

# CS224n HW1

Xinyu Tan

June 4, 2017

## 1 Softmax

(a)

Take a look at any element  $i$ ,

$$\text{softmax}(\mathbf{x} + c)_i = \frac{e^{\mathbf{x}_i + c}}{\sum_j e^{\mathbf{x}_j + c}} = \frac{e^c \cdot e^{\mathbf{x}_i}}{e^c \cdot \sum_j e^{\mathbf{x}_j}} = \text{softmax}(\mathbf{x})_i$$

Therefore, we have  $\text{softmax}(\mathbf{x} + c) = \text{softmax}(\mathbf{x})$

(b)

Note the first case illustrate the broadcasting principle (dimension match) in numpy:

```
1 if len(x.shape) > 1:
2     #matrix
3     x = x - np.max(x, axis=1, keepdims=True)
4     x = np.exp(x) / np.sum(np.exp(x), axis=1, keepdims=True)
5 else:
6     # vector
7     x = x - x.max() # normalize
8     x = np.exp(x) / np.sum(np.exp(x))
```

## 2 Neural Network Basics

(a)

$$\frac{d\sigma(x)}{dx} = \frac{e^{-x}}{(1 - e^{-x})^2} = \sigma(x)(1 - \sigma(x))$$

(b)

First, we have

$$\hat{y}_i = \frac{e^{\theta_i}}{\sum_j e^{\theta_j}}$$

For one-hot encoding, only  $k$ -th element in  $\mathbf{y}$  is *one*, so we have

$$\begin{aligned} CE(\mathbf{y}, \hat{\mathbf{y}}) &= -\log \hat{y}_k = -\log \frac{e^{\theta_k}}{\sum_j e^{\theta_j}} \\ &= -\theta_k + \log \sum_j e^{\theta_j} \end{aligned}$$

Therefore,

$$\frac{\partial CE(\mathbf{y}, \hat{\mathbf{y}})}{\partial \theta_k} = -1 + \frac{e^{\theta_k}}{\sum_j e^{\theta_j}}$$

$$\frac{\partial CE(\mathbf{y}, \hat{\mathbf{y}})}{\partial \theta_i} = \frac{e^{\theta_i}}{\sum_j e^{\theta_j}}, \forall i \neq k$$

Put them altogether,

$$\frac{\partial CE(\mathbf{y}, \hat{\mathbf{y}})}{\partial \boldsymbol{\theta}} = -\mathbf{y} + \text{softmax}(\boldsymbol{\theta}) = \hat{\mathbf{y}} - \mathbf{y}$$

(c)

Denote:

$$\mathbf{a}_1 = \mathbf{x}\mathbf{W}_1 + \mathbf{b}_1 \rightarrow \mathbf{h} = \text{sigmoid}(\mathbf{a}_1)$$

$$\mathbf{a}_2 = \mathbf{h}\mathbf{W}_2 + \mathbf{b}_2 \rightarrow \hat{\mathbf{y}} = \text{softmax}(\mathbf{a}_2)$$

From last question, we have

$$\frac{\partial}{\partial \mathbf{a}_2} CE(\mathbf{y}, \hat{\mathbf{y}}) = \hat{\mathbf{y}} - \mathbf{y}$$