Handy DL

Konopczynski

March 7, 2017

Abstract

Extended calculations and derivatives used in DL in order to make them easier to follow.

1 Activation functions

These functions are used at the end of a layer, usually on outputs of an affine function introducing non-linearity to the equatinon. They are applied on an input vector o_k and produce an output vector of activations p_k.

1.1 Sigmoid

Not centered, values range is $(0, 1)$.

It is defines as:

$$f(x) = \frac{1}{1 + e^{-x}} \tag{1}$$

and it's derivative is:

$$f'(x) = f(x)(1 - f(x)) \tag{2}$$

1.2 Tanh

Centered, values range is $(-1, 1)$.

It is defines as:

$$f(x) = tanh(x) = \frac{2}{1 + e^{-2x}} - 1 \tag{3}$$

and it's derivative is:

$$f'(x) = 1 - f(x)^2 \tag{4}$$

1.3 ReLU

Nice and easy, but kills the gradient if $x < 0$. The values range is $(0, +\infty)$

It is defines as:

$$f(x) = \begin{cases} 0 & \text{for } x < 0 \\ x & \text{for } x \geq 0 \end{cases} \tag{5}$$

and it's derivative is:

$$f'(x) = \begin{cases} 0 & \text{for } x < 0 \\ 1 & \text{for } x \geq 0 \end{cases} \tag{6}$$

1.4 Parametric and leaky ReLU

same as ReLU but with some parameter α instead of 0, when $x < 0$. For Leaky ReLU the α is equal some small value. e.g. $\alpha = 0.01$. The values range is $(-\infty, +\infty)$

It is defines as:

$$f(\alpha, x) = \begin{cases} \alpha x & \text{for } x < 0 \\ x & \text{for } x \geq 0 \end{cases} \tag{7}$$

and it's derivative is:

$$f'(\alpha, x) = \begin{cases} \alpha & \text{for } x < 0 \\ 1 & \text{for } x \geq 0 \end{cases} \tag{8}$$

1.5 Softmax

The softmax function is often used at the end of the network together with cross entropy forming the Cross-entropy Loss for categorical loss. The softmax function is defined as:

$$p_i = \frac{e^{o_i}}{\sum_{k=1}^{N} e^{o_k}}, \forall k = \{1, ..., N\} \tag{9}$$

where N is the total number of classes, and o_k is the outcome for the class k. The softmax computes normalized probability p_i for the class i within a range from 0 to 1.

To compute the derivatives of the softmax one should use the quotient rule

$$f(x) = \frac{g(x)}{h(x)} \rightarrow f'(x) = \frac{g'(x)h(x) - g(x)h'(x)}{(h(x))^2} \tag{10}$$

Lets subset $\Sigma = \sum_{k=1}^{N} e^{o_k}$. Now, using it in the equation:

$$\frac{\partial p_i}{\partial o_j} = \frac{\partial}{\partial o_j}(\frac{e^{o_i}}{\sum_{k=1}^{N} e^{o_k}}) = \frac{\partial}{\partial o_j}(\frac{e^{o_i}}{\Sigma}) \tag{11}$$

$$\frac{\partial p_i}{\partial o_j} = \frac{(e^{o_i})'\Sigma - e^{o_i}(\Sigma)'}{\Sigma^2} \tag{12}$$

where

$$(\Sigma)' = \frac{\partial}{\partial o_j}\sum_{k=1}^{N} e^{o_k} = \frac{\partial}{\partial o_j}e^{o_j} + \sum_{k!=j}^{N} e^{o_k} = e^{o_j} \tag{13}$$

if $i \neq j$:

$$\frac{\partial p_i}{\partial o_j} = \frac{(e^{o_i})'\Sigma - e^{o_i}(\Sigma)'}{\Sigma^2} = \frac{-e^{o_i}(\Sigma)'}{\Sigma^2} = \frac{-e^{o_i}e^{o_j}}{\Sigma^2} = \frac{-e^{o_i}}{\Sigma}\frac{e^{o_j}}{\Sigma} = -p_i p_j \quad (14)$$

if $i = j$:

$$\frac{\partial p_i}{\partial o_j} = \frac{(e^{o_i})'\Sigma - e^{o_i}(\Sigma)'}{\Sigma^2} = \frac{e^{o_j}\Sigma - e^{o_j}e^{o_j}}{\Sigma^2} = \frac{e^{o_j}}{\Sigma}\frac{(\Sigma - e^{o_j})}{\Sigma} = p_j(1 - p_j) \quad (15)$$

2 Loss functions

The loss functions are generally divided into classification or regression loss functions. Depending on the task, one should desing his own loss function. For regression these are usually Rigid Regression or LASSO objectives (basically L_p norms). For classification, the most popular is the cross-entropy loss, but one can use something else, e.g. SVM loss.

2.1 Cross-Entropy Loss

Cross-entropy loss is defined as

$$L = -\sum_{i=1}^{N} y_i \log p_i \quad (16)$$

where

$$p_i = \frac{e^{o_i}}{\sum_{k=1}^{N} e^{o_k}} \quad (17)$$

is a softmax function. Now, using the chain rule, one can compute the partial derivative:

$$\frac{\partial L}{\partial o_j} = \frac{\partial}{\partial o_j}(-\sum_{i=1}^{N} y_i \log p_i) = \frac{\partial p_i}{\partial o_j}\frac{\partial L}{\partial p_i} \quad (18)$$

the solution for the first part one can recall from the section 1.1:

$$\frac{\partial p_i}{\partial o_j} = \begin{cases} -p_i p_j, & \text{if } i \neq j \\ p_j(1 - p_j), & \text{if } i = j \end{cases} \quad (19)$$

The second part:

$$\frac{\partial L}{\partial p_j} = \frac{\partial}{\partial p_j}(-\sum_{i=1}^{N} y_i \log p_i) = -\sum_{i=1}^{N} y_i \frac{\partial}{\partial p_j} \log p_i) = -\sum_{i=1}^{N} y_i \frac{1}{p_i} \quad (20)$$

Taking it all together:

$$\frac{\partial L}{\partial p_i}\frac{\partial p_i}{\partial o_j} = -\sum_{i=1}^{N} y_i \frac{1}{p_i}\frac{\partial p_i}{\partial o_j} = -y_j \frac{1}{p_j}(p_j(1 - p_j)) - \sum_{i \neq j}^{N} y_i \frac{1}{p_i}(-p_i p_j) \quad (21)$$

$$= -y_j(1-p_j) + \sum_{i \neq j}^{N} y_i p_j = -y_j + y_j p_j + \sum_{i \neq j}^{N} y_i p_j = -y_j + \sum_{i=1}^{N} y_i p_j = p_j - p_i \quad (22)$$

since $\sum_{i=1}^{N} y_i = 1$, beacause the sum of probabilities should be equal 1.

2.2 SVM Loss

2.3 Squared Norm

Squared norm is a squared L2 norm, and is defined as

$$L = \frac{1}{N} \sum_{i}^{N} L_i = \frac{1}{N} \sum_{i}^{N} \frac{1}{2} (y_i - p_i)^2 \quad (23)$$

where p_i is the predicted vector, and y_i the vector of true values for N number of classes $i = \{1, ..., N\}$.

The derivative is

$$\frac{\partial L}{\partial p_j} = \frac{1}{N} \sum_{i}^{N} \frac{\partial L_i}{\partial p_j} \quad (24)$$

Using the chain rule, the derivative

$$\frac{\partial L_i}{\partial p_j} = \frac{\partial}{\partial p_j} \frac{1}{2} (y_i - p_i)^2 = \frac{\partial L_i}{\partial (y_i - p_i)} \frac{\partial (y_i - p_i)}{\partial p_j} = (y_i - p_i)(\frac{\partial}{\partial p_j} y_i - \frac{\partial}{\partial p_j} p_i) \quad (25)$$

$$= (y_i - p_i)(0 - 1) = -(y_i - p_i) = p_i - y_i \quad (26)$$

And summarizing over all the features:

$$\frac{\partial L}{\partial p_j} = \frac{1}{N} \sum_{i}^{N} \frac{\partial L_i}{\partial p_j} = \frac{1}{N} \sum_{i}^{N} (p_i - y_i) \quad (27)$$

2.4 Lp norm

In this subsection, we consider all the L_p norms together with the pseudo norms, even though not all of them are differentiable. The L_p norm is defined as

$$||\mathbf{x}||_p = (\sum_{i=1}^{N} |x_i|^p)^{\frac{1}{p}}, \text{ for } p > 0 \quad (28)$$

And the derivative, subseting $\Sigma = \sum_{i=1}^{N} |x_i|^p$, and using the chain rule is

$$\frac{\partial ||\mathbf{x}||_p}{\partial x_j} = \frac{\partial \Sigma^{\frac{1}{p}}}{\partial \Sigma} \frac{\partial \Sigma}{\partial |x_i|} \frac{\partial |x_i|}{\partial x_j} \quad (29)$$

Calculating the three derivatives, one gets:

$$\frac{\partial \Sigma^{\frac{1}{p}}}{\partial \Sigma} = \frac{1}{p}\Sigma^{\frac{1}{p}-1} = \frac{1}{p}(\Sigma^{\frac{1}{p}})^{1-p} = \frac{1}{p}(||\mathbf{x}||_p)^{1-p} \tag{30}$$

$$\frac{\partial \Sigma}{\partial |x_i|} = \frac{\partial}{\partial |x_i|}\sum_{i=1}^{N}|x_i|^p = p\sum_{i=1}^{N}|x_i|^{p-1} \tag{31}$$

$$\frac{\partial |x_i|}{\partial x_j} = \delta_{ij}\frac{x_i}{|x_i|}, \text{ for } x_i \neq 0 \tag{32}$$

where δ_{ij} is the Kronecker delta. Notice, it's not defined at $x_i = 0$.

Getting it all together:

$$\frac{\partial ||\mathbf{x}||_p}{\partial x_j} = \frac{1}{p}(||\mathbf{x}||_p)^{1-p}p\sum_{i=1}^{N}|x_i|^{p-1}\delta_{ij}\frac{x_i}{|x_i|} = ||\mathbf{x}||_p^{1-p}|x_j|^{p-1}\frac{x_j}{|x_j|} \tag{33}$$

$$= \frac{x_j|x_j|^{p-2}}{||\mathbf{x}||_p^{p-1}}, \text{ for } p > 0, \text{ and } x_i \neq 0 \tag{34}$$

3 Affine functions

3.1 Fully conected layer

Fully conected layer is basically a matrix multiplication of an input vector with a matrix of trainable weights.

$$o_i = w_i^T x \rightarrow o = w^T x \tag{35}$$

where $o \in \mathbb{R}^{\mathrm{N}}$ is an output vector of some length N, $x \in \mathbb{R}^{\mathrm{M}}$ an input vector of some length M and $w \in \mathbb{R}^{\mathrm{M}\times\mathrm{N}}$ the weigth matrix of a shape $M \times N$ with parameters to learn.

The derivatives are simply:

$$\frac{\partial o_i}{\partial w_i} = x \rightarrow \frac{\partial o}{\partial w} = x \tag{36}$$

and

$$\frac{\partial o_i}{\partial x} = w_i \rightarrow \frac{\partial o}{\partial x} = w \tag{37}$$

4 Misc and Normalization