Name: Omar Rashad Salem

Course: CV - prof.Heba

Assignemnt No.: 2

> ## QUESTIONS

_1)_ What happens to the softmax loss when the predicted probability for the true label increases?

ans: Softmax loss decreases (SVM can get to 0 loss alot easier but Softmax continues to get better _(decrease loss)_ with better true lable scores).

_2)_ What is the key difference between softmax loss and multi-class SVM loss in terms of their approach to handling multi-class classification?

ans: multiclass SVM is margin based loss function while Softmax is has probabilistic & log-likelihood approach.

_3)_ For the multiclass SVM loss

$$Li = \sum_{j \neq y_i} \max(0, s_j - S_y, +1)$$

_a._ At initialization W is small so all s ≈ 0. What is the loss Li, assuming N examples and C classes?

3a-ans: $N(C-1)$

_b._ What if the sum was over all classes? _(Including ▯ = ▯▯)_

3b-ans: the final loss will be increased by $1 * N$

so for $s \approx 0$ answer = $N[(C-1)+1]$

_4)_ For the following scores calculate softmax and multiclass SVM loss if the first class is the correct class.

a. [10, -100, -100]

b. [10, 9, 9]

4a-ans:

$$\begin{gathered}
\text{for SVM loss:} \\
max(0, -100 + 9 + 1) = 0 \\
max(0, -100 + 9 + 1) = 0 \\
L_i = 0 + 0 \\
\therefore L_i(SVM) = 0 \\
. \\
\text{for Softmax loss:} \\
\text{take exp. of all scores: } [e^{10}, e^{-100}, \\
e^{-100}] = [22026.47, 3.72 * 10^{-44}, \\
3.72 * 10^{-44}] \\
\text{normalize: } [1,\ 0,\ 0] \\
-ln(1) > 0 \text{ positive small value}
\end{gathered}$$

$$ln(1) > 0 \text{ positive small value}$$

$$\therefore L_i\,(Softmax) \approx 0$$

4b-ans:

$$\text{for SVM loss:}$$
$$max(0, 9-10+1) = 0$$
$$max(0, 9-19+1) = 0$$
$$L_i = 0+0$$
$$\therefore L_i\,(SVM) 0$$
$$.$$
$$\text{for Softmax loss:}$$
$$\text{take exp. of all scores: } [e^{10}, e^{9}, e^{9}]$$
$$= [22026.47, 8103.1, 8103.1]$$
$$\text{normalize: } [0.5761, 0.2119, 0.2119]$$
$$-ln(0.5761) = 0.5515$$
$$\therefore L_i\,(Softmax) = 0.5515$$

What does the calculated loss tells you about the difference between the two losses?

ans: the SVM can easily equal 0 loss whenever it's satisfied (which only needs the true label to be higher by 1 from any of other scores) also changing the scores a little bit for SVM probably won't change the 0 loss result. Softmax on other hand never fully satisfied but it gets better and better values and separates the true label from other labels more and more by increasing it's probability value

---

> ## QUESTION 5 (programming assignment)

In [87]:

```
#implementing 'calc_loss' fucntion
import numpy as np


def calc_loss( img : np.ndarray, W: np.ndarray, true_label: int): #NOTE: assume 3 classe
s 1)cat, 2)dog, 3)ship
   classes = ["cat", "dog", "ship"] #'true_label' arg is the true label index

   #NOTE: first do linear classifier f(x,W) = Wx + biase
   flattened_img = img.flatten()
   pixels = len(flattened_img)

   #biase vector
   biase = np.ndarray( shape= (len(classes),1) )
   biase.fill(1)

   #cross product Wx
   flattened_img = flattened_img.reshape(pixels,1)
   Wx = np.dot(W , flattened_img)

   linear_classifier_scores = np.add(Wx , biase) #shape = (classes no,)
   linear_classifier_scores = np.array([3.2,5.1,-1.7])


   #NOTE: second do multi class SVM loss
   true_lablel_score = linear_classifier_scores [int(true_label)]

   #max(0 , sj - Syi + 1)
   maximums = [max(0 , score - true_lablel_score + 1) for score in linear_classifier_sco
res ]
   SVM_Loss = np.sum(maximums) - 1
```

```
    #NOTE: finally  do Softmax loss

    #get the exp value of all scores
    exp_vec = np.exp(linear_classifier_scores )

    #get the normalized value of all scores
    non_normalized_sum = np.sum(exp_vec)
    normalized_value = exp_vec[true_label] / non_normalized_sum
    true_label_probability = normalized_value

    #get the Softmax loss
    SOFTMAX_Loss = -np.log(true_label_probability)  #natural log ln()


    return SOFTMAX_Loss , SVM_Loss

    # 24.5325  164.0219  0.1826835  188.73708
```

In [88]:

```
#using the 'calc_loss' function
img = np.array([[56,231],
                [24,2]])
w =np.array([[0.2,-0.5,0.1,2.0],
             [1.5,1.3,2.1,0.0],
             [0,0.25,0.2,-0.3]])

softmax_loss , svm_loss = calc_loss(img= img, W= w , true_label= 0)

print (f"softmax loss = {softmax_loss} \nSVM loss = {svm_loss}")
```

```
softmax loss = 2.04035515280017
SVM loss = 2.8999999999999995
```