N3

i) It's desired to maximize the difference between the target class (correct) prediction and the incorrect ones.

I.e. we want to increase the probability mass for the correct class and minimize the prob of incorrect ones.

Finally, the margin expresses the desired margin between predictions similar to the SVM setup.

ii) $\frac{d\,p_k}{d\,o_k} = p_k(1-p_k)$ $\quad \frac{d\,p_i}{d\,o_k} = -p_i p_k$, $\quad k$ – some random index, V – vocab size

$$\frac{d\,J^{(i)}_{hinge}}{d\,o_k} = \frac{d\,J^{(i)}_{hinge}}{d\,\underline{p}^{(i)}} \frac{d\,\underline{p}^{(i)}}{d\,o_k}$$ here $\underline{p}^{(i)}$ – vector of probabilities

1) $$\frac{d\,\underline{p}^{(i)}}{d\,o_k} = \begin{bmatrix} \frac{dp^{(i)}_1}{do_k} & \frac{dp^{(i)}_2}{do_k} & \dots & \frac{dp^{(i)}_V}{do_k} \end{bmatrix}$$

2) $$\frac{d\,J^{(i)}_{hinge}}{d\,\underline{p}^{(i)}} = \begin{bmatrix} \frac{d\,J^{(i)}_{hinge}}{d\,p^{(i)}_1} \\ \vdots \\ \frac{d\,J^{(i)}_{hinge}}{d\,p^{(i)}_V} \end{bmatrix}$$

$$\frac{d\,J^{(i)}_{hinge}}{d\,p^{(i)}_k} = \begin{cases} \sum_{j \neq y_i} (-1)\, \mathbb{I}[x_j > 0] & \text{if } k = y_i \\ 1 \cdot \mathbb{I}[x_k > 0] & \text{else} \end{cases}$$

where $x_j = p_j - p_{y_i} + \text{margin}$.

N4

activations and deltas

One could store only temporary values in memory and the parameters on the harddisk. And whenever necessary, preload memory to the RAM and perform the computations and updates.

Finally we can exploit sparsity of updates (e.g. when one-hot-vector is given as input which is common for language), and use only portions of parameters.

Use SVD or alike methods to reduce the parameter size

```
alpha =

    0.0150

------------------------------
------------------------------
------------------------------
Iteration # 1
Forward pass

s =

    0.4580    0.8690    0.7040
    0.1205    0.1615    0.0440
   -0.4420   -0.1810    0.7040
    0.1195    0.1185   -0.0440


z =

    0.4285    0.7009    0.6069
    0.1199    0.1601    0.0440
   -0.4153   -0.1790    0.6069
    0.1189    0.1179   -0.0440


s_out =

    0.0842
    0.0112
    0.0409
    0.0020


z_out =

    0.0842
    0.0112
    0.0409
    0.0020

loss:  1.951977
Backward pass

delta_out =

   -0.8390
   -0.9778
    0.9984
    1.0000


delta_1 =
```

```
   -0.0168   -0.0252   -0.0755
   -0.0196   -0.0293   -0.0880
         0         0    0.0899
    0.0200    0.0300         0

Updated parameters

w =

    0.0316
    0.0421
    0.0899


W =

    0.6002    0.7003    0.0021
    0.0102    0.4303    0.8799

-----------------------------
-----------------------------
-----------------------------
Iteration # 2
Forward pass

s =

    0.4583    0.8695    0.7055
    0.1205    0.1616    0.0444
   -0.4420   -0.1809    0.7023
    0.1195    0.1185   -0.0436


z =

    0.4287    0.7011    0.6079
    0.1200    0.1602    0.0444
   -0.4153   -0.1790    0.6058
    0.1190    0.1180   -0.0435


s_out =

    0.0977
    0.0145
    0.0338
    0.0048


z_out =

    0.0977
    0.0145
```

```
    0.0338
    0.0048

loss:  1.931873
Backward pass

delta_out =

   -0.8148
   -0.9712
    0.9989
    1.0000


delta_1 =

   -0.0257   -0.0343   -0.0732
   -0.0307   -0.0409   -0.0873
         0         0    0.0898
    0.0316    0.0421         0

Updated parameters

w =

    0.0430
    0.0539
    0.0895


W =

    0.6005    0.7007    0.0042
    0.0106    0.4308    0.8798

------------------------------
------------------------------
------------------------------
Iteration # 3
Forward pass

s =

    0.4588    0.8702    0.7070
    0.1206    0.1617    0.0448
   -0.4419   -0.1808    0.7006
    0.1196    0.1186   -0.0431


z =

    0.4291    0.7015    0.6088
    0.1200    0.1603    0.0448
   -0.4152   -0.1789    0.6048
```

```
    0.1190    0.1180   -0.0431


s_out =

    0.1108
    0.0178
    0.0266
    0.0076


z_out =

    0.1108
    0.0178
    0.0266
    0.0076

loss:  1.912364
Backward pass

delta_out =

   -0.7916
   -0.9647
    0.9993
    0.9999


delta_1 =

   -0.0340   -0.0427   -0.0708
   -0.0415   -0.0520   -0.0863
         0         0    0.0894
    0.0430    0.0539         0

Updated parameters

w =

    0.0543
    0.0655
    0.0890


W =

    0.6009    0.7011    0.0063
    0.0111    0.4314    0.8796
```

N1

1) $\frac{d\mathcal{L}}{dW_{out}} = \frac{d\mathcal{L}}{dy_{out}} \frac{dy_{out}}{dW_{out}} = \frac{d\mathcal{L}}{dy_{out}} \frac{dy_{out}}{ds_{out}} \frac{ds_{out}}{dW_{out}} = (y_{out} - y_{gt}) f_3'(s_{out}) \cdot z_2$ (derivative)

2) $\frac{d\mathcal{L}}{dW_2} = \frac{d\mathcal{L}}{dy_{out}} \frac{dy_{out}}{ds_{out}} \frac{ds_{out}}{dz_2} \frac{dz_2}{ds_2} \cdot \frac{ds_2}{dW_2} = (y_{out} - y_{gt}) f_3'(s_{out}) \cdot W_{out}^T f_2'(s_2) \cdot z_1$

3) $\frac{d\mathcal{L}}{dW_1} = \frac{d\mathcal{L}}{dy_{out}} \frac{dy_{out}}{ds_{out}} \frac{ds_{out}}{dz_2} \frac{dz_2}{ds_2} \frac{ds_2}{dz_1} \frac{dz_1}{ds_1} \frac{ds_1}{dW_1} =$

$= (y_{out} - y_{gt}) f_3'(s_{out}) W_{out}^T f_2'(s_2) \cdot W_2^T \cdot f_1'(s_1) x_{in}$

4) $\frac{d\mathcal{L}}{dW_k} = \frac{d\mathcal{L}}{ds_k} \frac{ds_k}{dW_k} = \delta_k \cdot z_{k-1}$

N2

1) Forward pass: 1.1) $\underbrace{s_1, s_2, s_3, s_{out}}_{\underline{s}}$ $z_{out} = f_{out}(\underline{s}_{out})$

1.2) $\underbrace{z_1, z_2, z_3}_{\underline{z}}, z_{out}$

2) $\mathcal{L} = 0.5(y_{out} - y)^2$ 3) $\underline{\delta}_{out} = \frac{d\mathcal{L}}{dz_{out}} \frac{dz_{out}}{d\underline{s}_{out}} = (y_{out} - y_{gt}) f_{out}'(\underline{s}_{out})$ (tanh)

4) $\underline{\delta}_1 = \underline{\delta}_{out} \cdot \underline{W}^T f_2'(\underline{s})$ ← relu

5) $\Delta W = \underline{\delta}_{out} \cdot \underline{z}^T$ $\Delta W = \underline{\delta}_1 \cdot \underline{z}_0$ (initial input ($\underline{x}_{in}$))

The output log is attached