CS224N: ASSIGNMENT 1

KEXIN HUANG

1. Softmax

a. We have

$$softmax(x)_i = \frac{e^(x)_i}{\sum_j e^(x)_j}$$

Therefore,

$$softmax(x+c)_i = \frac{e_i^{x+c}}{\sum_j e_j^{x+c}} = \frac{e_i^x * e^c}{e^c * \sum_j e_j^x} = \frac{e_i^x}{\sum_j e_j^x} = softmax(x)$$

Therefore,

softmax(x)=*softmax*(x+c)

b. See code q1.ipynb

2. Neural Network Basics

a.

$$\sigma' = -\frac{1}{(1+e^{-x})^2} * e^{-x} * (-1) = \frac{1+e^{-x}-1}{1+e^{-x}} * \frac{1}{1+e^{-x}} = (\frac{1+e^{-x}}{1+e^-x} - \frac{1}{1+e^{-x}}) * \frac{1}{1+e^{-x}} = (1-\sigma) * \sigma$$

b. As y is a one-hot label vector, denote k as the index where y_k is 1 and others are 0 , and sum of y is 1

$$CE(y, \overline{y}) = -\sum_i y_i log(\overline{y}_i) = -\sum_i y_i * (loge_i^\theta - log\sum e_j^\theta) = -\sum_i y_i * \theta_i - y_i * log\sum e_j^\theta)$$

$$\frac{fCE(y,\overline{y})}{f\theta_a} = -y_a + \sum y_i * \frac{1}{\sum e_i^\theta * e_a^\theta}$$

Note that $\frac{1}{\sum e_i^\theta * e_a^\theta}$ is equal to $\overline{y}_a$ Therefore the above equation becomes

$$\frac{fCE(y,\overline{y})}{f\theta_a} = -y_a + \sum y_i * \overline{y}_a$$

As $\sum y_i = 1$ as y is a one hot vector, the above equation becomes

$$\frac{fCE(y,\overline{y})}{f\theta_a} = -y_a + \overline{y}_a$$

c.

$$\frac{dCE}{dx} = \frac{dCE}{dhw2+b2} * \frac{dhw2+b2}{dh} * \frac{dh}{dxw1+b1} * \frac{dxw1+b1}{dx}$$

$\frac{dCE}{dhw2+b2}$ equals to $\overline{y} - y$,
$\frac{dhw2+b2}{dh}$ equals to W2,
$\frac{dh}{dxw1+b1}$ equals to h*(1-h) as it is a sigmoid function,
$\frac{dxw1+b1}{dx}$ equals to W1

Therefore, $\frac{dCE}{dx} = (\overline{y} - y) * W2 * h * (1-h) * W1$

d. $W1 : D_x * H$
$b1 : H$
$W2 : D_y * H$
$b2 : D_y$

e. f. g. see code q2.ipynb

3. word2vec

a. Denote θ as $u^T v_c$ $\frac{dJ}{dv_c} = \frac{dJ}{d\theta} * \frac{d\theta}{dv_c} = u^T * (\overline{y} - y)$

b. Similarity, $\frac{dJ}{du_w} = \frac{dJ}{d\theta} * \frac{d\theta}{du_w} = v_c * (\overline{y} - y)^T$

c. Denote first term as θ, second term as γ $\frac{dJ}{dv_c} = \frac{dJ}{d\theta} * \frac{d\theta}{dv_c} + \frac{dJ}{d\gamma} * \frac{d\gamma}{dv_c} = (\sigma(u_o^T * v_c) - 1) * u_0 - \sum(\sigma(-u_k^T * v_c) - 1) * u_k$

$\frac{dJ}{du_o} = \frac{dJ}{d\theta} * \frac{d\theta}{du_o} = (\sigma(u_o^T * v_c) - 1) * v_c$

$\frac{dJ}{du_k} = \frac{dJ}{d\gamma} * \frac{d\gamma}{du_k} = -(\sigma(-u_k^T * v_c) - 1) * v_c$

This is faster than directly doing softmax because as it considers several samples of word vectors instead of all the word vectors. This largely improves the speed especially due to less backpropagation.

d. We already know $\frac{dF(w_j,v_c}{dU}$and $\frac{dF(w_j,v_c}{dv_c}$, denote as $A_j$and B_j

$\frac{dJ_{skipgram}}{dU} = \sum A_j$ $\frac{dJ_{skipgram}}{dV_c} = \sum B_j$ $\frac{dJ_{skipgram}}{dV_j} = 0$ for j not equal to c

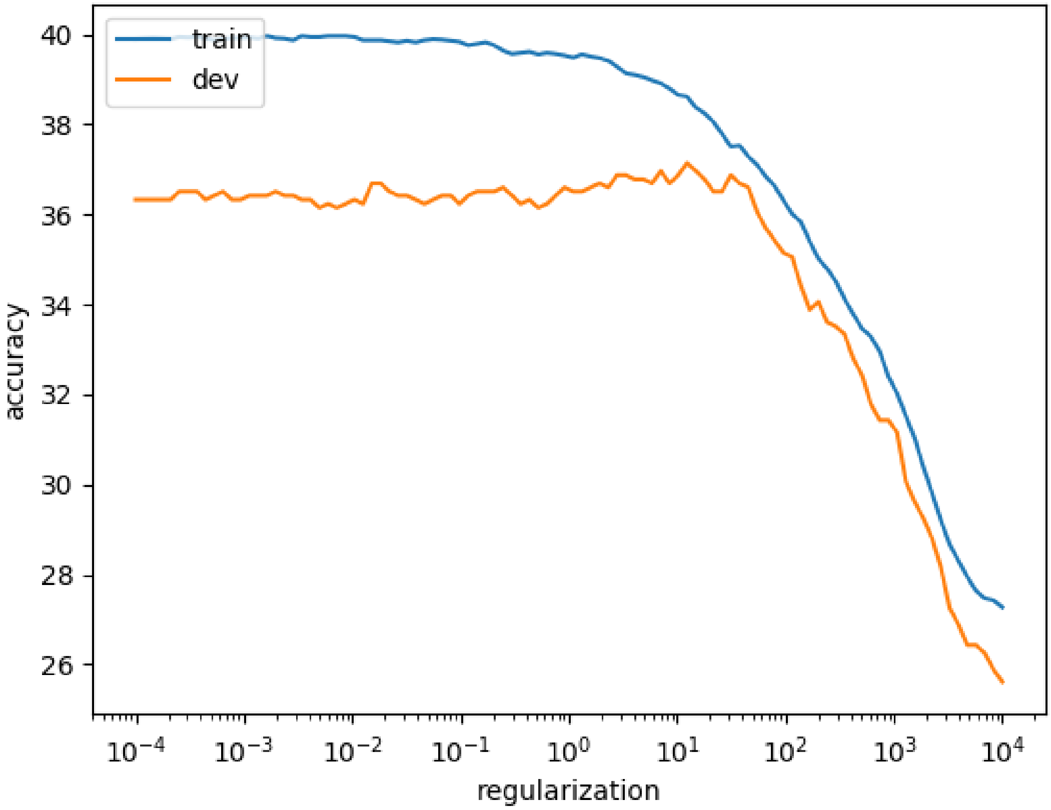
similar for CBOW

e. f. g. See code q3.ipynb

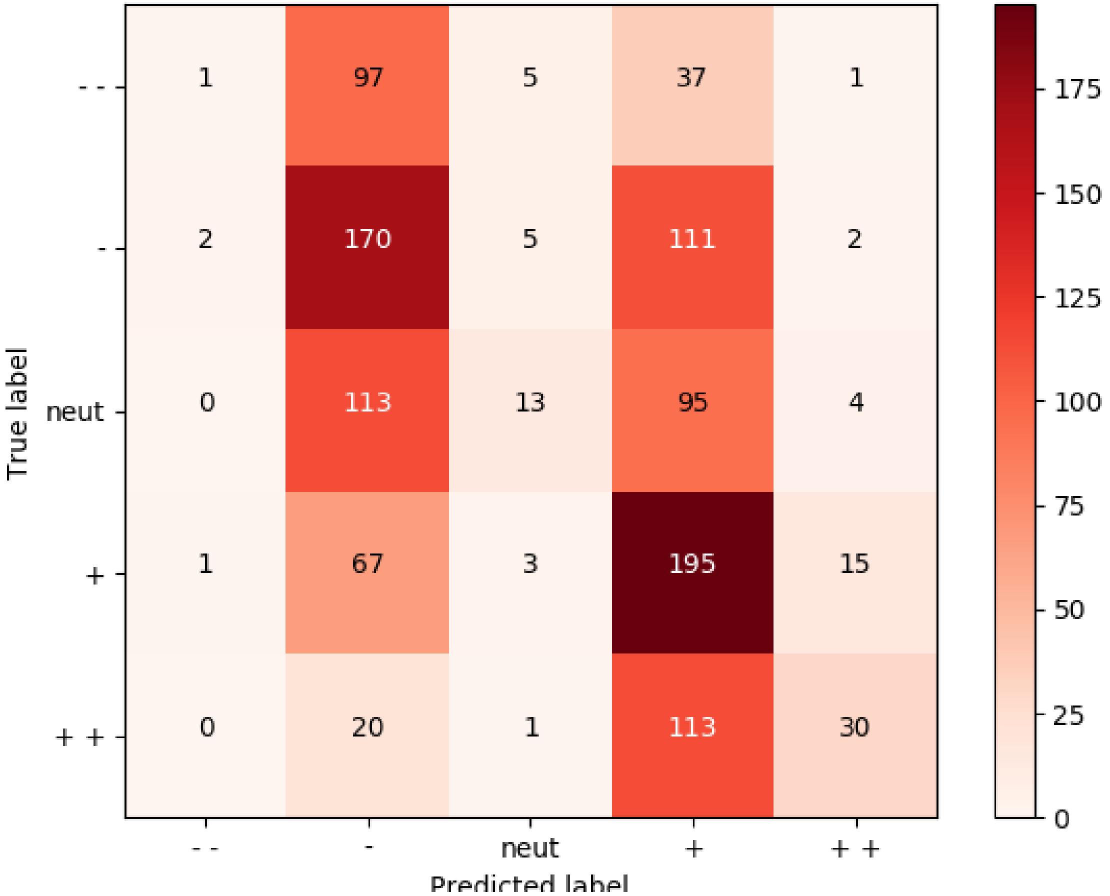
4. Sentiment Analysis

a. See code q4.ipynb
b. prevent overfitting **c.** Glove: 37.56 on test, My vectors: 29.55
The main advantage of Glove is it combines word count into cost function and it is more general as it is trained on large

datasets


train
dev
40
38
36
34
32
30
28
26
accuracy
10^{-4}
10^{-3}
10^{-2}
10^{-1}
10^{0}
10^{1}
10^{2}
10^{3}
10^{4}
regularization

d.


True label
- -
-
neut
+
+ +
1
97
5
37
1
2
170
5
111
2
0
113
13
95
4
1
67
3
195
15
0
20
1
113
30
- -
-
neut
+
+ +
Predicted label
175
150
125
100
75
50
25
0

e.