Model Answer, Simple Exercise on Recurrent Neural Networks

This exercise is about understanding how a character-generating recurrent network works. We will assume that the training text is as follows: *We are the students of the Master of Science in Applied Computational Science and Engineering, and we are really interested in Machine Learning.*
We will not differentiate between upper and lower-case letters.
We will also assume that the hidden vector h_t is of dimension 100x1 and that the weights/biases are initialized randomly.

1.What is the dimension of the vocabulary vector? What does it contain?

Answer

The following letters are present in the training text (we ignore the difference between upper and lower case) : a, c, d, e, f, g, h, i, l, m, n, o, p, r, s, t, u, w, y (19 characters) and we have also to count the special characters: blank, comma, dot (3 characters) for a vocabulary's dimension of 19+3=22. The (transposed) vocabulary vector is: (a,c,d,e,f,g,h,i,l,m,n,o,p,r,s,t,u,w,y, ,.,,)

2.Using the hot encoding notation, how would you represent the vector associated with the letter d or to the blank space character?

Answer

Transposed of vector associated with the letter d:
(0,0,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0)
Transposed of vector associated with the blank character:
(0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,0,0)

3. How many parameters, - or degrees of freedoms - do we have to train, in the case where we have no bias terms in the calculations of $h_t$and y_t, and in the case where we have bias terms?

Answer

If we follow this morning's course, we know that (assuming that there is no bias term):

$$\boldsymbol{h_t} = \mathbf{tanh}(\boldsymbol{W_{hh}}\boldsymbol{h_{t-1}} + \boldsymbol{W_{xh}}\boldsymbol{x_t})$$

Where $W_{hh}$is of dimension 100x100, and W_{xh} is of dimension 100x22.
If we do not have a bias term for the calculation of h_t and y_t, and since we have

$$\boldsymbol{y_t} = \boldsymbol{W_{hy}}\boldsymbol{h_t}$$

the number of degrees of freedom is equal to the sum of the dimensions of each of the three matrices W, that is: 100x100+100x22+22x100= 14400 parameters to fit. If we have

bias terms in the calculation of $h_t$and y_t, we must add 100 parameters for the calculation of $h_t$and 22 parameters for the calculation of y_t, that is 14400+100+22=14522 unknowns. This means that we should definitely expect some overfitting considering the very short length of our training sentence!

4. Typically the value of h_0 is taken to be the null vector, meaning that:

$$h_1 = \tanh(W_{xh}x_0)$$

Since x_0 is the one hot encoding of the first letter w of the training sentence, the (transposed) vector x_0 is:

(0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,0,0,0,0)

and $W_{xh}x_0$ is the 18th column of W_{xh} . The coordinates of h_1 are between -1 and +1 because of $tanh$. Suppose that the calculation of

$$y_1 = W_{hy}h_1$$

has produced the following values for the transposed of y_1 :

$y_1^T =$ (0.1,-0.1,0.2,0.1,-0.3,0.2,0.4,-0.1,0.2,-0.3,0.4,-0.30,-0.5,0.3,0.2,0.5,-0.1,0.3, 0.1,0.1,0.2,-0.3)

What is the value of the loss function associated with this first calculation?

Answer

We first have to apply Softmax to the above vector, then calculate the cross-entropy between the Softmax result and the target letter. This target is the letter e, hot-encoded as:

(0,0,0,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0)

Hence the cross-entropy will be equal to minus the log of the fourth coordinate of the Softmax image of the y_1^T vector:

Cross-Entropy = - $\boldsymbol{log\left(\frac{e^{0.1}}{\sum_{i=1}^{22} e^{y_{1i}}}\right)}$ **where the $y'_{1i}s$ are the coordinates of the vector y_i.**

We obtain:

Cross-Entropy = - $\boldsymbol{log\left(\frac{e^{0.1}}{\sum_{i=1}^{22} e^{y_{1i}}}\right) = -log\left(\frac{1.105}{24.144}\right) = 1.339}$