# word2vec gradient derivation

word2vec은 두 차례에 걸쳐서 나왔는데, 첫 번째는 naive softmax를 이용하여, 두 번째로는 negative sampling을 이용하여 loss function을 정의했다. 이번 리뷰에서는 각 function에서 parameter에 대한 gradient가 어떻게 유도되는지 알아보고 이를 python에서 구현해본다. 이는 cs224n의 과제이기도 하다.

먼저 어떤 파라미터들이 있는지 살펴보자. word2vec에서는 center vector와 이의 주변에 있는 oustide vectors가 있다. 각각 사용자가 지정한 $d$차원 벡터이며 center vector는 $v$, outside vector는 $u$로 표기한다. 파라미터 전체를 하나의 열 벡터로 적으면 아래와 같다.

$$\theta = \begin{bmatrix} u_1 \\ u_2 \\ \cdots \\ u_{V-1} \\ u_V \\ v_1 \\ v_2 \\ \cdots \\ v_{V-1} \\ v_V \end{bmatrix} \in \mathbb{R}^{2dV} \tag{1}$$

각 벡터 $u_1$는 $d$차원의 벡터이고 이 벡터들이 총 $2V$개가 있으므로 파라미터의 총 차원은 $2dV$이다. 이를 matrix notation으로 적으면 아래와 같다. 차원은 각각 (vocab size, dimension of embedding vector)이다.

$$\mathbf{U} = [u_1 \cdots u_V], \mathbf{V} = [v_1 \cdots v_V] \tag{2}$$

true empirical distribution $\mathbf{y}$는 $o$번째 단어의 요소는 1이고 나머지는 모두 0인 one-hot vector이다.

$$y_w = \begin{cases} 1 & w = o \\ 0 & w \neq o \end{cases}$$

또한 predicted distribution $\hat{\mathbf{y}}$는 아래의 softmax 확률에 의한 분포이다.

$$P(O = o \mid C = c) = \frac{exp\left(u_o^T v_c\right)}{\sum_{w \in Vocab} exp\left(u_w^T v_c\right)} \tag{3}$$

(3)을 이용한 naive softmax 손실 함수는 아래와 같다. $v_c$는 center vector, $o$는 outside vectors의

index, $\mathbf{U}$는 $W$개의 outside vectors을 모두 모아 둔 matrix이다.

$$\begin{aligned}\mathbf{J}_{naive-softmax}(v_c, o, \mathbf{U}) &= -log\, P(O = o \mid C = c) \\ &= -u_o^T v_c + log \sum_{w \in Vocab} exp\left(u_w^T v_c\right)\end{aligned} \tag{4}$$

(4)를 계산할 때 softmax의 분모 계산량이 너무 많아 대안으로 나온 것이 아래의 negative sampling을 이용한 손실 함수이다.

$$\mathbf{J}_{neg-sample}(v_c, o, \mathbf{U}) = -log\, \sigma(u_o^T v_c) - \sum_{k=1}^{K} log\, \sigma(-u_o^T v_c) \tag{5}$$

이제 loss function이 (4)인 경우에 대해서 gradient을 구해보자.

- gradient w.r.t $v_c$

$$\begin{aligned}\frac{\partial \mathbf{J}_{naive-softmax}}{\partial v_c} &= -u_o + \frac{\sum_{x \in Vocab} exp\left(u_x^T v_c\right) u_x}{\sum_{w \in Vocab} exp\left(u_w^T v_c\right)} \\ &= -u_o + \sum_{x \in Vocab} P(O = x \mid C = c) u_x \\ &= -\sum_{x \in Vocab} y_x u_x + \sum_{x \in Vocab} \hat{y}_x u_x \\ &= \mathbf{U}^T(\hat{\mathbf{y}} - \mathbf{y})\end{aligned}$$

- gradient w.r.t $u_w$

  만약 $w = o$, 즉 outside words라면

$$\begin{aligned}\frac{\partial \mathbf{J}_{naive-softmax}}{\partial u_o} &= -v_c + \frac{exp\left(u_o^T v_c\right) v_c}{\sum_{w \in Vocab} exp\left(u_w^T v_c\right)} \\ &= v_c \left(P(O = o \mid C = c) - 1\right)\end{aligned}$$

  만약 $w \neq o$, 즉 다른 words라면

$$\begin{aligned}\frac{\partial \mathbf{J}_{naive-softmax}}{\partial u_w} &= \frac{exp\left(u_w^T v_c\right) v_c}{\sum_{w \in Vocab} exp\left(u_w^T v_c\right)} \\ &= v_c P(O = o \mid C = c)\end{aligned}$$

$$\therefore \frac{\partial \mathbf{J}_{naive-softmax}}{\partial \mathbf{U}} = (\hat{\mathbf{y}} - \mathbf{y}) v_c^T$$

이제 loss function이 (5)인 경우 gradient을 구해보자.

- gradient w.r.t $v_c$

$$\frac{\partial \mathbf{J}_{neg-sample}}{\partial v_c} = -\left(1-\sigma\left(u_o^T v_c\right)\right) u_o - \sum_{k=1}^{K}\left(1-\sigma\left(-u_k^T v_c\right)\right)\left(-u_k\right)$$

- grdient w.r.t $u_o$

$$\frac{\partial \mathbf{J}_{neg-sample}}{\partial u_o} = -\left(1-\sigma\left(u_o^T v_c\right)\right) v_c$$

  (5)의 second term은 negative sample, 즉 $O = o$을 포함하지 않기 때문에 이에 대해 미분하면 0이 된다.

- gradient w.r.t $u_k$

$$\frac{\partial \mathbf{J}_{neg-sample}}{\partial u_k} = \left(1-\sigma\left(-u_k^T v_c\right)\right) v_c$$

이제 skig gram word2vec의 일반적인 loss function알 살펴보자. context window의 길이를 $m$으로 정했다면 이 context window에 대한 loss function은 다음과 같다.

$$\mathbf{J}_{skip-gram}(v_c, w_{t-m}, \cdots, w_{t+m}, \mathbf{U}) = \sum_{-m \le j \le m,\ j \neq 0} \mathbf{J}(v_c, w_{t+j}, \mathbf{U})$$

(4), (5)에 따라서 이는 naive softmax 또는 negative sampling이 되기도 한다.