Stanford CS 224n Assignment 2

Shivanshu Shekhar

December 2021

1 Written: Understanding word2vec (23 points)

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

$$-\sum_{w \in Vocab} y_w \log(\hat{(}y_w)) = -\log(\hat{y_o})$$

Answer: Since y is a one hot vector so $y_w = 1$ if $w = 0$ else $y_w = 0$ so the summation just reduces to $-\log(\hat{y_0})$.

(b) (5 points) Compute the partial derivative of $J_{naive-softmax}(\mathbf{v_c}, o, \mathbf{U})$ with respect to $\mathbf{v_c}$. Please write your answer in terms of $\mathbf{y}$, $\hat{\mathbf{y}}$, and $\mathbf{U}$. **Answer**: We know that y is an one hot vector which is 1 if w = o otherwise is 0.

$$\frac{\partial \mathbf{J}}{\partial \mathbf{v_c}} = -\mathbf{u_o} + \frac{\sum_{w \in Vocab} \exp \mathbf{u_w^\intercal v_c u_w}}{\sum_{w \in Vocab} \exp \mathbf{u_w^\intercal v_c}}$$
$$\hat{\mathbf{y}} = softmax(\mathbf{U^\intercal v_c})$$
$$\mathbf{J} = -\mathbf{Uy} + \mathbf{U}\hat{\mathbf{y}}$$
$$\mathbf{J} = \mathbf{U}(\hat{\mathbf{y}} - \mathbf{y})$$

(c) (5 points) Compute the partial derivatives of $J_{naive-softmax}(\mathbf{v_c}, o, \mathbf{U})$ with respect to each of the 'outside' word vectors, $\mathbf{u_w}$'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 $\mathbf{y}$, $\hat{\mathbf{y}}$, and $\mathbf{v_c}$.

Answer:

if w = o then:

$$\frac{\partial \mathbf{J}}{\partial \mathbf{u_w}} = -v_c + \frac{\exp \mathbf{u_o^\intercal v_c}}{\sum_{w \in Vocab} \exp \mathbf{u_w^\intercal v_c}}$$

if $w \neq o$ then:

$$\frac{\partial \mathbf{J}}{\partial \mathbf{u_w}} = \frac{\exp \mathbf{u_w^\intercal v_c v_c}}{\sum_{w \in Vocab} \exp \mathbf{u_w^\intercal v_c}}$$

Finally in vector form we get

$$\begin{aligned}\frac{\partial \mathbf{J}}{\partial \mathbf{u_w}} &= -\mathbf{v_c}\mathbf{y}^\intercal + \mathbf{v_c}\hat{\mathbf{y}}^\intercal \\ &= \mathbf{v_c^\intercal}(\hat{\mathbf{y}} - \mathbf{y})\end{aligned}$$

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

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

Please compute the derivative of $\sigma(x)$ with respect to x, where x is a scalar. Hint: you may want to write your answer in terms of $\sigma(x)$.

Answer:

$$\begin{aligned}\frac{\mathrm{d}\sigma(x)}{\mathrm{d}x} &= \frac{e^{-x}}{(1+e^{-x})^2} \\ &= \sigma(x)(1-\sigma(x))\end{aligned}$$

(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 $w_1,\ w_2,\ .\ .\ .\ ,\ w_K$ and their outside vectors as $\mathbf{u_1}, ..., \mathbf{u_K}$. Note that $o \notin w_1, ..., w_K$. For a center word c and an outside word o, the negative sampling loss function is given by:

$$J_{neg-samples}(\mathbf{v_c}, o, \mathbf{U}) = -\log(\sigma(\mathbf{u_o^\intercal v_c})) - \sum_{k=1}^{\mathrm{K}} \log(\sigma(-\mathbf{u_k^\intercal v_c}))$$

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

Please repeat parts (b) and (c), computing the partial derivatives of $\mathbf{J}_{neg-sample}$ with respect to $\mathbf{v_c}$, with respect to u_o, and with respect to a negative sample u_k. Please write your answers in terms of the vectors $u_o, v_c, and u_k$, where $k \in [1, \mathrm{K}]$. 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. **Answer**:

$$\begin{aligned}\frac{\partial \mathbf{J}}{\partial \mathbf{u_0}} &= (\sigma(\mathbf{u_0^\intercal v_c}) - 1)\mathbf{v_c} \\ \frac{\partial \mathbf{J}}{\partial \mathbf{v_c}} &= (\sigma(\mathbf{u_0^\intercal v_c}) - 1)\mathbf{u_0} - \sum_{k=1}^{\mathrm{K}}(\sigma(-\mathbf{u_k^\intercal v_c}) - 1)\mathbf{u_k} \\ \frac{\partial \mathbf{J}}{\partial \mathbf{u_k}} &= (\sigma(\mathbf{u_k^\intercal v_c}) - 1)\mathbf{v_c}\end{aligned}$$

For softmax we have to sum over all the outside vectors U which is computationally heavy task, whereas negative sampling only calculates sum over only K samples so it is much more efficient.

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

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

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

Write down three partial derivatives:

(i) $\frac{\partial \mathbf{J}_{skip-gram}(\mathbf{v_c}, w_{t-m}, ..., w_{t+m}, \mathbf{U})}{\partial \mathbf{U}}$

(ii) $\frac{\partial \mathbf{J}_{skip-gram}(\mathbf{v_c}, w_{t-m}, ..., w_{t+m}, \mathbf{U})}{\partial \mathbf{v_c}}$

(iii) $\frac{\partial \mathbf{J}_{skip-gram}(\mathbf{v_c}, w_{t-m}, ..., w_{t+m}, \mathbf{U})}{\partial \mathbf{v_w}}$ when $w \ne c$

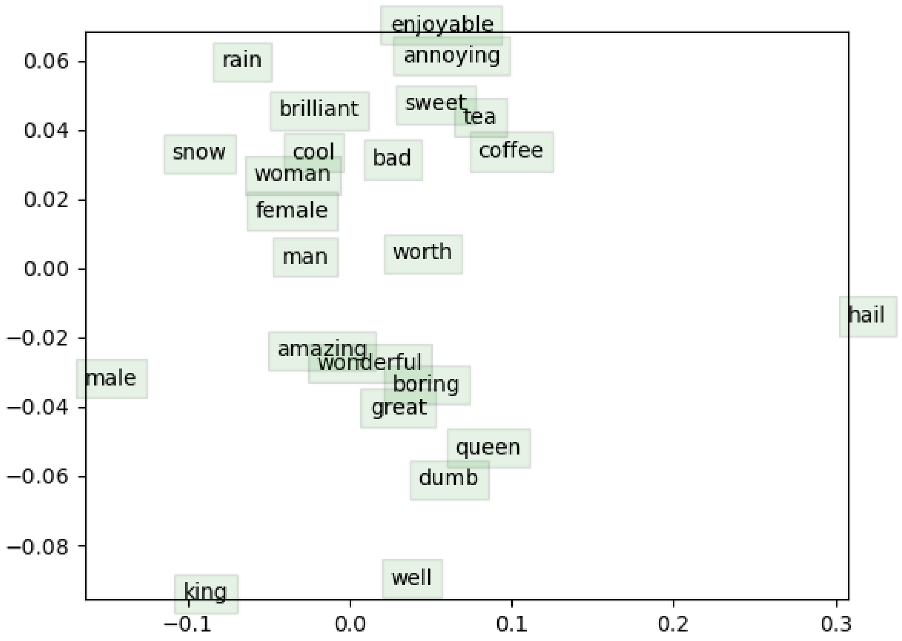
Write your answers in terms of $\frac{\partial \mathbf{J}(\mathbf{v_c}, w_{t-m}, ..., w_{t+m}, \mathbf{U})}{\partial \mathbf{U}}$ and $\frac{\partial \mathbf{J}(\mathbf{v_c}, w_{t-m}, ..., w_{t+m}, \mathbf{U})}{\partial \mathbf{v_c}}$. This is very simple – each solution should be one line. **Answer**:

$$\frac{\partial \mathbf{J}_{skip-gram}(\mathbf{v_c}, w_{t-m}, ..., w_{t+m}, \mathbf{U})}{\partial \mathbf{U}} = \sum_{\substack{-m \le jm \le \\ j \ne 0}} \frac{\partial \mathbf{J}(\mathbf{v_c}, w_{t+j}, \mathbf{U})}{\partial \mathbf{U}}$$

$$\frac{\partial \mathbf{J}_{skip-gram}(\mathbf{v_c}, w_{t-m}, ..., w_{t+m}, \mathbf{U})}{\partial \mathbf{v_c}} = \sum_{\substack{-m \le jm \le \\ j \ne 0}} \frac{\partial \mathbf{J}(\mathbf{v_c}, w_{t+j}, \mathbf{U})}{\partial \mathbf{v_c}}$$

$$\frac{\partial \mathbf{J}_{skip-gram}(\mathbf{v_c}, w_{t-m}, ..., w_{t+m}, \mathbf{U})}{\partial \mathbf{v_w}}(w \ne c) = 0$$

2(c):



We see that alike words form a sort of cluster like amazing, wonderful, great are together. The model also puts some opposite words together such as boring with wonderful, enjoyable with annoying. The model also separates related words such as man and king but as it is only a word2vec model and is trained on a relatively small dataset so error like these were very likely.