AIMS African Institute for Mathematical Sciences RWANDA

Assignment: Foundations of Deep Learning (AMMI)

African Institute for Mathematical Sciences

Quantum Leap Africa

Foundations of Deep Learning

Author: *Lionel Ngoupeyou Tondji*

Date: November 13, 2018

1 Nonlinear Activation Functions

Assume the error back-propagated to y is $\frac{\partial E}{\partial y}$. For each activation function, write the expression for $\frac{\partial E}{\partial x}$ in terms of $\frac{\partial E}{\partial y}$.

We have : $\frac{\partial E}{\partial x} = \frac{\partial E}{\partial y}\frac{\partial y}{\partial x}$

- For Sigmoid:

$$y = \sigma(x) = \frac{1}{1+\exp(-x)}$$

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

  So we get :

  $\frac{\partial E}{\partial x} = \frac{\partial E}{\partial y}\sigma(x)(1-\sigma(x))$

- For Tanh:

$$y = tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}} = \frac{\sinh x}{\cosh x}$$

$$\begin{aligned}\frac{\partial y}{\partial x} &= \frac{\partial}{\partial y}\left(\frac{\sinh x}{\cosh x}\right)\\ &= \frac{\cos^2 hx - \sin^2 hx}{\cos^2 hx}\\ &= 1 - \tanh^2 x\end{aligned}$$

  So we get :

  $\frac{\partial E}{\partial x} = \frac{\partial E}{\partial y}\left(1 - \tanh^2 x\right)$

- For Relu:

$$y = (x)^+ = \max(0,x) = \left\{ \begin{array}{lll} 0 & if & x \leq 0 \\ x & if & x > 0 \end{array} \right.$$

$$\frac{\partial y}{\partial x} = \left\{ \begin{array}{lll} 0 & if & x \leq 0 \\ 1 & if & x > 0 \end{array} \right.$$

So we get :

$$\frac{\partial E}{\partial x} = \left\{ \begin{array}{lll} 0 & if & x \leq 0 \\ \frac{\partial E}{\partial y} & if & x > 0 \end{array} \right.$$

2 Vanishing and Exploding Gradients

Consider a fully-connected linear network with 51 layers, where each layer has the same square $n \times n$ weight matrix W.

Assume that we are given the vector $\frac{\partial E}{\partial y}$, i.e. the gradient of the error with respect to the output, and that $\left\| \frac{\partial E}{\partial y} \right\|_2 = 1$

1. Write down $\frac{\partial E}{\partial h_{k-1}}$ in terms of $\frac{\partial E}{\partial h_k}$ and U, Σ, V, for $1 \leq k \leq 51$.

Let k such that $1 \leq k \leq 51$, we have :

$$h_k = W^k x = W(W^{k-1}x) = Wh_{k-1}$$

So : $\frac{\partial E}{\partial h_{k-1}} = \frac{\partial E}{\partial h_k} \frac{\partial h_k}{\partial h_{k-1}} = \frac{\partial E}{\partial h_k} W = \frac{\partial E}{\partial h_k} U \Sigma V^T$

In the next line we will suppose that $\Sigma = kI$ for simplicity.

From the previous question we have the relation :

$$\frac{\partial E}{\partial h_{k-1}} = \frac{\partial E}{\partial h_k} W$$

So

$$\left\|\frac{\partial E}{\partial h_{k-1}}\right\|_2 = \left\|\frac{\partial E}{\partial h_k}UkIV^t\right\|_2 = |k|\left\|\frac{\partial E}{\partial h_k}UV^t\right\|_2$$

But

$$\left\|\frac{\partial E}{\partial h_k}UV^t\right\|_2^2 = \left(\frac{\partial E}{\partial h_k}UV^t\right)\left(\frac{\partial E}{\partial h_k}UV^t\right)^t = \left(\frac{\partial E}{\partial h_k}UV^tVU^t\left(\frac{\partial E}{\partial h_k}\right)^t\right) = \left\|\frac{\partial E}{\partial h_k}\right\|_2^2$$

and therefore

$$\left\|\frac{\partial E}{\partial h_k}UV^t\right\|_2 = \left\|\frac{\partial E}{\partial h_k}\right\|_2$$

Here we took the l_2 norm $l_2(X) = (XX^t)^{\frac{1}{2}}$ given a vector X, because our vector here is a row vector so it's a $1 \times dim(h_k)$ and because U and V are orthogonal, we have $VV^t = I$ and $UU^t = I$, therefore we have $(UV^tVU^t) = U\left(V^tV\right)U^t = UIU^t = I$

So

$$\left\|\frac{\partial E}{\partial h_{k-1}}\right\|_2 = |k|\left\|\frac{\partial E}{\partial h_k}\right\|_2$$

$$\left\|\frac{\partial E}{\partial h_1}\right\|_2 = |k|\left\|\frac{\partial E}{\partial h_2}\right\|_2 = \ldots|k|^{50}\left\|\frac{\partial E}{\partial h_{51}}\right\|_2 = |k|^{50}\left\|\frac{\partial E}{\partial y}\right\|_2 = |k|^{50}$$

with the assumption that $h_{50} = y$

2. Let $\Sigma = I$, the identity matrix. What is the l_2 norm of $\frac{\partial E}{\partial h_1}$, ie the norm of the gradient with respect to the first layer hidden units ? Explain why.

Previously we saw that:

$$\left\|\frac{\partial E}{\partial h_1}\right\|_2 = |k|^{50}$$

In this case $k = 1$, So we obtain $\left\|\frac{\partial E}{\partial h_1}\right\|_2 = 1$

2. Let $\Sigma = \frac{1}{2}I$. What is the l_2 norm of $\frac{\partial E}{\partial h_1}$? Explain why.

Previously we saw that:

$$\left\|\frac{\partial E}{\partial h_1}\right\|_2 = |k|^{50}$$

In this case $k = \frac{1}{2}$, So we obtain $\left\|\frac{\partial E}{\partial h_1}\right\|_2 = \frac{1}{2^{50}}$

So in this case we have the vanishing gradient problem because $\dfrac{1}{2^{50}} \longrightarrow 0$

3. Let $\Sigma = 2I$. What is the l_2 norm of $\frac{\partial E}{\partial h_1}$? Explain why.

Previously we saw that:

$$\left\|\frac{\partial E}{\partial h_1}\right\|_2 = |k|^{50}$$

In this case $k = 2$, So we obtain $\left\|\frac{\partial E}{\partial h_1}\right\|_2 = 2^{50}$

So in this case we have the exploding gradient problem because $2^{50} \longrightarrow +\infty$

3. Sentence classification

(a) CNN

1. Design a one-layer CNN which first maps the sentence to a vector of length 5 (with the help of convolution and pooling), then feeds this vector to a fully connected layer with soft(arg)max to get the probability values for possible 3 classes.

2. Clearly mention the sizes for your input, kernel, outputs at each step (till you get the final 3*1 output vector from soft(arg)max).

- Input size : 10×300
- Region size : (2, 3, 4, 5, 6)

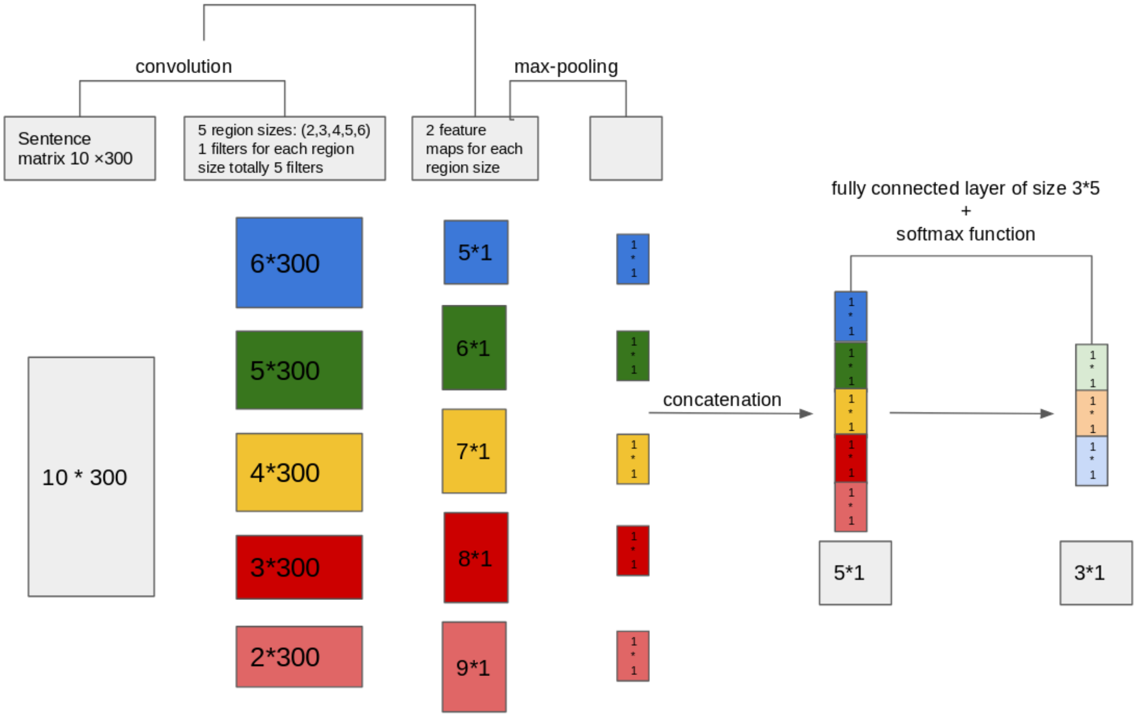
convolution
max-pooling
Sentence matrix 10 ×300
5 region sizes: (2,3,4,5,6) 1 filters for each region size totally 5 filters
2 feature maps for each region size
fully connected layer of size 3*5
+
softmax function
10 * 300
6*300
5*300
4*300
3*300
2*300
5*1
6*1
7*1
8*1
9*1
1*1
concatenation
5*1
3*1

Figure 1: one-layer CNN

- Number of Kernel : 5
- Differents Kernel size K1, K2, K3, K4, K5:
  - K1 : 6×300
  - K2 : 5×300
  - K3 : 4×300
  - K4 : 3×300
  - K5 : 2×300
- Kernel size of max-pooling for each kernel :
  - K1 : 5×1
  - K2 : 6×1
  - K3 : 7×1
  - K4 : 8×1
  - K5 : 9×1

3. Please describe the effect of small filter size vs. large filter size during the convolution? What would be your approach to select the filter sizes for classification task?

From the One-layer CNN we can see that, with small filter we get more information that when we use large filter because we are loosing less information when we use small filter. So in classification task we tend to choose small filter size.

(b) RNN

1. How can a simple RNN which is trained for language modelling be used to get the sentence vector?

Language models assign probability values to sequences of words. For each word in its vocabulary, the language model computes the probability that it will be the next word, but it will only show to the users the top three most probable words corresponding to the possible classes which is the result of the composition of affine transformation and soft(arg)max. So at the end we will end up with the sentence vectors.

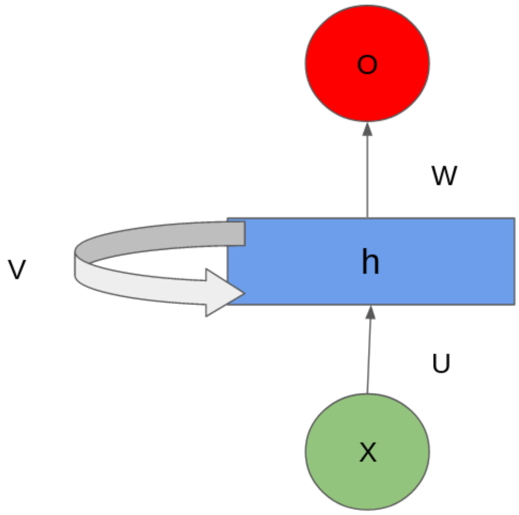

Figure 2: Basic recurrent neural network

2. Design a simple RNN which first maps the sentence to a vector of length 50, then feeds this vector to fully connected layer with soft(arg)max to get the probability values for possible 4 classes.

From the basic recurrent neural network, we design our simple RNN.

3. Clearly mention the sizes of all the RNN components such as your input vector, hidden layer weight matrix, hidden state vectors, cell state vector, output layers (RNN components sizes would be same at each time step).

- h_i : Hidden state vector of size 50×1
- x_i : input vector of size 1×300

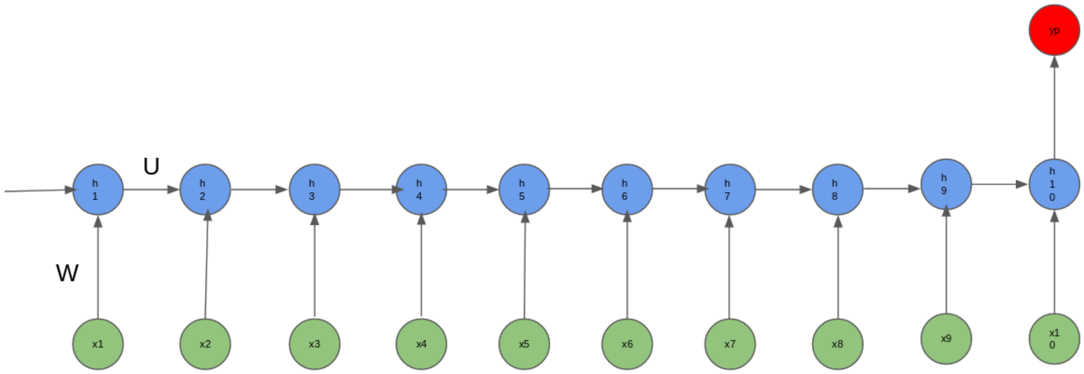

Figure 3: simple RNN

- W, U : Hidden layer weight matrix. shape(U) = 50×50 and shape(W) $= 50 \times 300$
- y_p : output layer of size 50×1

4 Image classification

4.1 New Model Class

See Ipython notebook file

4.2 Zero Padding Model Tensors

See Ipython notebook file

4.3 Valid Padding Model Tensors

See Ipython notebook file