定义如下符号

- d，词向量的维度
- n，词汇数量
- $\boldsymbol{U} \in \mathbb{R}^{d \times n}$，outside词向量矩阵，每一列是一个词向量，$\boldsymbol{u}_w \in \mathbb{R}^{d \times 1}$
- $\boldsymbol{V} \in \mathbb{R}^{d \times n}$，center词向量矩阵，每一列是一个词向量，$\boldsymbol{v}_w \in \mathbb{R}^{d \times 1}$
- $\boldsymbol{y}$，真实值，one-hot，表示是哪个词，$\boldsymbol{y} \in \mathbb{R}^{n \times 1}$
- $\hat{\boldsymbol{y}}$，预测值，表示属于某个词的概率，$\hat{\boldsymbol{y}} \in \mathbb{R}^{n \times 1}$

1-(a)

$$\boldsymbol{J}_{\text{naive-softmax}}(\boldsymbol{v}_c, o, \boldsymbol{U}) = -\log P(O = o | C = c). \tag{2}$$

(a) (3 points) Show that the naive-softmax loss given in Equation (2) is the same as the cross-entropy loss between $\boldsymbol{y}$ and $\hat{\boldsymbol{y}}$; i.e., show that

[1]Assume that every word in our vocabulary is matched to an integer number k. $\boldsymbol{u}_k$ is both the k^{th} column of $\boldsymbol{U}$ and the 'outside' word vector for the word indexed by k. $\boldsymbol{v}_k$ is both the k^{th} column of $\boldsymbol{V}$ and the 'center' word vector for the word indexed by k. **In order to simplify notation we shall interchangeably use k to refer to the word and the index-of-the-word.**

[2]The Cross Entropy Loss between the true (discrete) probability distribution p and another distribution q is $-\sum_i p_i \log(q_i)$.

$$-\sum_{w \in Vocab} y_w \log(\hat{y}_w) = -\log(\hat{y}_o). \tag{3}$$

Your answer should be one line.

A:因为one-hot向量只有标记的那一维是1，其他维度都是0，所以展开左边的$\sum$自然就是右边的代数式。

1-(b)

(b) (5 points) Compute the partial derivative of $\boldsymbol{J}_{\text{naive-softmax}}(\boldsymbol{v}_c, o, \boldsymbol{U})$ with respect to $\boldsymbol{v}_c$. Please write your answer in terms of $\boldsymbol{y}$, $\hat{\boldsymbol{y}}$, and $\boldsymbol{U}$.

直接对$\boldsymbol{J}_{naive-softmax}(\boldsymbol{v}_c, \boldsymbol{o}, \boldsymbol{U}) = -\log P(O=o|C=c)$求导，

$$
\begin{aligned}
&\frac{\partial}{\partial \boldsymbol{v}_c}\left[-\left(\boldsymbol{u}_o^T \boldsymbol{v}_c - \log \sum_{w \in Vocab} \exp\left(\boldsymbol{u}_w^T \boldsymbol{v}_c\right)\right)\right] \\
&= \frac{\partial}{\partial \boldsymbol{v}_c}\left(\log \sum_{w \in Vocab} \exp\left(\boldsymbol{u}_w^T \boldsymbol{v}_c\right)\right) - \boldsymbol{u}_o \\
&= \frac{\sum\limits_{x \in Vocab} \exp\left(\boldsymbol{u}_x^T \boldsymbol{v}_c\right) \boldsymbol{u}_x}{\sum\limits_{w \in Vocab} \exp\left(\boldsymbol{u}_w^T \boldsymbol{v}_c\right)} - \boldsymbol{u}_o \\
&= \sum_{x \in Vocab} \frac{\exp\left(\boldsymbol{u}_x^T \boldsymbol{v}_c\right) \boldsymbol{u}_x}{\exp\left(\boldsymbol{u}_w^T \boldsymbol{v}_c\right)} - \boldsymbol{u}_o \\
&= \sum_{x \in Vocab} P(O=x|C=c) \boldsymbol{u}_x - \boldsymbol{u}_o
\end{aligned}
$$

该题要求使用$\boldsymbol{y}, \hat{\boldsymbol{y}}, \boldsymbol{U}$表示出结果，我们整理一下推导结果即可。推导的结果表示损失函数对于$\boldsymbol{v}_c$的偏导数是“outside词向量的期望减去当前outside词向量”，根据这个意思即可以整理答案为

$$
\frac{\partial \boldsymbol{J}}{\partial \boldsymbol{v}_c} = U(\hat{\boldsymbol{y}} - \boldsymbol{y})
$$

1-(c)

(c) (5 points) Compute the partial derivatives of $\boldsymbol{J}_{\text{naive-softmax}}(\boldsymbol{v}_c, o, \boldsymbol{U})$ with respect to each of the 'outside' word vectors, $\boldsymbol{u}_w$'s. There will be two cases: when $w = o$, the true 'outside' word vector, and $w \neq o$, for all other words. Please write you answer in terms of $\boldsymbol{y}$, $\hat{\boldsymbol{y}}$, and $\boldsymbol{v}_c$.

该题求的是$\boldsymbol{J}$对$\boldsymbol{U}$的偏导数，可以先求$\dfrac{\partial \boldsymbol{J}}{\partial \boldsymbol{u}_w}$，这个推导和1-(b)的推导过程非常的相似，我们此处直接写出答案，就不写详细的推导了。

$$
\frac{\partial \boldsymbol{J}}{\partial \boldsymbol{u}_w} = \begin{cases} (\hat{y}_w - 1)\boldsymbol{v}_c, & w = o \\ \hat{y}_w \boldsymbol{v}_c, & w \neq o \end{cases}
$$

根据标量$\boldsymbol{f}$对矩阵$\boldsymbol{X}$的求导规律：$\dfrac{\partial f}{\partial X} = \left[\dfrac{\partial f}{\partial X_{ij}}\right]$，可以得到答案

$$
\frac{\partial \boldsymbol{J}}{\partial \boldsymbol{U}} = \boldsymbol{v}_c(\hat{\boldsymbol{y}} - \boldsymbol{y})^T
$$

小技巧：如果对于向量或者矩阵的求导犯迷糊了，可以多从维度的角度去check。

1-(d)

(d) (3 Points) The sigmoid function is given by Equation 4:

$$\sigma(\boldsymbol{x}) = \frac{1}{1+e^{-\boldsymbol{x}}} = \frac{e^{\boldsymbol{x}}}{e^{\boldsymbol{x}}+1} \tag{4}$$

Please compute the derivative of $\boldsymbol{\sigma}(x)$ with respect to $\boldsymbol{x}$, where $\boldsymbol{x}$ is a vector.

标量$\boldsymbol{f}$对向量$\boldsymbol{x}$求导，结果是$\frac{\partial f}{\partial \boldsymbol{x}} = \left[\frac{\partial f}{\partial x_1}, \dots, \frac{\partial f}{\partial x_m}\right]^T$，又$\sigma'(x) = \sigma(x)(1-\sigma(x))$，所以有

$$\sigma'(\boldsymbol{x}) = \begin{bmatrix} \sigma(x_1)(1-\sigma(x_1)) \\ \sigma(x_2)(1-\sigma(x_2)) \\ \dots \\ \sigma(x_m)(1-\sigma(x_m)) \end{bmatrix}$$

1-(e)

(e) (4 points) Now we shall consider the Negative Sampling loss, which is an alternative to the Naive Softmax loss. Assume that K negative samples (words) are drawn from the vocabulary. For simplicity of notation we shall refer to them as $w_1, w_2, \dots, w_K$ and their outside vectors as $\boldsymbol{u}_1, \dots, \boldsymbol{u}_K$. Note that $o \notin \{w_1, \dots, w_K\}$. For a center word c and an outside word o, the negative sampling loss function is given by:

$$\boldsymbol{J}_{\text{neg-sample}}(\boldsymbol{v}_c, o, \boldsymbol{U}) = -\log(\sigma(\boldsymbol{u}_o^\top \boldsymbol{v}_c)) - \sum_{k=1}^{K} \log(\sigma(-\boldsymbol{u}_k^\top \boldsymbol{v}_c)) \tag{5}$$

for a sample $w_1, \dots w_K$, where $\sigma(\cdot)$ is the sigmoid function.[3]

Please repeat parts (b) and (c), computing the partial derivatives of $\boldsymbol{J}_{\text{neg-sample}}$ with respect to $\boldsymbol{v}_c$, with respect to $\boldsymbol{u}_o$, and with respect to a negative sample $\boldsymbol{u}_k$. Please write your answers in terms of the vectors $\boldsymbol{u}_o$, $\boldsymbol{v}_c$, and $\boldsymbol{u}_k$, where $k \in [1, K]$. After you've done this, describe with one sentence why this loss function is much more efficient to compute than the naive-softmax loss. Note, you should be able to use your solution to part (d) to help compute the necessary gradients here.

$$\begin{aligned} \frac{\partial J_{\text{neg-sample}}}{\partial \boldsymbol{v}_c} &= -\frac{\sigma(\boldsymbol{u}_o^T \boldsymbol{v}_c)(1-\sigma(\boldsymbol{u}_o^T \boldsymbol{v}_c))\boldsymbol{u}_o}{\sigma(\boldsymbol{u}_o^T \boldsymbol{v}_c)} - \sum_{k=1}^{K} \frac{\sigma(-\boldsymbol{u}_k^T \boldsymbol{v}_c)(1-\sigma(-\boldsymbol{u}_k^T \boldsymbol{v}_c))(-1)\boldsymbol{u}_k}{\sigma(-\boldsymbol{u}_k^T \boldsymbol{v}_c)} \\ &= (\sigma(\boldsymbol{u}_o^T \boldsymbol{v}_c) - 1)\boldsymbol{u}_o + \sum_{k=1}^{K} (1-\sigma(-\boldsymbol{u}_k^T \boldsymbol{v}_c))\boldsymbol{u}_k \end{aligned}$$

类似的有

$$\frac{\partial J_{\text{neg-sample}}}{\partial \boldsymbol{u}_o} = (\sigma(\boldsymbol{u}_o^T \boldsymbol{v}_c) - 1)\boldsymbol{v}_c$$

类似的有

$$\frac{\partial J_{\text{neg-sample}}}{\partial \boldsymbol{u}_k} = (1-\sigma(-\boldsymbol{u}_k^T \boldsymbol{v}_c))\boldsymbol{v}_c$$

$J_{\text{neg-sample}}$比$J_{\text{naive-softmax}}$训练效率高的主要原因是前者只需要和相关的词向量进行内积，而后者的偏导数里面有$\hat{\boldsymbol{y}}$这一项，该项使用Softmax公式得出，需要计算$\boldsymbol{v}_c$和所有的词向量内积，这个步骤会消耗较多的计算资源。

1-(f)

(f) (3 points) Suppose the center word is $c = w_t$ and the context window is $[w_{t-m}, \ldots, w_{t-1}, w_t, w_{t+1}, \ldots, w_{t+m}]$, where m is the context window size. Recall that for the skip-gram version of `word2vec`, the total loss for the context window is:

$$\boldsymbol{J}_{\text{skip-gram}}(\boldsymbol{v}_c, w_{t-m}, \ldots w_{t+m}, \boldsymbol{U}) = \sum_{\substack{-m \le j \le m \\ j \ne 0}} \boldsymbol{J}(\boldsymbol{v}_c, w_{t+j}, \boldsymbol{U}) \tag{6}$$

Here, $\boldsymbol{J}(\boldsymbol{v}_c, w_{t+j}, \boldsymbol{U})$ represents an arbitrary loss term for the center word $c = w_t$ and outside word w_{t+j}. $\boldsymbol{J}(\boldsymbol{v}_c, w_{t+j}, \boldsymbol{U})$ could be $\boldsymbol{J}_{\text{naive-softmax}}(\boldsymbol{v}_c, w_{t+j}, \boldsymbol{U})$ or $\boldsymbol{J}_{\text{neg-sample}}(\boldsymbol{v}_c, w_{t+j}, \boldsymbol{U})$, depending on your implementation.

Write down three partial derivatives:

(i) $\partial \boldsymbol{J}_{\text{skip-gram}}(\boldsymbol{v}_c, w_{t-m}, \ldots w_{t+m}, \boldsymbol{U}) / \partial \boldsymbol{U}$

(ii) $\partial \boldsymbol{J}_{\text{skip-gram}}(\boldsymbol{v}_c, w_{t-m}, \ldots w_{t+m}, \boldsymbol{U}) / \partial \boldsymbol{v}_c$

[3] Note: the loss function here is the negative of what Mikolov et al. had in their original paper, because we are doing a minimization instead of maximization in our assignment code. Ultimately, this is the same objective function.

(iii) $\partial \boldsymbol{J}_{\text{skip-gram}}(\boldsymbol{v}_c, w_{t-m}, \ldots w_{t+m}, \boldsymbol{U}) / \partial \boldsymbol{v}_w$ when $w \ne c$

Write your answers in terms of $\partial \boldsymbol{J}(\boldsymbol{v}_c, w_{t+j}, \boldsymbol{U}) / \partial \boldsymbol{U}$ and $\partial \boldsymbol{J}(\boldsymbol{v}_c, w_{t+j}, \boldsymbol{U}) / \partial \boldsymbol{v}_c$. This is very simple – each solution should be one line.

损失函数就是窗口滑动的过程中，每个窗口的损失累加，所以偏导数也容易得出（“多项和的导数”等于“多项导数的和”）。

$$\frac{\partial J_{\text{skip-gram}}}{\partial \boldsymbol{U}} = \sum_{-m \le j \le m, j \ne 0} \frac{\partial J(\boldsymbol{v}_c, w_{t+j}, \boldsymbol{U})}{\partial \boldsymbol{U}}$$

$$\frac{\partial J_{\text{skip-gram}}}{\partial \boldsymbol{v}_c} = \sum_{-m \le j \le m, j \ne 0} \frac{\partial J(\boldsymbol{v}_c, w_{t+j}, \boldsymbol{U})}{\partial \boldsymbol{v}_c}$$

$$\frac{\partial J_{\text{skip-gram}}}{\partial \boldsymbol{v}_w} = 0, \text{when } w \ne c$$