Assignment 1 Notes

mn

Contents

1 Definitions

- row vectors:
- 1 layer network:
  - Input dimension: $\mathrm{D_x}$ ($\boldsymbol{x} \in [N, D_x]$)
  - Hidden units: H ($\mathbf{W}_1 \in [D_x, H]$, $\boldsymbol{b}_1 \in [H]$)
  - Ouput dimension: $\mathrm{D_y}$ ($\mathbf{W}_2 \in [H, D_y]$, $b_2 \in [D_y]$)

2 Assigment 1

Org latex export is not that great yet, and one still has to work on the alignment.

2.1 1. Softmax

2.1.1 1.a) Softmax function

Softmax function: $\mathrm{softmax}(x)_i = \frac{e^{x_i}}{\sum_j e^{x_j}}$

$$\mathrm{softmax}(x+c)_i = \frac{e^{x_i+c}}{\sum_j e^{x_j+c}} = \frac{e^c e^{x_i}}{e^c \sum_j e^{x_j}} = \frac{e^{x_i}}{\sum_j e^{x_j}} = \mathrm{softmax}(x)_i$$

2.2 2. Neural Network Basics

2.2.1 2.a) Derivative of the sigmoid function

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

$$\frac{\partial}{\partial x}\sigma(x) = \frac{e^{-x}}{(1+e^{-x})^2} = \frac{1}{1+e^{-x}}\,\frac{e^{-x}}{1+e^{-x}} = \frac{1}{1+e^{-x}}\,\frac{(1+e^{-x})-1}{1+e^{-x}} = \sigma(x)(1-\sigma(x))$$

2.2.2 2.b) Derivative of the Cross Entropy Loss

Cross Entropy Loss: $CE(\boldsymbol{y}, \hat{\boldsymbol{y}}) = -\sum_i y_i \log(\hat{y}_i)$

$$\hat{\boldsymbol{y}} = \text{softmax}(\theta)$$

$$\frac{\partial}{\partial \theta_j} CE(\boldsymbol{y}, \hat{\boldsymbol{y}})_j = -\frac{\partial}{\partial \theta_j} \sum_i y_i \log \hat{y}_i = -\frac{\partial}{\partial \theta_j} \sum_i y_i \log(e^{\theta_i} / \sum_l e^{\theta_l}) = -\frac{\partial}{\partial \theta_j} \sum_i y_i (\theta_i - \log \sum_l e^{\theta_l})$$

$\boldsymbol{y}$ is a *one-hot* label vector with a non-zero value (=1) at index k.
Given that

$$\frac{\partial}{\partial \theta_j} \log \sum_l e^{\theta_l} = \frac{e^{\theta}_j}{\sum_l e^{\theta_l}} = \hat{y}_j$$

if $j \neq k$:

$$\frac{\partial}{\partial \theta_j} CE(\boldsymbol{y}, \hat{\boldsymbol{y}})_j = \frac{\partial}{\partial \theta_j} \log \sum_l e^{\theta_l} = \hat{y}_j$$

if $j = k$:

$$\frac{\partial}{\partial \theta_j} CE(\boldsymbol{y}, \hat{\boldsymbol{y}})_j = \hat{y}_j - 1$$

Combining (vectorizing) these results:

$$\frac{\partial}{\partial \boldsymbol{\theta}} CE(\boldsymbol{y}, \hat{\boldsymbol{y}}) = \hat{\boldsymbol{y}} - \boldsymbol{y}$$

2.2.3 2.c) Gradient of the one-layer network

$$\frac{\partial J(x)}{\partial x} = \frac{\partial}{\partial x} CE(\boldsymbol{y}, \hat{\boldsymbol{y}}) = -\frac{\partial}{\partial x} \sum_i y_i \log(\text{softmax}(\sigma(\boldsymbol{x}\mathbf{W}_1 + \boldsymbol{b}_1)\mathbf{W}_2 + \boldsymbol{b}_2))$$

With $\hat{\boldsymbol{y}} = \text{softmax}(\theta)$, $\theta = \sigma(\boldsymbol{h})\mathbf{W}_2 + \boldsymbol{b}_2$ and $\boldsymbol{h} = \boldsymbol{x}\mathbf{W}_1 + \boldsymbol{b}_1$

$$\frac{\partial J(x)}{\partial x} = \sum_j \frac{\partial J}{\partial \theta_j} \frac{\partial \theta_j}{\partial x} = \sum_j \sigma(\boldsymbol{h})(1 - \sigma(\boldsymbol{h}))\mathbf{W}_1 \mathbf{W}_{2,j} \times \begin{cases} \hat{y}_j & \text{if } j \neq k \\ \hat{y}_j - 1 & \text{if } j = k \end{cases}$$

1. Alternative (more layer-systematic) derivation
   - Forward pass:
     $$J(x) = CE(\boldsymbol{y}, \hat{\boldsymbol{y}}) = -\sum_i y_i \log(\text{softmax}(\sigma(\boldsymbol{x}\mathbf{W}_1 + \boldsymbol{b}_1)\mathbf{W}_2 + \boldsymbol{b}_2))$$
     - Input: $\boldsymbol{x}$
     - L1 affine transformation: $\boldsymbol{z}_1 = \boldsymbol{x}\mathbf{W}_1 + \boldsymbol{b}_1$
     - L1 activation: $\boldsymbol{h} = \sigma(\boldsymbol{z}_1)$
     - L2 affine transformation: $\boldsymbol{z}_2 = \boldsymbol{h}\mathbf{W}_2 + \boldsymbol{b}_2$
     - Scores computation: $\hat{\boldsymbol{y}} = \text{softmax}(\boldsymbol{z}_2)$
     - Cross-Entropy Loss: $CE = \sum_i \boldsymbol{y} \log \hat{\boldsymbol{y}}$
   - Backward pass (gradient derivation):
     $$\frac{\partial J}{\partial x} = \frac{\partial J}{\partial \hat{\boldsymbol{y}}} \frac{\partial \hat{\boldsymbol{y}}}{\partial \boldsymbol{z}_2} \frac{\partial \boldsymbol{z}_2}{\partial \boldsymbol{h}} \frac{\partial \boldsymbol{h}}{\partial \boldsymbol{z}_1} \frac{\partial \boldsymbol{z}_1}{\partial x} = \frac{\partial J}{\partial x} = (\hat{\boldsymbol{y}} - \boldsymbol{y})\mathbf{W}_2^T \sigma(\boldsymbol{z}_1)(1 - \sigma(z_1))\mathbf{W}_1^T$$
     - $dz_2 = \frac{\partial J}{\partial \boldsymbol{z}_2} = \hat{\boldsymbol{y}} - \boldsymbol{y}$
     - $dh_1 = \frac{\partial J}{\partial \boldsymbol{h}} = dz_2 \frac{\partial \boldsymbol{z}_2}{\partial \boldsymbol{h}} = dz_2 \mathbf{W}_2^T$
     - $dz_1 = \frac{\partial J}{\partial \boldsymbol{z}_1} = dh_1 \frac{\partial \boldsymbol{h}}{\partial \boldsymbol{z}_1} = dh_1 \circ \frac{\partial \sigma(\boldsymbol{z}_1)}{\partial \boldsymbol{z}_1}$
     - $dx = \frac{\partial J}{\partial \boldsymbol{x}} = dz_1 \frac{\partial \boldsymbol{z}_1}{\partial \boldsymbol{x}} = dz_1 \mathbf{W}_1^T$
   - Other gradients:
     - dW2 $= \frac{\partial J}{\partial W_2} = \frac{\partial J}{\partial z_2} \frac{\partial z_2}{\partial \mathbf{W}_2} = \boldsymbol{h}^T dz_2$
     - db2 $= \frac{\partial J}{\partial b_2} = \frac{\partial J}{\partial z_2} \frac{\partial z_2}{\partial \boldsymbol{b}_2} = dz_2$ (summed over the first dimension)
     - dW1 $= \frac{\partial J}{\partial W_1} = \frac{\partial J}{\partial z_1} \frac{\partial z_1}{\partial \mathbf{W}_1} = \boldsymbol{x}^T dz_1$
     - db1 $= \frac{\partial J}{\partial b_1} = \frac{\partial J}{\partial z_1} \frac{\partial z_1}{\partial \boldsymbol{b}_1} = dz_1$ (summed over the first dimension)

2.2.4 2.d) Number of parameters

Number of parameters: $(D_x+1)H+(H+1)D_y$

2.3 3. word2vec

2.3.1 3.a) Derivative of the skipgram 1 — with respect to $\boldsymbol{v}_c$

Skipgram: given a center word c ($v_c \in \mathbb{R}^n, \boldsymbol{v}_c = \mathbf{V}\boldsymbol{c}$, $\boldsymbol{c} \in \mathbb{R}^{|V|}$ is one-hot-vector of word c), predict the surrounding context words ($2m$ words).

Word prediction softmax function:

$$\hat{\boldsymbol{y}}_o = p(\boldsymbol{o}|c) = \frac{\exp(\boldsymbol{u}_o^T \boldsymbol{v}_c)}{\sum_w \exp(\boldsymbol{u}_w^T \boldsymbol{v}_c)}$$

where $\boldsymbol{o}$ is the expected output word, and $\boldsymbol{u}_w \in \mathbb{R}^n$ ($w \in [1, W]$) are the output word vectors.

$$J_{softmax-CE}(\boldsymbol{o}, \boldsymbol{v}_c, \mathbf{U}) = CE(\boldsymbol{y}, \hat{\boldsymbol{y}}) = -\sum_i \boldsymbol{y} \log \hat{\boldsymbol{y}} = -\boldsymbol{u}_o^T \boldsymbol{v}_c + \log\left(\sum_w \exp(\boldsymbol{u}_w^T \boldsymbol{v}_c)\right) == -\boldsymbol{u}_o^T \boldsymbol{v}_c + \log\left(\sum [\exp(\mathbf{U}\boldsymbol{v}_c)]\right)$$

where $\mathbf{U} \in \mathbb{R}^{|V| \times n}$ is the output word matrix (n: dimension of embedding space; $|V|$: dimension of vocabulary).

$$\frac{\partial}{\partial \boldsymbol{v}_c} CE(\boldsymbol{y}, \hat{\boldsymbol{y}}) = -\boldsymbol{u}_o^T + \sum_i \boldsymbol{u}_i^T \frac{\exp(\boldsymbol{u}_i^T \boldsymbol{v}_c)}{\sum_w \exp(\boldsymbol{u}_w^T \boldsymbol{v}_c)} = -\boldsymbol{u}_o^T + \sum_i \boldsymbol{u}_i^T \hat{y}_i = (\hat{\boldsymbol{y}} - \boldsymbol{y})^T \mathbf{U}$$

since $\boldsymbol{u}_o = \mathbf{U}^T \boldsymbol{y}$, $\sum_i \boldsymbol{u}_i^T \hat{y}_i = \hat{\boldsymbol{y}}^T \mathbf{U}$, where $\boldsymbol{y}$ is the one-hot-vector with 1 at o and 0 elsewhere.

2.3.2 3.b) Derivative of the skipgram 2 — with respect to $\boldsymbol{u}_w$

$$\frac{\partial}{\partial \mathbf{U}} CE(\boldsymbol{y}, \hat{\boldsymbol{y}}) == \frac{\partial}{\partial \boldsymbol{u}_i} CE(\boldsymbol{y}, \hat{\boldsymbol{y}}) = \hat{y}_i \boldsymbol{v}_c + \begin{cases} -\boldsymbol{v}_c & \boldsymbol{u}_i = \boldsymbol{u}_o \\ 0 & \boldsymbol{u}_i \neq \boldsymbol{u}_o \end{cases}$$

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

2.3.3 3.c) Derivative of the skipgram 3 — negative sampling loss

Using the negative sampling loss for the predicted vector v_c; K negative samples (words) are drawn. $\boldsymbol{o}$ is the expected output vector with $o \notin \{1, .., K\}$. Again, for a given word $\boldsymbol{o}$, the output vector is $\boldsymbol{u}_o$.

The negative sampling loss function is

$$J_{neg-sample}(\boldsymbol{o}, \boldsymbol{v}_c, \mathbf{U}) = -\log(\sigma(\boldsymbol{u}_o^T \boldsymbol{v}_c)) - \sum_{k=1}^{K} \log(\sigma(-\boldsymbol{u}_k^T \boldsymbol{v}_c))$$

Solution:

$$\frac{\partial}{\partial \boldsymbol{v}_c} J_{neg-sample} =$$

$$\frac{\partial}{\partial \mathbf{U}} J_{neg-sample} =$$