Crossentropy

Remi Boutin

June 2019

1 Crossentropy is just the likelihood

Let's assume $\mathbf{D} = \{(X_i, Y_i), i = 1, ..., n\}$ our i.i.d observations such that : Let's assume $(y_1, ..., y_n)$ an i.i.d sample draw from the TRUE distribution P. We define the following for our classes c with $c = 1, ..., K$:

$$
\begin{aligned}
k_c &:= \sum_{i=1}^{N} \mathbb{1}_{y_i=c} \\
p_c &:= \frac{k_c}{N} \\
\hat{k}_c &:= \sum_{i=1}^{N} \mathbb{1}_{\hat{y}_i=c} \\
\hat{p}_c &:= \frac{\hat{k}_c}{N}
\end{aligned} \tag{1}
$$

We can now calculate the likelihood of our data according to the estimated probabilities (the model) :

$$
\begin{aligned}
\mathbb{P}(Y_1 = y_1, ..., Y_n = y_n | model) &= \prod_{i=1}^{N} \mathrm{p}(y_i | model) \\
&= \prod_{i=1}^{N} \hat{p}_{y_i} \\
&= \prod_{C=1}^{K} \hat{p}_c^{k_c}
\end{aligned} \tag{2}
$$

Let's take the log likelihood :

$$
l(y_1, ..., y_n | model) = \sum_{c=1}^{K} k_c \log(\hat{p}_c) \tag{3}
$$

Dividing by N and multiplying by -1 gives us :

$$
-\frac{1}{N} l(y_1, ..., y_n | model) = -\frac{1}{N} \sum_{c=1}^{K} k_c \log(\hat{p}_c) = -\sum_{c=1}^{K} p_c \log(\hat{p}_c) := H(\mathrm{P}, \mathrm{Q}) \tag{4}
$$

where Q corresponds to the estimated distribution.

2 Crossentropy in Deep Learning

Now that we understand the cross entropy as a mean to compare two distributions (between p the real distribution of the categories and q defined by the estimated $\hat{p}_c$), let's take a look at its evaluation during the training of the model.

2.1 Evaluation of the lost

We suppose here that the categories are encoded as a one hot encoded vector, that is, a vector $c = [0, .., 1, 0, .., 0]$ with one on the position corresponding to the right category of the observation and the zero otherwise. To relate to the previous section, we now have :

$$p_c = \begin{cases} 1 & \text{if } y = c \\ 0 & \text{otherwise} \end{cases}$$

We write $L(f(x), y)$ instead of $L(q, p)$ since $q(|x) = f(x)$ and $p(|x) = y$. Therefore, the loss evaluated on one observation gives us:

$$\tag{5}$$

$$\begin{aligned} L(f(x), y) &= -\sum_c p_c * log(q_c) \\ &= -log(q_{\text{real categ}}) \end{aligned} \tag{6}$$

The loss only evaluates the probability the model gives to the real category of your observation. But since $\sum_c p_c = 1$, it changes the whole distribution.

A quick look at the function $f : x \mapsto -log(x)$ give a good insight on how the estimated probability affect the loss function (or not):

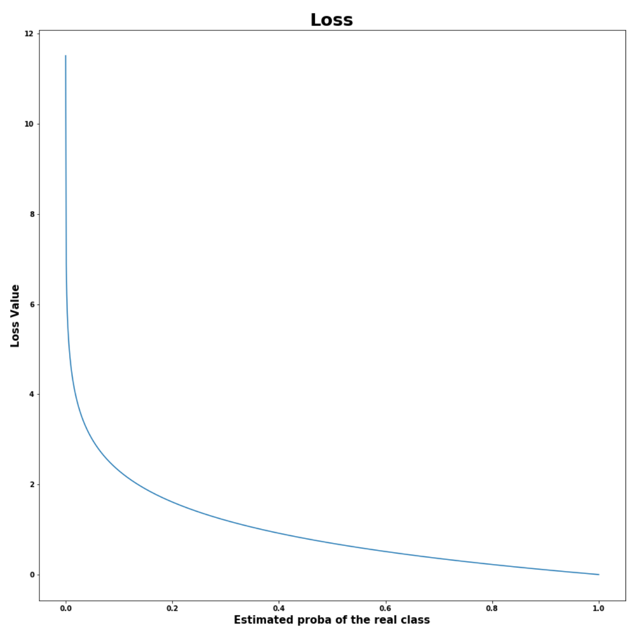

Figure 1: Impact of the probability of the real class on the loss

2.2 Pseudo-code of how it works step by step

- E = Number of epochs
- B = Number of batches per epochs
- $B(b) = \{(x, y) \in \text{batch b at the epoch considered}\}$, we don't specify the epoch for the sake of clear reading
- n_b = batch size

NEED TO IMPROVE : DETAILS ON THE BACK-PROPAGATION ALGORITHM. USE $\tilde{p}_c(\theta)$ AND DO THE DERIVATION OF THE SOFTMAX FUNCTION !

Initialization;
for *epoch = 1 to E* **do**
  for *b = 1 to B* **do**

$$\begin{aligned} c(b) &= \frac{1}{n_b} \sum_{(x,y) \in B(b)} L(f(x), y) \\ &= -\frac{1}{n_b} \sum_{(x,y) \in B(b)} log(\tilde{p}_{\mathrm{c}}) \qquad \text{where c is the real class of the input} \end{aligned} \tag{7}$$

    Perform back-propagation with cost c(b)
  end
end

Algorithm 1: Step by step cross entropy