정보검색 과제 2: word2vec

B511074 박준형

1. 서술형: word2vec 이해

\# 정보검색 과제2 ; word2vec

B511074 박준형

(a) $-\sum_{w\in Vocab} y_w \log(\hat{y}_w) = -(y_1\log(\hat{y}_1)+y_2\log(\hat{y}_2)+\cdots+y_o\log(\hat{y}_o)+\cdots+y_{V-1}\log(\hat{y}_{V-1})+y_V\log(\hat{y}_V))$

(이때, $Y = [0, 0, \cdots 1, \cdots 0, 0]$ ($w = 1, 2, \cdots o, \cdots V-1, V$) 로 이므로) $= -(\cancel{0\times\log(\hat{y}_1)+\cdots}+1\times\log(\hat{y}_o)\cancel{+\cdots+0\times\log(\hat{y}_V)})$

$\therefore -(1\times\log(\hat{y}_o)) = -\log(\hat{y}_o)$ 이다.

(b) $J_{N\text{-}S} = [-\log(\hat{y}_o)]$, $\frac{\partial J_{N\text{-}S}}{\partial v_c} = \frac{\partial[-\log(\hat{y}_o)]}{\partial v_c} = \frac{\partial[-\log(\frac{e^{u_o^T v_c}}{\sum_{w\in V} e^{u_w^T v_c}})]}{\partial v_c}$

$$= -\frac{\partial[\log(e^{u_o^T v_c}) - \log(\sum_{w\in V} e^{u_w^T v_c})]}{\partial v_c} = -(u_o) + \frac{\partial}{\partial v_c}[\log(\underbrace{\sum_{w\in V} e^{u_w^T v_c}}_{f(v_c)})]$$

$$= -(u_o) + \frac{\partial}{\partial v_c}\log f(v_c) = -(u_o) + \frac{f(v_c)'}{f(v_c)} = -(u_o) + \frac{\sum_{x\in V} u_x\cdot e^{u_x^T v_c}}{\sum_{w\in V} e^{u_w^T v_c}}$$

$$= -u_o + \sum_{x\in V}\frac{e^{u_x^T v_c}}{\sum_{w\in V}e^{u_w^T v_c}}\cdot u_x = -u_o + \sum_{x\in V} p(u_x|v_c)\cdot u_x = -u_o + \sum_{x\in V}\hat{y}_x\cdot u_x$$

(c) (1) w=o일때,

$$\frac{\partial[-\log(\hat{y}_o)]}{\partial u_{w=o}} = \frac{\partial[-\log(\frac{e^{u_o^T v_c}}{\sum_{w\in V}e^{u_w^T v_c}})]}{\partial u_{w=o}} = -\frac{\partial(u_o^T v_c)}{\partial u_{w=o}} + \frac{\partial[\overbrace{\log(\sum_{w\in V} e^{u_w^T v_c})}^{\log f(u_w)}]}{\partial u_{w=o}}$$

$$= -v_c + \frac{\partial \log f(u_{w=o})}{\partial u_{w=o}} = -v_c + \frac{f(u_{w=o})'}{f(u_{w=o})} = -v_c + \underbrace{\frac{e^{u_o^T v_c}}{\sum_{w\in V}e^{u_w^T v_c}}}_{\hat{y}_o}\cdot v_c = -v_c+\hat{y}_o\cdot v_c = -v_c(1-\hat{y}_o)$$

(2) w≠o일때,

$$\frac{\partial[-\log(\hat{y}_o)]}{\partial u_{w\neq o}} = \frac{\partial[-\log(\frac{e^{u_o^T v_c}}{\sum_{w\in V} e^{u_w^T v_c}})]}{\partial u_{w\neq o}} = -\frac{\partial u_o^T v_c}{\partial (u_{w\neq o})} + \frac{\partial[\log(\sum_{w\in V} e^{u_w^T v_c})]}{\partial u_{w\neq o}}$$

$$= 0 + \left(\frac{e^{u_{w\neq o}^T v_c}}{\sum_{w\in V}e^{u_w^T v_c}}\cdot v_c\right) = v_c\cdot\hat{y}_{w\neq o}$$

(d)

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

미분.

$$= \frac{-1+e^{-x}+1}{(1+e^{-x})^2} = \left(\frac{1}{1+e^{-x}}\right)\times\left(\frac{e^{-x}+1}{1+e^{-x}} - \frac{1}{1+e^{-x}}\right) = \left(\frac{1}{1+e^{-x}}\right)\left(1-\frac{1}{1+e^{-x}}\right) = \sigma(x)(1-\sigma(x))$$

(f) (i) $\dfrac{\partial \sum_{-m \le j \le m, j \ne 0} J(v_c, w_{t+j}, U)}{\partial U} = \sum_{\substack{-m \le j \le m, \\ j \ne 0}} \partial J(v_c, w_{t+j}, U) / \partial U$

(ii) $\sum_{\substack{-m \le j \le m, \\ j \ne 0}} \partial J(v_c, w_{t+j}, U) / \partial v_c$

(iii) $w \ne c$ 일때, 편미분을 하면, $\dfrac{\partial}{\partial v_w} \left(J(v_c, w_{t-m}, \dots, w_{t+m}, U) = 0 \right.$ 이된다.

$$\frac{\partial [-\log(\hat{y}_o)]}{\partial v_{w \ne c}} = \frac{\partial \left[-\log\left(\frac{e^{u_o^T v_c}}{\sum_{w \in V} e^{u_w^T v_c}}\right)\right]}{\partial V_{w \ne c}} = -\frac{u_o^T v_c}{\partial v_{w \ne c}} + \frac{\partial \left[\log\left(\sum_{w \in V} e^{u_w^T v_c}\right)\right]}{\partial v_{w \ne c}}$$

$$= 0 + \sum_{w \in V} 0 = 0$$

2. 코드: word2vec 구현

(a)

1) sigmoid = $1/(1 + e^{-x})$

```python
def sigmoid(x):
    """
    Compute the sigmoid function for the input here.
    Arguments:
    x -- A scalar or numpy array.
    Return:
    s -- sigmoid(x)
    """

    ### YOUR CODE HERE (~1 Line)
    s = 1 / (1 + np.exp(-x))
    ### END YOUR CODE

    return s
```

2) naiveSoftmaxLossAndGradient

centerWord에 대한 하나의 outSideWord의 가중치를 계산하는 함수

```
### YOUR CODE HERE (~6-8 Lines)

uv = np.dot(outsideVectors, centerWordVec) # uv.shape: (d, c) by (c, ) -> (d, )
y_hat = softmax(uv)  # sum = 1
y_o = y_hat[outsideWordIdx]  # w=o에 해당하는 확률만
loss = np.log(y_o) # if y_o == 0.9 -> -0.1, y_o == 0.04 -> -96

# -uoT + sum(y * u)
gradCenterVec = -outsideVectors[outsideWordIdx] + np.sum(y_hat * outsideVectors.T, axis = 1)

gradOutsideVecs = centerWordVec * y_hat.reshape(-1, 1) # (c, ) * (d, ), w != o 일 경우
gradOutsideVecs[outsideWordIdx] = centerWordVec * (y_o - 1) # w = o 일 경우

### Please use the provided softmax function (imported earlier in this file)
### This numerically stable implementation helps you avoid issues pertaining
### to integer overflow.

### END YOUR CODE

return loss, gradCenterVec, gradOutsideVecs
```

3) skipgram

centerWord에 대한 outSideWord들의 가중치를 계산하는 함수

(naiveSoftmaxLossAndGradient 이용)

```
### YOUR CODE HERE (~8 Lines)
centerWordVec = centerWordVectors[word2Ind[currentCenterWord]] #Vc

for i in outsideWords:
    # word2vecLossAndGradient == naiveSoftmaxLossAndGradient
    loss1, gradCenterVec, gradOutsideVecs = word2vecLossAndGradient(centerWordVec, word2Ind[i], outsideVectors, dataset)
    loss -= loss1 # Vc와 wt-m, . . . , wt+m 에 대한 loss 값 음수 합
    gradCenterVecs[word2Ind[currentCenterWord]] -= gradCenterVec
    gradOutsideVectors -= gradOutsideVecs

### END YOUR CODE

return loss, gradCenterVecs, gradOutsideVectors
```

(b) sgd.py sgd메소드 구현

loss, grad = fx

x 가중치 학습

```
loss = None
### YOUR CODE HERE (~2 lines)
loss, grad = f(x)
x -= step*grad
### END YOUR CODE
```

(c) run.py를 수행, word vectors.png 결과

2번의 40000 수행 한 결과 같은 Loss값, 같은 png 플롯 결과가 도출됨

```
iter 39910: 28.128792
iter 39920: 28.066165
iter 39930: 28.096045
iter 39940: 28.262813
iter 39950: 28.361684
iter 39960: 28.535282
iter 39970: 28.363794
iter 39980: 28.260476
iter 39990: 28.181798
iter 40000: 28.324200
sanity check: cost at convergence should be around or below 10
training took 32415 seconds

(base) C:\Users\Jun\a1>
```

```
iter 39920: 28.066165
iter 39930: 28.096045
iter 39940: 28.262813
iter 39950: 28.361684
iter 39960: 28.535282
iter 39970: 28.363794
iter 39980: 28.260476
iter 39990: 28.181798
iter 40000: 28.324200
sanity check: cost at convergence should be around or below 10
training took 32636 seconds
```

결론: 생각보다 loss는 낮아지지 않았기 때문에 처음엔 코드가 잘못된 줄 알았지만, 여러 실험, 조사 후 이게 맞다는 것을 알게 되었다. 생각보다 (female, woman), (worth, bad) 등 눈에 띄게 관련된 단어가 근접하게 plot된 것을 보고 신기하였지만, 이정도로 정확하다고 할 순 없을 것 같다. 추가적인 학습이 요구될 것으로 보인다.