrace{h(W_c^{(2)} h(W^{(1)} x_i))}_{\to z_c^{(2)}} = r z_c^{(2)}$$

- Gradient with respect to element of matrix 'W$^{(2)}$':

$$\frac{\partial}{\partial W_{cj}^{(2)}}[f(w, W^{(2)}, W^{(1)})] = \underbrace{f_i'(w^T h(W^{(2)} h(W^{(1)} x_i)))}_{r} \underbrace{w_c h'(W_c^{(2)} h(W^{(1)} x_i))}_{v_c} \underbrace{h(W_j^{(1)} x_i)}_{\to z_j^{(1)}} = r v_c z_j^{(1)}$$

- Gradient with respect to element of matrix 'W$^{(1)}$':

$$\frac{\partial}{\partial W_{cj}^{(1)}}[f(w, W^{(2)}, W^{(1)})] = \underbrace{f_i'(w^T h(W^{(2)} h(W^{(1)} x_i)))}_{r} \sum_{c'=1}^{k} \left[\underbrace{w_{c'} h'(W_{c'}^{(2)} h(W^{(1)} x_i))}_{v_{c'}} \underbrace{W_{c'c}^{(2)} h'(W_c^{(1)} x_i)}_{u_{cc'}} x_{ij}\right] = r(v^T u_c) x_{ij}$$

Last Time: Backpropagation with 3 Hidden Layers

- General objective function with 3 layers and 1 example:

$$f(w, W^{(3)}, W^{(2)}, W^{(1)}) = f_i(w^T h(W^{(3)} h(W^{(2)} h(W^{(1)} x_i))))$$

- Gradients have the form:

$$\frac{\partial f}{\partial w_c} = r z_c^{(3)} \qquad \frac{\partial f}{\partial W^{(3)}_{cc'}} = \underbrace{r v_c}_{} z_{c'}^{(2)} = a_c z_{c'} \qquad \frac{\partial f}{\partial W^{(2)}_{cc'}} = r(v^T u_c) z_{c'}^{(1)} = \underbrace{a^T u_c}_{} z_{c'}^{(1)} = b_c z_{c'} \qquad \frac{\partial f}{\partial W^{(1)}_{cj}} = \underbrace{b^T d_c}_{} x_{ij} = e_c x_{ij}$$

- Backpropagation algorithm:
  - Forward propagation computes ‘r’ and $z^{(m)}$ for all ‘m’.
  - Backpropagation step 1: use ‘r’ to get gradient of ‘w’.
  - Backpropagation step 2: use a_c to get gradient of $W^{(3)}$.
  - Backpropagation step 3: use b_c to get gradient of $W^{(2)}$.
  - Backpropagation step 4: use d_c to get gradient of $W^{(1)}$.

Last Time: Backpropagation with 3 Hidden Layers

- Backpropagation algorithm:
  - Forward propagation computes ‘r’ and $z^{(m)}$ for all ‘m’.
  - Backpropagation step 1: use r to get gradient of ‘w’.
  - Backpropagation step 2: use a_c to get gradient of $W^{(3)}$.
  - Backpropagation step 3: use b_c to get gradient of $W^{(2)}$.
  - Backpropagation step 4: use d_c to get gradient of $W^{(1)}$.
- Cost of backpropagation:
  - Forward pass dominated by multiplications by $W^{(1)}$, $W^{(2)}$, $W^{(3)}$, and ‘w’.
    - If have ‘m’ layers and all z_i have ‘k’ elements, cost would be $O(dk + mk^2)$.
  - Backward pass has same cost.
- For multi-class or multi-label classification, replace ‘w’ by matrix.
  - Softmax loss is called “cross entropy” in neural network papers.