Deep Learning Assignment 3

Anonymous Author(s)
Affiliation
Address
email

1 General Questions

(a) Say if the first module is:

$$max(W_1X) \tag{1}$$

where the W input layer maybe doing summation and summation just like matrix mutiplication does WX, and the max function is a non-linear active function modifying the valuse like a neuron does before entering the next module:

$$W_2(max(W_2X)) \tag{2}$$

If now we don't have the active function then the formula will looks like:

$$W_2(W_1X) \to \bar{W}X \tag{3}$$

which eventually all W_i can become a single module $\bar{W}$

2 Softmax regression gradient calculation

Given

$$\hat{y} = \sigma(Wx + b) \textbf{ , where } x \in \mathbb{R}^d\textbf{,}W \in \mathbb{R}^{k\times d}\textbf{, } b \in \mathbb{R}^k \tag{4}$$

where d is the input dimension, k is the number of classes, σ is the softmax function:

$$\sigma(a)_i = \frac{exp(a_i)}{\sum_j exp(a_j)} \tag{5}$$

Which means a given input x will output y with probability of each class

2.1 Derive $\frac{\partial l}{\partial W_{ij}}$

If the given cross-entropy loss defined as followed:

$$l(y, \hat{y}) = -\sum_i y_i \log \hat{y}_i \tag{6}$$

As W_{ij} will affect the prediction of class i by multipling index j in x, therefore we can derive:

$$\frac{\partial l}{\partial W_{ij}} = \frac{\partial l}{\partial \hat{y}_i}\frac{\partial \hat{y}_i}{\partial W_{ij}} \tag{7}$$

where:

$$l(y, \hat{y}) = -\sum_i y_i \log \hat{y}_i = -(y_i \log \hat{y}_1 + y_2 \log \hat{y}_2 + \cdots + y_i \log \hat{y}_i + \dots) \tag{8}$$

and therefore

$$\frac{\partial l}{\partial \hat{y}_i} = \frac{-y_i}{\hat{y}_i} \tag{9}$$

Submitted to 29th Conference on Neural Information Processing Systems (NIPS 2016). Do not distribute.

And we can rewrite for only for $\hat{y}_i$:

$$\hat{y}_i = \frac{exp(a_i)}{\Sigma_j exp(a_j)} = \frac{exp(a_i)}{C + exp(a_i)}, \textbf{ where } C = \sum_{k \neq i} exp(a_k) \tag{10}$$

Since

$$\frac{\partial exp(a_i)}{\partial W_{ij}} = W_{ij} exp(a_i) \tag{11}$$

Therefore

$$\frac{\partial \hat{y}_i}{\partial W_{ij}} = W_{ij} \hat{y}_i (1 - \hat{y}_i) \tag{12}$$

Finally, we will get the result of $\frac{\partial l}{\partial W_{ij}}$:

$$\frac{\partial l}{\partial W_{ij}} = \frac{\partial l}{\partial \hat{y}_i} \frac{\partial \hat{y}_i}{\partial W_{ij}} = -X_j y_i (1 - \hat{y}_i) \tag{13}$$

2.2 What happen when $y_{c_1} = 1, \hat{y}_{c_2} = 1, c_1 \neq c_2$

This means something like $y = [1, 0, 0]^T$ and $\hat{y} = [0, 0, 1]^T$, and the predict is far different from true lable. This will cause the log part in loss (3) become negative infinity. We may not need to worry this because before one of the class predicted close to 1 and everything else close to 0, it will generate a great positive loss the the class that is miss-predicted trying to make the predict right to true label.

3 Chain rule

4 Variants of pooling

5 Convolution

(a) As it is using 3x3 kernal along x and y axis of input, which is 5 and 5 respectively. The output of this layer will be $(5 - 3 + 1) \times (5 - 3 + 1)$ which is 3x3.
(b) Assuming the kernel operation is point-point multiplication and summation, then the output of this layer is:

$$\begin{pmatrix} 109 & 92 & 72 \\ 108 & 85 & 74 \\ 110 & 74 & 79 \end{pmatrix}$$

(c) $\begin{pmatrix} 4 & 7 & 10 & 6 & 3 \\ 9 & 17 & 25 & 16 & 8 \\ 11 & 23 & 34 & 23 & 11 \\ 7 & 16 & 24 & 17 & 8 \\ 2 & 6 & 9 & 7 & 3 \end{pmatrix}$

6 Optimization

7 Top-k error

8 t-SNE

9 Proximal gradient descent