Deep Learning Assignment 1

Anonymous Author(s)
Affiliation
Address
`email`

1 MLP backprop and NumPy implementation

1.1 Analytical derivation of gradients

Question 1.1 a)

1. Cross entropy module
   $\frac{\partial L}{\partial x^{(N)}} = -\sum_i t_i \frac{1}{x_i^{(N)}} = \frac{1}{x^{(N)}_{argmax(t)}}$

2. Softmax module
   $\frac{\partial x_i^{(N)}}{\partial \tilde{x}_j^{(N)}} =$
   $\frac{exp(\tilde{x}_j^{(N)})}{\sum_{i=1}^{d_N} exp(\tilde{x}_i^{(N)})} - (\frac{exp(\tilde{x}_j^{(N)})}{\sum_{i=1}^{d_N} exp(\tilde{x}_i^{(N)})})^2$ if $i = j$
   $-\frac{exp(\tilde{x}_i^{(N)})}{\sum_{i=1}^{d_N} exp(\tilde{x}_i^{(N)})} \frac{exp(\tilde{x}_j^{(N)})}{\sum_{i=1}^{d_N} exp(\tilde{x}_i^{(N)})}$ if $i \neq j$
   $= \delta_{ij}(x_i^{(N)} - x_i^{N2}) + (1 - \delta_{ij})(-x_i x_j)$

3. ReLU module
   $\frac{\partial x^{(l<N)}}{\partial \tilde{x}^{(l<N)}} =$
   1 if $\tilde{x}^{(l)} \geq 0$
   0 if $\tilde{x}^{(l)} < 0$

4. Linear module
   $\frac{\partial \tilde{x}^{(l)}}{\partial x^{(l-1)}} = W^{(l)}$
   $\frac{\partial \tilde{x}^{(l)}}{\partial W^{(l)}} = x^{(l-1)}$
   $\frac{\partial \tilde{x}^{(l)}}{\partial b^{(l)}} = 1$

Question 1.1 b)

1. $\frac{\partial L}{\partial \tilde{x}^{(N)}} = \frac{\partial L}{\partial x^{(N)}} \frac{\partial x^{(N)}}{\partial \tilde{x}^{(N)}} =$
   $\sum_i t_i \frac{1}{x_i^{(N)}} (\delta_{ij}(x_i^{(N)} - x_i^{N2}) + (1 - \delta_{ij})(-x_i x_j))$

2. $\frac{\partial L}{\partial \tilde{x}^{(l<N)}} = \frac{\partial L}{\partial x^{(N)}} \frac{\partial x^{(N)}}{\partial \tilde{x}^{(N)}} =$
   $\frac{\partial L}{\partial x^{(N)}} \frac{\partial x^{(N)}}{\partial \tilde{x}^{(N)}} \frac{\partial \tilde{x}^{(N)}}{\partial x^{(N-1)}} \frac{\partial x^{(N-1)}}{\partial \tilde{x}^{(N-1)}} \prod_{n=l}^{N-1} \frac{\partial \tilde{x}^{(n+1)}}{\partial x^{(n)}} \frac{\partial x^{(n)}}{\partial \tilde{x}^{(n)}}$

Submitted to 32nd Conference on Neural Information Processing Systems (NIPS 2018). Do not distribute.

1.2 NumPy implementation

2 PyTorch MLP

3 Custom Module: Batch Normalization

3.1 Automatic differentiation

4 PyTorch CNN