### **Why CE?**

I remember when I first saw the phrase 'cross-entropy (or CE for short)', I had no idea what it meant. But it seemed to exist in almost every ML/DL paper, where people use it as the loss function of their model. After reading some articles explaining this loss function, I still found it not straightforward enough (considering I'm not a Math/CS student), so I decided to write something about it and try to explain it from a more intuitive perspective.

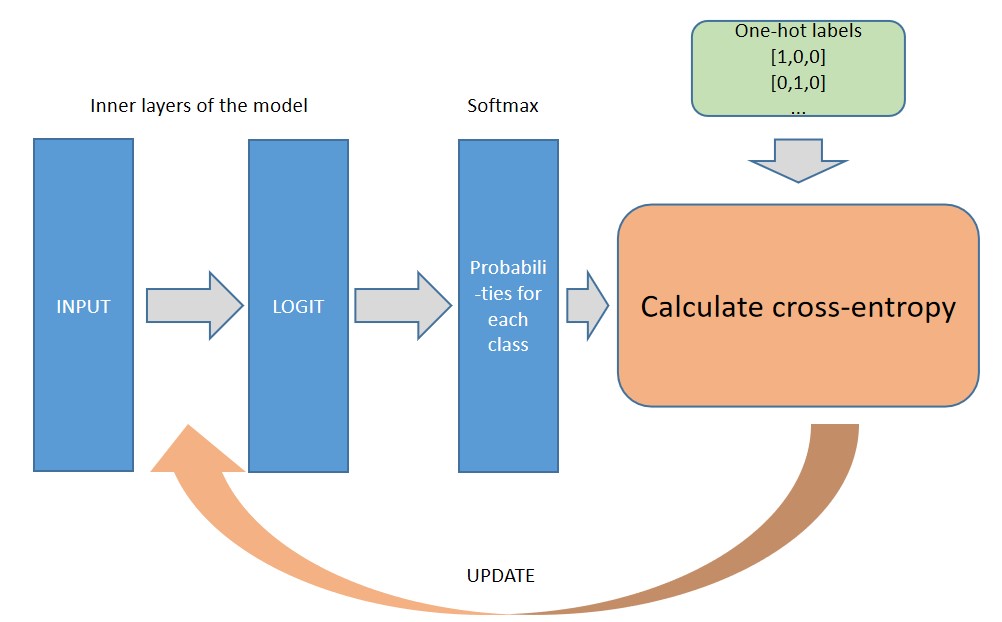
### **What is CE?**

To make it simple, it's just one of the many loss functions (but the most popular one), with which we use to update the parameters of our model.

Notice that loss is different from error rate. Error rate directly answers the question "How many mistakes did the model make (in percentage)?". While loss function could be considered as something to approximate error rate, but is used to update parameters (machine uses this loss function to 'learn' by itself).

### **Where does CE fit in the DL pipeline?**

Before I show an example of calculating CE, I would like to show you where this CE sits in our model and what it does in general.



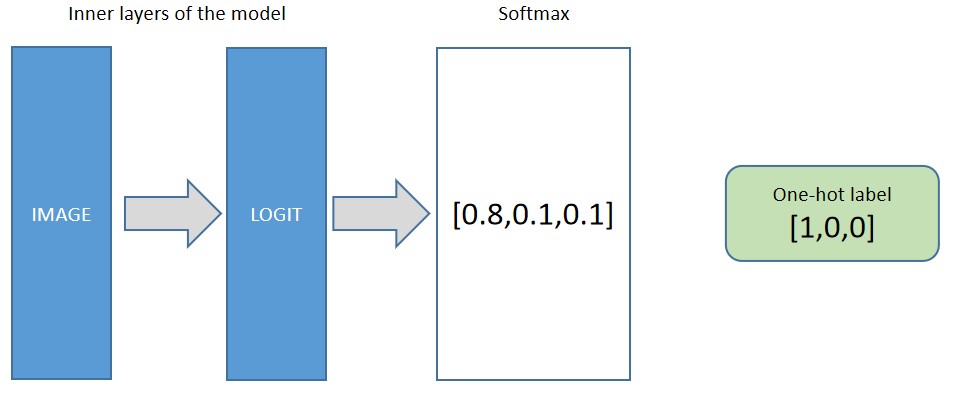
The CE function sits right after Softmax function (which is used to calculate the probabilities of this sample for being each of the classes in a classification problem). It has two inputs: outputs from Softmax function and the one-hot encodings of the targets.

### Details of CE?

As mentioned above, CE is probably the most popular loss function that is used in a classification problem.Thus, looking into the details of CE would help us understand a lot of models proposed by other people and make modifications if needed.

**Here goes an example.**

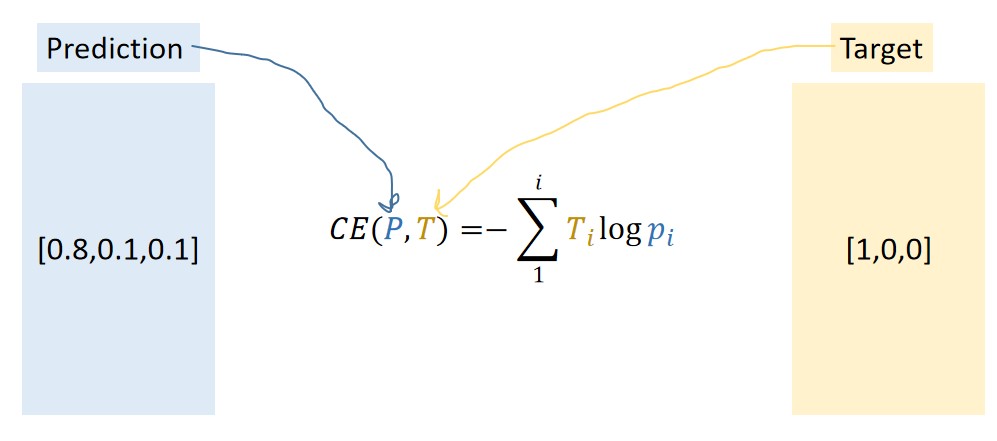
Suppose we need to classify an image into one of the three classes (e.g. cat/dog/human). After we feed this image into our inner layers of the model and the Softmax function, it gives us an output [0.8, 0.1, 0.1], where the three numbers respectively represent the probability of this image being a cat/dog/human. Now we know that the actual label (I.e. target) is [1,0,0] (in a 1-hot-encoded way), the question now would be “How do we figure out a way to quantify the difference between predictions and targets?”



A simple way to do is calculate the dot product of these two vectors. The idea of multiplying these two vectors would give us a results that represents how similar they are.

Since: 0.8\*1=1, 0.1\*0=0, 0.1\*0=0, the dot product is now [1,0,0]. So we have a way to measure their similarity, but how do we measure the difference? To use Cross-entropy!

**Here is how we calculate CE:**



Pi is the output of the Softmax function and Ti is the target. What CE does is that it multiplies the log of each number in the prediction with target and add them all up. Since the log usually makes it negative, we have a ‘-’ at front to make the number positive.

Additionally, we could take the average of this CE(P,T) so that even when we have a large number of data points (I.e. when i is large), we still end up having a small number of cross-entropy.