Cross Entropy as a Loss Function for a Multi-Class Classification Task

Austin Barton

May 5, 2023

Introduction

This short paper will go over the basic notions of *loss* and a ubiquitous function in the classification world known as *cross entropy*. Here, we lay out the fundamental knowledge of what loss and cost functions are and then we develop some basic information theory to give a rigorous and accurate definition of cross entropy in a classification setting. Rather than simply introducing entropy and then saying cross entropy is "like entropy but measures the distance from a true distribution", we introduce relative entropy (KL divergence) and go from there.

1 Training and Optimization

Training and optimization are the bread and butter of statistical and deep learning. A lot of what makes up a good neural network model is our ability to train and optimize it. What follows is some theory behind how we approach classification problems, properties of a specific loss function used for classification problems, and the underlying assumptions we make when utilizing these approaches for classification.

1.1 Loss Function

Definition 1.1. Loss Function. A *loss* function is a real valued function of two variables denoted as $L(\theta, a)$ where $\theta \in \Omega$ and $a \in \mathbb{R}$. It denotes a measure of "loss" in accuracy from estimating a parameter θ as a. If θ is exactly equal to a then $L(\theta, a) = 0$. The a is fixed in the loss, hence, when minimizing we are minimizing over all θ in our parameter space Ω.

This is essentially how our models will "know" how bad their results/predictions are and what direction they should go with their parameters in order to get better results i.e. minimize this loss function.

First, before we move on, a change of notation. Because we are working under a classification setting, a will be a target value and θ will be the output of our predicted target value. That is, our loss function will be evaluated as $L(\hat{y}, y)$. We know that $\hat{y}$ is a function of our input and parameters defining that function that we wish to learn. Let $\vec{\theta}$ now denote the parameters of our function f that maps an input vector $\vec{x}$ to $f(\vec{x}; \vec{\theta}) = \hat{y}$. The reason why I introduced loss with the other notation initially is because loss is generally used in our to learn parameters such as by finding the Bayes estimator for conjugate hyperparameters. If this is unfamiliar, don't worry! Just take note of this crucial change in notation for the following material.

We often use a *Cost Function* which acts as an aggregate measurement of loss in our batch of predictions. Typically the cost function is the sample mean of the loss function. That is,

$$J(\vec{\theta}) = \frac{1}{n}\sum_{i=1}^{n} L(f(\vec{x}_i, \vec{\theta}), y_i) = \frac{1}{n}\sum_{i=1}^{n} L(\hat{y}_i, y_i)$$

Lots of times people use loss and cost interchangeably but here we distinguish the two since it is the most agreed upon notation. Loss is the measurement for *one single* example and cost is the aggregate measurement of loss in an entire batch of observations. The cost function essentially summarizes and estimates the loss for each sample and its minimization corresponds to minimizing the expected value of the loss, or so we hope. The behavior of the cost function is important and multiplying it by a positive constant doesn't affect the problem of minimizing loss so much. For example, you may see people multiply $\frac{1}{2}$ by the cost function when mean squared error is used since it simplifies the derivative a bit.

In theory, the best cost function to minimize is the expected value of our loss function over the training set,

$$J(\theta) = \mathbb{E}_{(\vec{x},y)\in X}(L(f(\vec{x}, \vec{\theta}), y))$$

(**?**). However, the sample mean is a good estimate when the probability distribution of our data generating distribution is uniform or normal. Otherwise, we may need to use a different estimator but the sample mean is typically good (by good, we mean that it is unbiased and consistent). Note in our equation above that the expected value is taken over all pairs $(\vec{x}, y)$, where each y is the target value and a component of our target vector $\vec{y}$.

1.1.1 Loss for Classification versus Regression

It is important now to keep in mind how classification and regression often require much different loss functions. For a regression problem, it is very typical to use squared error as your loss function and mean squared error as your cost function. However, in a classification problem, the notion of distance as in regression is not so clear. And we typically utilize a probabilistic approach with a decision boundary in order to classify our problems. But how do we characterize a loss function that has desirable properties such as differentiability in order to find its minimum and thus, optimize our solution? That is what we will build in the next section.

1.2 Entropy

Definition 1.2. Entropy. If X is a discrete random variable supported on χ with probability mass function p, then the entropy of X is defined as

$$H(X) = -\sum_{x\in\chi} p(x)\log p(x)$$

Note that $H(X) = -\mathbb{E}[\log(p(X))]$

Lemma 1.3. $H(X) \geq 0$.

Proof. $p(x) \geq 0$ for all $x \in \chi$ and thus, $\log p(x) \leq 0$. Further, this implies that $p(x)\log p(x) \leq 0$ for all x. Therefore, $\sum_{x\in\chi} p(x)\log p(x) \leq 0$ which implies that $-\sum_{x\in\chi} p(x)\log p(x) \geq 0$. □

Remarks. Entropy is a measure of the "uncertainty" of a random variable. The term $\log p(x)$ is referred to as the "information content" and $p(x)$ is just the weighted probability of that event occurring.

Also note that log is typically base 2 but not necessarily. In the case that it is base 2, then we refer to the real number $H(X)$ as the expected number of "bits" required to encode X. It forms a lower bound on the number of binary questions needed in order to properly encode the random variable X.

Entropy is a measure of uncertainty of a random variable when we know the probability mass function of this random variable. But what if we don't know p? Well, it doesn't actually change the probability of the event x occurring, but it certainly does change the information content with each associated event observed or question asked. So, we might stipulate that there is some similar idea of entropy that only measures the added relative uncertainty to a random variable with an assumption of probability mass function q that may or may not be correct.

Definition 1.4. Relative Entropy/Kullback-Leichler Divergence. Let X be a discrete random variable supported on χ. The *relative entropy* $D(p||q)$ of the probability mass function p with respect to q is defined by

$$D(p||q) = \sum_{x \in \chi} p(x) \log \frac{p(x)}{q(x)}$$

Note that it is equivalently defined as $D(p||q) = -\mathbb{E}[\log \frac{p(X)}{q(X)}]$.

Discussion. Relative entropy is a measure of the distance between two distributions. In statistics, it arises as an expected logarithm of the likelihood ratio. It is a measure of inefficiency of assuming that the distribution is q when in fact the true distribution is p (**?**).

Theorem 1.5. Information Inequality. Let $p(x), q(x)$, where $x \in \chi$, be two probability mass functions. Then

$$D(p||q) \geq 0$$

with equality if and only if $p(x) = q(x)$ for all x.

Remark. The proof of this theorem essentially follows from Jensen's Inequality and the fact that $\log t$ is a strictly concave function of t. The following proof comes directly from page 28 of "Elements of Information Theory" by Cover and Thomas.

Proof. Recall Jensen's inequality which states that if f is a convex function and X is a random variable, then

$$\mathbb{E}[f(X)] \geq f(\mathbb{E}[X])$$

If f is strictly convex, in the case of equality we have that $\mathbb{E}[X] = X$ with probability 1.

Let $A = \{x : p(x) > 0\}$ be the support of the set of $p(x)$ and χ the support of $q(x)$. Then,

$$-D(p||q) = -\sum_{x \in A} p(x) \log \frac{p(x)}{q(x)} = \sum_{x \in A} p(x) \log \frac{q(x)}{p(x)}$$

Since log is a strictly concave function and $\sum_{x\in A} p(x)\log\frac{q(x)}{p(x)} = \mathbb{E}[\log\frac{q(x)}{p(x)}]$, by Jensen's inequality, we have that

$$\begin{aligned}\sum_{x\in A} p(x)\log\frac{q(x)}{p(x)} &\leq \log\sum_{x\in A} p(x)\frac{q(x)}{p(x)} = \log\sum_{x\in A} q(x)\\ &\leq \log\sum_{x\in\chi} q(x) = \log 1 = 0\end{aligned}$$

Therefore,

$$D(p||q) \geq 0$$

□

Before we introduce something called cross entropy, let's discuss what we've seen so far in the context of loss and the goal of classification. Suppose that given an input of observations with some number of features, we would like to classify the input data with as high precision and accuracy as possible. The classes that the data correspond to may or may not be deterministic, such as data with noise. However, this doesn't change the fact that whether or not the input data has a deterministic correspondence to the classes, their is an underlying true probability distribution associated with the input data and each of the classes which we wish to find. Similar to logistic regression, we may classify according to the class with the highest probability of the data belonging to it given an input. This requires getting as close as possible to the true probability distribution of each of the classes. That is, in order to classify as best as possible, we seek to get as *close* as possible to this *true probability distribution*.

As discussed before, relative entropy measures a notion of "distance" (it is not literally distance by definition. Note that it is not symmetric) between a true underlying probability distribution p and an assumed probability distribution q. Consider,

$$\begin{aligned}D(p||q) = \sum_{x\in\chi} p(x)\log\frac{p(x)}{q(x)} &= \sum_{x\in\chi} p(x)\log p(x) - \sum_{x\in\chi} p(x)\log q(x)\\ &= -H(X) - \sum_{x\in\chi} p(x)\log q(x)\end{aligned}$$

First, note that by the information inequality, $D(p||q) \geq 0$ with equality if and only if $p(x) = q(x)$ for all $x \in \chi$. Additionally, $-H(X) \leq 0$. This alone is enough to show that the term $-\sum_{x\in\chi} p(x)\log q(x)$ must be not only non-negative but also at least as large as $H(X)$ in magnitude. Otherwise, the relative entropy would be negative.

q is our assumed probability mass function here. In our application of classification using neural networks, q is the probability distribution of our classifications given an input of features. We seek to get q as "close" as possible to p. We don't know what p is, but say that we assume that a class is assigned to an input deterministically. Then, that means,

$$-\sum_{x\in\chi} p(x)\log q(x) = -\sum_{x\in\chi} \log q(x)$$

Hence, that term becomes a negative log loss term. This also means that the term $-H(X)$ of relative entropy is now equal to 0 since $p(x) = 1$ for each $x \in \chi$, thus, $\log p(x) = 0$ for each x. This leaves us to minimize the non-zero term above. This makes calculations simpler, but the assumption may seem unjustified in some cases. Even if that is true, it should be noted that no matter what p is,

$H(X)$ is a constant, and we are only trying to minimize $D(p||q)$ by getting q as similar to p as possible. Hence, we are trying to obtain the solution to

$$\text{minimize}[-\sum_{x\in\chi} p(x)\log q(x)]$$

Solving this equivalently minimizes the relative entropy as well. As stated above, in some cases it is reasonable to assume that a class is assigned to an input deterministically. Even more so, it is reasonable in most cases to assume that the true probability distribution for each of the classes will be close, or equal, to 1 for one of the classes and close, or equal, to 0 for the rest of the classes. Hence, for most cases, we assume a deterministic association of input to classes and attempt to obtain the solution to

$$\text{minimize}[-\sum_{x\in\chi} \log q(x)]$$

Now to formalize this unnamed term of relative entropy a bit.

Definition 1.6. Cross-Entropy. The cross-entropy of the probability mass function p with respect to the probability mass function q (or the cross-entropy of q from p) is defined as,

$$CE(p,q) = -\mathbb{E}_p[\log q(x)] = -\sum_{x\in\chi} p(x)\log q(x)$$

Remark. Note that cross-entropy is part of the relative entropy of p with respect to q. Specifically, $CE(p,q) = H(X) + D(p||q)$.

Theorem 1.7. The cross-entropy of p with respect to q satisfies the inequalities,

$$0 \le H(X) \le CE(p,q)$$
$$0 \le D(p||q) \le CE(p,q)$$

Proof. Consider,

$$\begin{aligned} D(p||q) &= \sum_{x\in\chi} p(x)\log p(x) - \sum_{x\in\chi} p(x)\log q(x) \\ &= -H(X) + CE(p,q) \end{aligned}$$

But $D(p||q) \ge 0$ so,

$$\begin{aligned} -H(X) + CE(p,q) &\ge 0 \\ \Rightarrow CE(p,q) &\ge H(X) \end{aligned}$$

Now, entropy is also non-negative and from the definition of cross-entropy we have, $CE(p,q) = H(X) + D(p||q)$. Therefore,

$$\begin{aligned} CE(p,q) &= D(p||q) + H(X) \\ \Rightarrow CE(p,q) &\ge D(p||q) \end{aligned}$$

Therefore,

$$0 \le H(X) \le CE(p,q)$$
$$0 \le D(p||q) \le CE(p,q)$$

□

Remark. The reason we opt for the notation CE to denote cross-entropy is for convenience and clarity. In some other texts, such as "Deep Learning" by Goodfellow et. al. they denote cross-entropy as $H(p,q)$, however, this is easily confused with the notation for joint entropy. The joint entropy between random variables X and Y is typically denoted as $H(X,Y)$. Using CE avoids this needless confusion.

Remark. It should be known that entropy is a strictly concave function. The proof behind this is introductory material, however, for the sake of brevity in this paper we omit it.

Remark. If X has the Bernoulli distribution with parameter p then the entropy of X is defined and denoted by,

$$H(X) = H(p) = -p\log p - (1-p)\log(1-p)$$

Now we introduce binary cross entropy. This is the cross entropy used when there are only two classifications of a dataset. After we develop our notion of binary cross entropy we will generalize it to categorical cross entropy.

Definition 1.8. Binary Cross-Entropy Let X be a random variable supported on $\chi = \{x_1, x_2\}$. Let p be the probability mass function of X and let q be a probability mass function. Then the binary cross-entropy of p with respect to q is,

$$CE(p,q) = -p(x_1)\log q(x_1) - p(x_2)\log q(x_2)$$

Note that $p(x_1)+p(x_2) = 1$ and $q(x_1)+q(x_2) = 1$. Thus, $1-p(x_1) = p(x_2)$ and $1-q(x_1) = q(x_2)$. Then this is,

$$-p(x_1)\log[q(x_1)] - (1-p(x_1))\log[1-q(x_1)]$$

We may shorten the notation of this definition if we let $p(x_1) = p$ and $q(x_1) = q$. Giving us,

$$-p\log q - (1-p)\log(1-q)$$

As we discussed earlier, the true probability mass function is typically assumed to be deterministic. In that case, let $y = I(X = x_1)$, where I is the indicator function. We may rewrite binary cross entropy using this as

$$-y\log q - (1-y)\log(1-q) = -y\log q(y) - (1-y)\log(1-q(y))$$

Where we used the fact that $q(x_1) = q(y)$. This is typically how you will see it written in articles and tutorials online. Remember, this is for one single sample. This equation is what will typically be used as our loss function.

Definition 1.9. The Binary Cross Entropy Cost Function Let $C = [C_1 \dots C_m]^T \in \mathbb{R}^{m\times 1}$ where each C_i is a random variable supported on $\{x_1, x_2\}$ given a parameter $d_i \in \mathbb{R}^{1\times n}$, where $n \in \mathbb{Z}^+$. The binary cross entropy cost of Y is,

$$-\frac{1}{m}\sum_{i=1}^{m}\Big[p(C_i = x_1|d_i)\log q(C_i = x_1|d_i) + p(C_i = x_2|d_i)\log q(C_i = x_2|d_i)\Big]$$

If the classification of each data point is deterministic, then this is

$$-\frac{1}{m}\sum_{i=1}^{m}\Big[y_i\log q(y_i) + (1-y_i)\log(1-q(y_i))\Big]$$

where $y_i = I(C_i = x_1 | d_i)$.

The cost function can be made more general than this, but we are defining it this way to motivate it in this the context of classification problems.

Up until this point, if you have ever read about binary cross entropy anywhere else, this may seem very convoluted. However, in order to justify these definitions and ideas, one needs to keep in mind many of the subtle, yet simple, key notions of probability and information theory.

Definition 1.10. Categorical Cross Entropy In a classification setting, the classification of a data point is typically deterministic given an input.

Let $y_k = \mathbb{P}(Y = k|X) = I(Y = k|X)$ where I is the indicator function and Y is a deterministic random variable supported on the set of classes $C = \{1, \ldots, k\}$. Let $\hat{y}_k = \Pr(\hat{Y} = k|X)$ where $\hat{Y}$ is a random variable. The categorical cross entropy of y_k with respect to $\hat{y}_k$ is

$$-\sum_{k=1}^{K} y_k \log \hat{y}_k$$

Typically the softmax function is used for our class probability estimate.

Definition 1.11. Categorical Cross Entropy Cost Function Given the definition of categorical cross entropy as our loss function, the categorical cross entropy cost of a sample of size m is,

$$-\frac{1}{m}\sum_{i=1}^{m}\sum_{k=1}^{K} y_{i,k} \log \hat{y}_{i,k}$$

1.2.1 Categorical Cross Entropy vs. Sparse Categorical Cross Entropy

Definition 1.12. Label Encoding. Assign each category a distinct integer value. For example, if we have K classes, we can encode each category as the integers 1 through K. A benefit to this is its brevity and lack of memory consumption when implementing. However, it can lead to odd results sometimes since it inherently creates an otherwise unknown or non-existing distance metric between each class. By this, I mean that class 1 for example is much further from class K that 2, however, this notion of distance may be completely arbitrary or wrong.

Definition 1.13. One-hot Encoding. Consider a list of classes $\mathscr{C}$ in some sort of arbitrary order (typically, it's in alphabetical order). If there are C classes, then the one-hot encoded classification is a vector $y \in \mathbb{R}^C$ where y is normalized and consists of one non-zero value equal to 1 in the c^{th} row corresponding to the c^{th} class in $\mathscr{C}$ to classify the data as.

That is, in order to classify the data as the c^{th} class in $\mathscr{C}$, then we output the vector $y = e_c \in \mathbb{R}^C$ where e_c is the c^{th} standard basis vector in $\mathbb{R}^C$.

The difference between categorical cross entropy loss and sparse categorical cross entropy loss lies in how we encode the class to which our output and target variables belong.

2 Conclusions, Summary, Questions

That's everything you would need to know in order to understand the underlying theory behind most classification problems! A lot of this you have probably seen before from logistic regression or another classification problem, but having a more generalized treatment of loss and entropy I believe helps us realize under what assumptions and perspectives we are taking on in order to do these sorts of problems.

The Bird Image dataset source: https://www.kaggle.com/datasets/gpiosenka/100-bird-species