Cross Entropy and KL Divergence

STATISTICS

How cross entropy measures the cost of encoding one distribution with another's codebook, and how KL divergence isolates the extra cost from a model's inaccuracy.

PUBLISHED
September 8, 2026

Series: A Statistical View of Data

6.6 min read

Motivation

So far we've discussed information, entropy, conditional entropy, and mutual information. So what is cross entropy, and why should we care? To answer that, let's look at information theory.

Let's say we have a sequence we want to encode. We'd like to give the most frequent term the shortest encoded codeword to make it more efficient. Let's say we use P our distribution to assign a frequency to each term. In other words, we look at natural English language and assign a probability to each term:

Term	English Frequency (P)	Codeword
A	1/2	0
B	1/4	10
C	1/4	11

Since A was the most frequently occurring term, we give it the shortest code.

However, let's say we're trying to encode a medical document now, where the frequencies differ. Let's say the medical frequencies are given by Q. Medical language is often obscure and uses terms that are less common in natural language English, so our initial encoding will be more expensive. Consider this small example:

Term	English Frequency (P)	Codeword	Medical Frequency (Q)
A	1/2	0	1/4
B	1/4	10	1/2
C	1/4	11	1/4

Now, B is our most common term in the medical document, so the cost of encoding this is higher.

More specifically, the cost under P can be obtained as follows: $1/2(1) + 1/4(2) + 1/4(2) = 1.5$ bits per term

and the cost of encoding Q with an encoder designed for P is as follows: $1/4(1) + 1/2(2) + 1/4(2) = 1.75$ bits per term

Now if we look at what we did carefully, in the second example (cost of encoding for Q), we're really computing the following term $\sum_t Q(t)\text{len}(\text{encode}(\text{t}))$

The $\text{len}(\text{encode}(\text{t}))$ can be obtained from $-\log_2 P(t)$. Note this is only true for our example, but the negative logs give us the ideal length. The proof of this being ideal length is not in the scope of this blogpost.

Putting this together, we have, $-\sum_t Q(t)\log_2 P(t)$

This looks extremely similar to the entropy terms we've been dealing with up until now! However, unlike the previous entropies we've discussed, this operates on two distributions. We call this the cross entropy:

$$H(Q, P) = -\sum_t Q(t)\log_2 P(t)$$

This can be thought of as the average cost of using a codebook designed for one distribution to encode data following another distribution.

> ⓘ **Remarks**
>
> 1. Previously, we looked at a term $H(X, Y)$. It is unfortunate that it uses the same notation as cross entropy. Cross entropy works on two distributions, whereas X, Y are two random variables operating on the same (joint) distribution.
> 2. In general $H(P, Q) \neq H(Q, P)$

Connection to ML

Naive Cross Entropy

In the earlier post, we calculated the entropy of a Wikipedia article. There, we took a naive approach and used word frequencies to build a probability distribution.

Let's call that $Q(t)$. Now, can we train a model to learn a codebook that makes our data cheaper to encode? Let's call this $P_\theta(t)$ which is the model's predicted distribution. Note that here θ refers to the model parameters.

Over time, we want to minimize the average cost of encoding our data. Therefore, we can use cross entropy (recall how, in the earlier section, we treated entropy as the cost of using an encoding scheme under a different distribution) as our loss function. So our loss function here becomes:

$$H(Q, P_\theta) = -\sum_t Q(t) \log_2 P_\theta(t)$$

Training now aims to assign probabilities, such that our average encoding cost is minimized. However, there is a flaw in the reasoning here. This assumes every occurrence of a word receives the same probability, regardless of the preceding words. Since our naive variant only ever cared about the raw frequencies, shuffling all the words would leave the loss function completely unchanged, and you could imagine how problematic this would be for training a model on natural language. To address this, we take a look at conditional cross entropy.

Conditional Cross Entropy

Really, what we want is the conditional cross entropy. We want to find the best possible encoding of the current term in the context of everything we've seen before. To motivate this with a small example, let's say previously the model assigned tea a fixed probability $P_\theta(\text{tea})$. However, if our context is say "I'd like a cup of", then "tea" should be assigned a higher probability and with respect to this context, I should have a shorter encoding.

Putting this more generally, the ideal cost of an observed next word y given some context x then becomes $-\log_2 P_\theta(y|x)$, so following the same idea, averaging over the true data distribution, we now have this formula for cross entropy on a fixed context x

$$H_x(Q, P_\theta) = -\sum_y Q(y|x) \log_2 P_\theta(y|x)$$

Now, if we drop the fixed context, and average over all contexts, we get

$$H(Q, P_\theta) = -\sum_x Q(x) \sum_y Q(y|x) \log_2 P_\theta(y|x)$$

Notice that we can rewrite:

$$Q(y|x) = Q(x, y)/Q(x)$$
$$Q(x, y) = Q(y|x)Q(x)$$

Putting this back into our equation we get:

$$H(Q, P_\theta) = -\sum_{x,y} Q(x, y) \log_2 P_\theta(y|x)$$

This is the conditional cross entropy loss. The two distributions still retain their original roles, the only difference is now they're conditioned on the context.

Relation to KL Divergence

Now, the natural next question is how much of this cost is from the uncertainty in the data, and how much of it is from using the model probabilities.

Let's pin a context x again, we know the cost we get from using the model is

$$-\sum_{y} Q(y|x)\log_2 P_\theta(y|x)$$

Instead, if we had used the true next word distribution, the ideal cost would simply be the entropy:

$$-\sum_{y} Q(y|x)\log_2 Q(y|x)$$

Now, to see how much extra cost we get from using the model probabilities, we simply take the difference between the two:

$$\begin{aligned}
&= -\sum_{y} Q(y|x)\log_2 P_\theta(y|x) - \left(-\sum_{y} Q(y|x)\log_2 Q(y|x)\right) \\
&= \sum_{y} Q(y|x)\log_2 \frac{Q(y|x)}{P_\theta(y|x)}
\end{aligned}$$

This is the KL divergence for a fixed context between the two distributions.

Now, if we average this extra cost over all the contexts, we get:

$$\begin{aligned}
&= \sum_{x} Q(x) D_{KL}(Q(\cdot|x) \parallel P_\theta(\cdot|x)) \\
&= \sum_{x} Q(x) \sum_{y} Q(y|x)\log_2 \frac{Q(y|x)}{P_\theta(y|x)} \\
&= \sum_{x,y} Q(x,y)\log_2 \frac{Q(y|x)}{P_\theta(y|x)} \\
&= \sum_{x,y} Q(x,y)[\log_2 Q(y|x) - \log_2 P_\theta(y|x)] \\
&= \sum_{x,y} Q(x,y)\log_2 Q(y|x) - \sum_{x,y} Q(x,y)\log_2 P_\theta(y|x)
\end{aligned}$$

The first term here is the negative of the conditional entropy. The second term here is the cross entropy between the two distributions.

To put it in words, we showed that:

$$\text{average KL Divergence} = \text{negative conditional entropy} + \text{cross entropy loss}$$

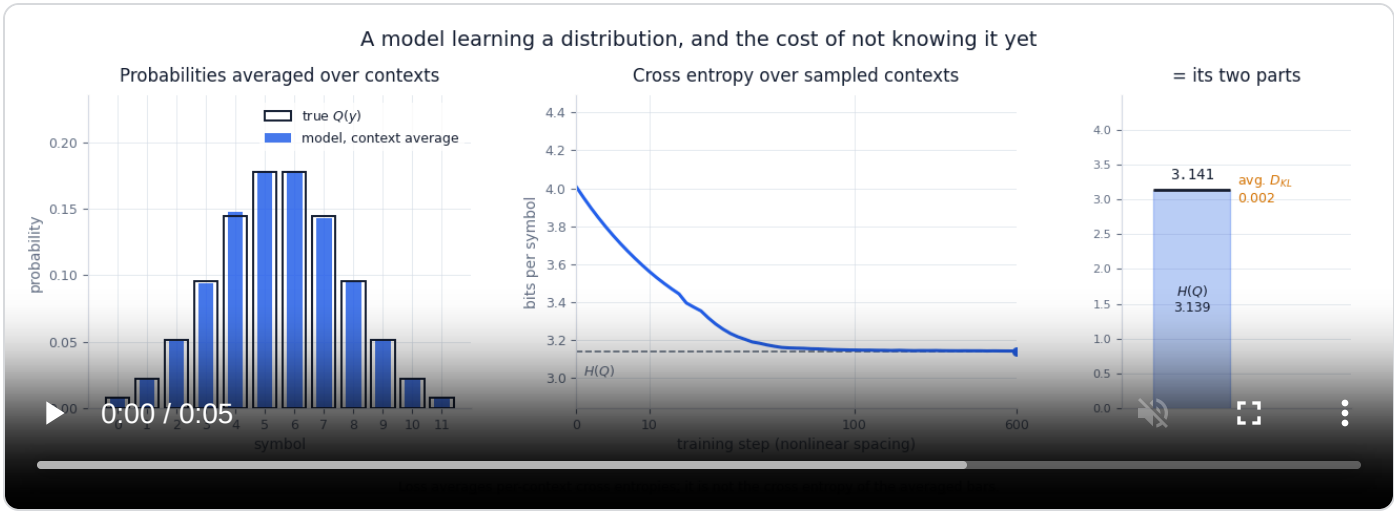
Changing the order, we get:

$$\text{cross entropy loss} = \text{conditional entropy} + \text{KL Divergence}$$

Note, this is now averaged across all contexts. To recap, we started off by fixing the context, then showing how KL divergence is interpreted as extra cost. Then we took the average of the KL divergence and rearranged it to get the average cross entropy loss and conditional entropy.

The interpretation for this is that the model cannot have a loss lower than the conditional entropy, because the KL Divergence is non-negative. See the proof for non-negativity of KL divergence in the appendix.

Seeing the Floor



Training Details

These details are not needed for the conceptual ideas in the post, but if you're curious, here is how the model in the video was trained.

Data

- We create 200,000 independent draws over 12 symbols. Each symbol is assigned a probability using a discretized normal distribution centered at 5.5.
- Symbols 5 and 6 are the most common, while 0 and 11 are rarer.
- There is no sequential pattern in the data: every draw is independent. Because of this independence, conditional entropy equals ordinary entropy: $H(Y \mid X) = H(Y)$.

- We use a sliding window to create training samples. The first four symbols form the input context, and the fifth is the target:

Window	Input context	Target
s_0, s_1, s_2, s_3, s_4	s_0, s_1, s_2, s_3	s_4

Each batch contains 512 examples.

Model Details

- One-block, decoder-only transformer with approximately 9,500 trainable parameters and two attention heads.
- Trained for 600 batches (600 parameter updates).
- Context length of four symbols.

Appendix

Proof of non-negativity of KL Divergence

See this short proof on the non-negativity of KL Divergence.