Stanford CS 224n Assignment 2

Akash Gupta

August 25, 2020

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 $\hat{y}$; i.e., show that

$$-\sum_{w \in Vocab} y_w \log \hat{y}_w = -\log \hat{y}_o$$

Your answer should be one line.

Answer: Since y is a one-hot vector where $y_w = 1$ when $w = o$ and $y_w = 0$ when $w \neq o$,

$$\begin{aligned}
-\sum_{w \in Vocab} y_w \log(\hat{y}_w) &= -[y_1 \log(\hat{y}_1) + \cdots + y_o \log(\hat{y}_o) + \cdots + y_w \log(\hat{y}_w)] \\
&= -y_o \log(\hat{y}_o) \\
&= -\log(\hat{y}_o)
\end{aligned}$$

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

Answer: Let input vector be $\theta = U^\intercal v_c$ and prediction function be $\hat{y} = softmax(\theta)$. From (a), we know the derivative of cross-entropy loss is equivalent to softmax loss for one hot vector $\mathbf{y}$, therefore,

$$\begin{aligned}
J &= CrossEntropy(y, \hat{y}) \\
\therefore \frac{\partial J}{\partial \theta} &= (\hat{y} - y)^\intercal
\end{aligned}$$

From above, we can use chain rule to solve the derivative:

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

(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, 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 y, $\hat{y}$, and v_c.

Answer: Similar to answer (b) above.

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

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

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

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: Apply quotient rule $f(x) = \frac{g(x)}{h(x)}$, $f\prime(x) = \frac{g\prime(x)h(x)+g(x)h\prime(x)}{h^2(x)}$, we have:

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

(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 $w_1, w_2, \cdots, w_K$ and their outside vectors as $\mathbf{u_1}, \cdots, \mathbf{u_K}$. 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:

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

for a sample $w_1, \cdots, w_K$, where $\sigma(.)$ is the sigmoid function.

Please repeat parts (b) and (c), computing the partial derivatives of $J_{neg-sample}$ with respect to 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 $k \in [1, K]$. 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:

$$\begin{aligned}\frac{\partial J}{\partial u_o} &= (\sigma(u_o^\intercal v_c) - 1)v_c \\ \frac{\partial J}{\partial u_k} &= (\sigma(u_k^\intercal v_c) - 1)v_c, \forall k \in [1, K] \\ \frac{\partial J}{\partial v_c} &= (\sigma(u_o^\intercal v_c) - 1)u_o - \sum_{k=1}^{K}(\sigma(-u_k^\intercal v_c) - 1)u_k\end{aligned}$$

Answer: For naive softmax loss, it computes the whole outside vectors U. But for negative sampling loss, it only calculates a fixed size K. Thus, negative sampling loss is more compute and memory efficient

(f) (3 points) Suppose the center word w_t and the context window is $[w_{t-m}, \cdots, w_{t-1}, w_t, w_{t+1}, \cdots, w_{t+m}]$, where m is the context window size. Recall that for the skip-gram version of word2Vec, the total loss for the context window is:

$$J_{skip-gram}(v_c, w_{t-m}, \cdots, w_{t+m}, U) = \sum_{\substack{-m \le j \le m \\ j \ne 0}} J(v_c, w_{t+j}, U) \tag{2}$$

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

Write down three partial derivatives:

(i) $\partial J_{skip-gram}(v_c, w_{t-m}, \cdots, w_{t+m})/\partial U$
(ii) $\partial J_{skip-gram}(v_c, w_{t-m}, \cdots, w_{t+m})/\partial v_c$
(iii) $\partial J_{skip-gram}(v_c, w_{t-m}, \cdots, w_{t+m})/\partial w_c$ when $w \ne c$

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

Answer:

$$\partial J_{skip-gram}(v_c, w_{t-m}, \cdots, w_{t+m})/\partial U = \sum_{\substack{-m \le j \le m \\ j \ne 0}} \frac{J(v_c, w_{t+j}, U)}{\partial U}$$

$$\partial J_{skip-gram}(v_c, w_{t-m}, \cdots, w_{t+m})/\partial v_c = \sum_{\substack{-m \le j \le m \\ j \ne 0}} \frac{J(v_c, w_{t+j}, U)}{\partial v_c}$$

$$\partial J_{skip-gram}(v_c, w_{t-m}, \cdots, w_{t+m})/\partial w_c (w \ne c) = 0$$