

---

So far we discussed the problem of inferring something about the world when the unknown properties of the world (e.g.,  $\theta$  or classification rules) were not designed by us, but rather existed there for us to find. In many cases we have some control over these. Indeed, this is the key idea of language where we define a set of symbols that are propagated between us to communicate meaning.

Information theory deals with ways of designing a communication language between two (or more) individuals. The key things we want such a language to achieve is:

- Messages in the language should not be too long, since it will take us too long to deliver them and this might be costly in energy, money etc.
- We should be able to communicate efficiently under noisy conditions. For example, speech would not be a good method of communication if small environmental noises degrade it so that it is incomprehensible.

The first mathematical approach to information theory (IT) was posed by Shannon in 1949. IT has since turned into a field with rich mathematics and applications in many fields, from digital communication to psychology, neuroscience, economy, machine learning and others.

We shall begin with the question of building a compact language for a set of messages. This is known as the problem of lossless coding. As an example, say your friend has traveled abroad and you want to send him a telegram describing the situation at home. Each letter in the telegram is very costly so you want to keep it as short as possible. You promise to send a telegram each week to keep him updated. The most common message you will send is that everything is alright. Also common is the message that you are busy at work. Less common is to say you are ill, and even less common is you are going on holiday. Your message is then a random variable  $X$  which has 4 possible states:  $\{1, 2, 3, 4\}$  with probability  $p(X = i)$  which we abbreviate by  $p_i$ .

Now lets say your telegrams cannot be sent in letters but just sequences of the symbols 0 and 1, what would you do? Seems to make sense to send the bit 0 to say everything is ok, the bit 1 to say you are busy at work and maybe 01 and 10 for the others. Why is that? How can we make a mathematical statement about optimality and how can we construct good codes? Turns out that the measure of entropy plays a key role in this analysis.

**Definition:** Given a random variable  $X$  with distribution  $p(x)$  its entropy is  $H(X) = -\sum_x p(x) \log p(x)$ . The numerical value clearly depends on the base of the log (up to a multiplicative factor). Common choices are base 2 (entropy is then said to be measured in bits) and the natural base  $e$  (is then measured in nats). When relevant, we will write  $H_D(X)$  to denote that entropy is calculated with a base  $D$  for the log.

Some properties of entropy:

- 
- It is always non-negative (exercise). For finite  $X$  it will be zero iff  $p(x) = 1$  for one specific assignment.
  - Entropy is related to  $KL$  divergence;  $H(X) = \log n - D_{KL}[p|u]$  (exercise) where  $u$  is the uniform distribution over  $X$ . From this we can conclude that: If  $p(x) = u(x)$  is a uniform distribution over  $n$  outcomes then  $H(X) = \log n$ . Also, this is the largest value an entropy of a distribution with  $n$  values can take.
  - It can be thought of as the expected value of  $\log \frac{1}{p(x)}$  under the distribution  $p(x)$ . The interpretation is that  $\log 1/p(x)$  is a measure of how surprising (or uncertain) an event is (low probability, high surprise). Extremely surprising events will have low probability and thus contribute little to the entropy. Entropy can thus be interpreted as the expected uncertainty in a variable  $X$ . The choice of log in the entropy turns out to be the only natural choice if some axioms of additivity are required.

The definition extends naturally to multiple variables:

**Definition:** Given two random variables  $X, Y$  and their joint distribution  $p(x, y)$  the joint entropy is defined as:  $H(X, Y) = -\sum_{x,y} p(x, y) \log p(x, y)$ . In other words it is the entropy of the distribution  $p(x, y)$ .

Another key question we want to ask is: if we observe the value of  $Y$  how uncertain are we about the value of  $X$ . If the variables are completely dependent (i.e., one is a function of the other) then there should be no uncertainty left.

**Definition:** Given two random variables  $X, Y$  and their joint distribution  $p(x, y)$  the conditional entropy  $H(X|Y)$  is defined as:

$$H(Y|X) = \sum_x p(x)H(Y|X = x) \quad (1)$$

where  $H(Y|X = x)$  is the entropy of the distribution over  $Y$  given by  $p(y|X = x)$ .

Some properties of conditional entropy:

- It is non-negative. Equals zero (for finite  $X, Y$ ) iff  $H(Y|X = x) = 0$ , or in other words if  $p(y|X = x)$  has probability one just for one assignment. In other words when  $Y$  is a function of  $X$ .
- The following holds:  $H(X, Y) = H(X) + H(Y|X)$  (and also  $H(X, Y) =$

---

$H(Y) + H(X|Y)$ ). **Proof:**

$$\begin{aligned}
 -\sum_{x,y} p(x,y) \log p(x,y) &= -\sum_{x,y} p(x)p(y|x) \log p(x)p(y|x) \\
 &= -\sum_{x,y} p(x)p(y|x) \log p(x) - \sum_{x,y} p(x)p(y|x) \log p(y|x) \\
 &= -\sum_x p(x) \log p(x) + \sum_x p(x) \left[ -\sum_y p(y|x) \log p(y|x) \right]
 \end{aligned}$$

- If  $X$  and  $Y$  are independent then  $H(Y|X) = H(Y)$ . **Proof:** Because of independence we have that  $p(y|x) = p(y)$  for all  $x, y$ . Thus:

$$H(Y|X = x) = -\sum_y p(y|x) \log p(y|x) = H(Y) \quad (2)$$

and then  $H(Y|X) = \sum_x p(x)H(Y) = H(Y)$ .

- if  $X, Y$  are independent then  $H(X, Y) = H(X) + H(Y)$ , i.e., entropy decomposes into individual entropies.

The next key question is what is the reduction in uncertainty that results from seeing  $Y$  (in other words how much information do we gain about  $X$  from viewing  $Y$ ). There are several equivalent definitions of this measure known as Mutual information.

**Definition:** Given two random variables the mutual information between them is defined as:

$$I(X; Y) = \sum_{x,y} p(x,y) \log \frac{p(x,y)}{p(x)p(y)} = D_{KL}[p(x,y)|p(x)p(y)] \quad (3)$$

Some properties of the mutual information:

- $I(X; Y)$  is non negative and equals zero iff  $X$  and  $Y$  are independent.
- $I(X; Y) = H(X) + H(Y) - H(X, Y)$ .
- $I(X; Y) = H(Y) - H(Y|X) = H(X) - H(X|Y)$ . This conforms with our intuition of what information should be.
- $I(X; Y) \leq \min(H(X), H(Y))$
- $I(X; X) = H(X)$ .

Another interesting thing that follows from the above is that  $H(Y) - H(Y|X) \geq 0$ , so that  $H(Y) \geq H(Y|X)$ . This is known as the fact that **conditioning reduces entropy**. It's not surprising since seeing  $X$  can only decrease our uncertainty about  $Y$ .

## 1 Lossless Coding

As mentioned earlier, we are interested in the problem of coding a variable  $X \in \{1, \dots, n\}$  using an alphabet with  $D$  symbols. We shall denote the alphabet by  $\{0, \dots, D - 1\}$  (to be consistent with binary codes which use  $\{0, 1\}$ ).

A  $D$ 'ary code for  $X$  is defined as a function  $c(x) : X \rightarrow \{0, \dots, D - 1\}^*$  where the superscript  $*$  denotes we can use any sequence of these  $D$  symbols. We shall be specifically interested in the lengths of words in the code, and denote by  $\ell_x$  the length of the codeword for the message  $X = x$ .

We will be interested in sending sequences of encoded words (i.e., sentences). These will look like (in the binary case): 00011001... If we had words 0 and 00 we would not be able to decode this unambiguously. Thus we will require that our code is uniquely decodable. In other words any sequence of bits corresponds to exactly one sequence of codewords.

One class of uniquely decodable code is *prefix codes*. A code is a prefix code if no codeword is a prefix of another codeword. Thus 0, 00 is not a prefix code but 0, 11, 10 is a prefix code. It can be seen that such a code can be represented as a tree where each node has at most  $D$  children and the leaves represent the codewords.

It is important to emphasize that we are interested in recovering the exact  $X$  that was sent. We do not allow mistakes. In some cases it is natural to allow mistakes (e.g., when  $X$  is continuous). The field of rate-distortion-theory deals with this case.

We would like to make our code as short as possible. Clearly, if a given value  $x$  is not very commonly sent we should not worry too much about assigning it a long codeword. Thus what we want to minimize is the expected code length:

$$L(c) = \sum_x p(x)\ell_x \quad (4)$$

We shall now see that much can be said about the optimal (i.e., minimal) value of  $L(c)$  can if and how it can be attained.

A key result we will need is the Kraft-McMillan theorem, which characterizes code-lengths for prefix codes.

**Kraft-McMillan Inequality:** Given a  $D$ 'ary prefix code  $c(x)$  with code-lengths  $\ell_x$  it holds that:

$$\sum_x D^{-\ell(x)} \leq 1 \quad (5)$$

Conversely, given a set of lengths  $\ell(1), \dots, \ell(n)$  that satisfy the above inequality, there exists a prefix code with these lengths.

We will use the above to provide a lower bound on  $L(c)$  for *any* prefix code  $c$ . This is a typical information theoretic result, in that it tells us what the best we can hope for is.

The natural approach to bounding this is to minimize  $L(c)$  subject to the constraint that  $c$  is a prefix code.

$$\begin{aligned} \min_{\ell} \quad & \sum_x p(x)\ell(x) \\ \text{s.t.} \quad & \sum_x D^{-\ell(x)} \leq 1 \end{aligned} \quad (6)$$

**Theorem (Shannon):**  $L(c)$  is lower bounded by  $H_D(X)$  for any code  $c$ .

**Proof:**

$$\begin{aligned} L(c) - H_D(X) &= \sum_x p(x)\ell_x + \sum_x p(x) \log_D p(x) \\ &= -\sum_x p(x) \log_D D^{-\ell_x} + \sum_x p(x) \log_D p(x) \\ &= \sum_x p(x) \log_D \frac{p(x)}{D^{-\ell_x}} \\ &= \sum_x p(x) \log_D \frac{p(x)}{D^{-\ell_x} / \sum_{\bar{x}} D^{-\ell_{\bar{x}}}} - \log_D \sum_{\bar{x}} D^{-\ell_{\bar{x}}} \\ &= D_{KL}[p(x)|D^{-\ell_x} / \sum_{\bar{x}} D^{-\ell_{\bar{x}}}] - \log_D \sum_{\bar{x}} D^{-\ell_{\bar{x}}} \geq 0 \end{aligned}$$

Where in the last line we used the non-negativity of  $KL$  and Kraft-McMillan. Equality is attained when  $\sum_x D^{-\ell_x} = 1$  and  $p(x) = D^{-\ell_x}$ . In other words if  $p(x)$  is a distribution such that  $\log_D p(x)$  are integers. For example  $p(x) = [0.5, 0.25, 0.125, 0.125]$  will correspond to an optimal code of lengths 1, 2, 3, 3 (e.g., the code is 0, 10, 110, 111).

So ideally we would like to take  $\ell_x = \log_D \frac{1}{p(x)}$ . This will usually not result in integral lengths. What happens if we round it up?

**Theorem:** There always exists a code  $c$  with lengths  $\ell_x = \lceil \log_D \frac{1}{p(x)} \rceil$ . Its expected length satisfies  $L(c) \leq H_D(X) + 1$ . This code is known as a Shannon-Fano code.

**Proof:** First we need to show that  $\ell_x$  are valid lengths for a prefix code (i.e., satisfy Kraft-McMillan). Indeed:

$$\sum_x D^{-\lceil \log_D \frac{1}{p(x)} \rceil} \leq \sum_x D^{-\log_D \frac{1}{p(x)}} = \sum_x p(x) = 1 \quad (7)$$

Next, calculate the expected code length:

$$L(c) = \sum_x p(x)\ell(x) = \sum_x p(x) \lceil \log_D \frac{1}{p(x)} \rceil \leq \sum_x p(x) \left[ \log_D \frac{1}{p(x)} + 1 \right] = H_D(X) + 1 \quad (8)$$

We can conclude that if  $c^*$  is the optimal code for  $p(x)$  then it satisfies:  $H_D(X) \leq L(c^*) \leq H_D(X) + 1$ . We have shown that the upper bound can be achieved. In fact, it can be shown that there exists an optimal algorithm for this case which finds the best possible code. This algorithm is known

as Huffman coding. A disadvantage of Huffman coding is that it needs to know  $p(x)$  exactly to be optimal.

It turns out we can do better if we are willing to code blocks of words together. In other words instead of coding and sending each word  $X$  separately, we will consider a new variable  $S = X_1, \dots, X_n$  where  $X_i$  are drawn independently. We now build a code for  $S$  and send  $X$ s in blocks of  $n$ . It's not immediately clear that this should buy us anything, but it does!

Denote by  $c(s)$  a code for  $S$ . Then we are interested in  $\frac{1}{n}L(c)$  which is the number of code symbols per coded  $X_i$ . We then have, by the above bounds that:

$$\begin{aligned} H_D(X_1, \dots, X_n) &\leq L(c) \leq H_D(X_1, \dots, X_n) + 1 \\ \frac{1}{n}H_D(X_1, \dots, X_n) &\leq \frac{1}{n}L(c) \leq \frac{1}{n}H_D(X_1, \dots, X_n) + \frac{1}{n} \\ H_D(X) &\leq \frac{1}{n}L(c) \leq H_D(X) + \frac{1}{n} \end{aligned}$$

Thus asymptotically the Shannon-Fano code will have expected length per sent  $X$ .