Computation and Information

Before quantum computers

Contents

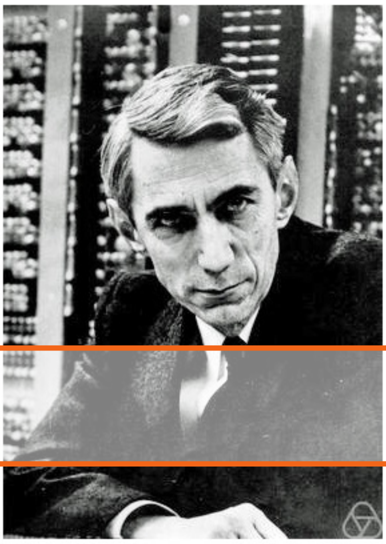
I Computation

II Information

Computation

II Information

1. Measures of information



This is the age of information. We open documents, consume media, exchange text messages, perform videoconferences or watch the weather forecast to name a few examples. But what is exactly information? Can we quantify it? Is it possible to say that one weather forecast contains more information than another? In this chapter, we will learn that this is indeed possible.

1.1 Surprising information

Let us warm up with a series of questions. The premise of all of them is the following: For some reason you want to know the weather forecast for tomorrow, but don't want to watch it or read it yourself. However, in order to get some information that allows you to choose appropriately your clothing for the next day, you ask a friend to send you a message to your phone every night with a summary of the forecast. You agree on a very simple encoding for the message, your friend will send a 1 if the forecast predicts precipitations and 0 otherwise. Think about these questions and come back to them after reading the whole chapter.

Location	**Days with no rain**	**Days with rain**
Rotterdam	212	153
Atacama desert	360	5

Table 1.1: Summary of precipitations in the year 2018.

Exercise 1.1 Let us assume that you are living in the Atacama desert where it rarely rains. You receive a 0. How much information does this message carry? ■

Exercise 1.2 Now let us assume that you live in the Netherlands where it does rain quite often, but certainly not every day. You receive a 0. Does the message contain information? What about if you receive a 1? Does the 1 message contain more or less information than the message 0? ■

Exercise 1.3 Finally, let us assume that you live in the Netherlands but (boldly) also that you are aware of the current season. You receive a 0. Does the message 0 carry the same information in summer and in winter? ■

Before you continue reading, pause for a moment and think what is common in your answers.

We have posed these questions to suggest a relation between the amount of information a message provides and how surprising it is. We will make this connection stronger in the rest of these chapter.

1.2 Refresher on probability theory

A basic understanding of probability theory is essential for the material that follows. Let us review the fundamental concepts and definitions together with the notation that we will use here. As we will only deal with discrete probability distributions, the definitions that follow are not fully general but sufficient for our purposes. If you have troubles following this section and doing the exercises please go back to your undergraduate text on the topic.

Given a finite set $\mathscr{X}$, we call a probability distribution a function $p : \mathscr{X} \to [0,1]$, that is a function from the elements of $\mathscr{X}$ to the closed interval in the real line between zero and one, with the condition that $\sum_{x\in\mathscr{X}} p(x) = 1$. Note that it follows automatically from our definition that for all $x \in \mathscr{X}$ $p(x) \geq 0$.

■ **Example 1.1** Let $\mathscr{X} = \{\text{tails}, \text{heads}\}$ we could define a probability distribution function p such that $p(\text{heads}) = 0.3$ and $p(\text{tails}) = 0.7$. ■

■ **Example 1.2** An important example is the uniform distribution. Given a finite set $\mathscr{X}$, a uniform distribution on the set $\mathscr{X}$ is a function p that for all $x \in \mathscr{X}$ assigns the value

$$p(x) = \frac{1}{|\mathscr{X}|} \,.$$

where, we denote by $|\cdot|$ the number of elements in the set. ■

We define an ensemble X as the tuple of a probability distribution p_X together with its domain $\mathscr{A}_X$. Generally, we will refer to $\mathscr{A}_X$ as the sample space of X and to its elements as events. In information theory, X typically models an object in a communications setup (see 3.1), in this context it is common to call $\mathscr{A}_X$ the alphabet of X and refer to its elements as letters.

Note that we can extend the definition of p_X to any subset $\mathscr{S} \subseteq \mathscr{A}_X$:

$$p_X(\mathscr{S}) = \sum_{x\in\mathscr{S}} p_X(x) \tag{1.1}$$

■ **Example 1.3** Let X be an ensemble with alphabet $\mathscr{A}_X = \{1,2,3\}$ and with p_X the uniform distribution. Then if $\mathscr{S} = \{1,2\}$, $p(\mathscr{S}) = 1/3 + 1/3 = 2/3$ ■

Abusing notation, we will also call event any subset of the alphabet of an ensemble. For this particular case of set we will drop the calligraphic notation for sets. Let a and b be two events in $\mathscr{X}$, we define $a \cup b$ and $a \cap b$ as the union and intersection of a and b. $a \cup b$ is the event that contains all outcomes belonging to a, to b and to both, we will also denote the event $a \cup b$ by a or b. $a \cap b$ is the event that contains all outcomes belonging to both a and b, we will also denote the event $a \cap b$ by a and b. Two events are disjoint if their intersection is null.

Given an ensemble X and two events a, b we say that they are independent if:

$$p_X(a \text{ and } b) = p_X(a)p_X(b) \tag{1.2}$$

Let a and b be two events with non zero probability. We call $p_X(a|b) = p_X(a \text{ and } b)/p_X(b)$ the conditional probability of a given that b occurs. It follows that if and only if a and b are independent $p_X(a|b) = p_X(a)$.

In the following, we will use the explicit notation $p_X, \mathscr{A}_X$ for the probability distribution of ensemble X and its alphabet whenever confusion can arise but we will drop the subscript whenever possible.

Exercise 1.4 Let X be an ensemble modelling two fair coins. Identify two events a, b that are independent and verify that $p_X(a|b) = p_X(a)$ ■

Given two alphabets $\mathscr{A}_X, \mathscr{A}_Y$ we can define a joint ensemble on them with sample space or alphabet the direct product: $\mathscr{A}_{XY} = \mathscr{A}_X \times \mathscr{A}_Y$. We can associate, as well, a probability distribution function to map all tuples (x,y) to $[0,1]$.

The probability of an event in the joint ensemble is equally defined as the sum of the probability of the individual events. In particular, we can define for every $x \in \mathscr{X}$ the probability of $p_X(x)$ as the sum of $p_{XY}(x,y)$ for all $y \in \mathscr{Y}$:

$$p_X(x) = \sum_y p_{XY}(x,y) \tag{1.3}$$

and equivalently $p_Y(y)$:

$$p_Y(y) = \sum_x p_{XY}(x,y) \tag{1.4}$$

■ **Example 1.4** Consider n repetitions of an experiment, each repetition can be modelled by ensemble X and events in different experiments are independent. We can model the set of n repetitions via the joint ensemble $X_1 \dots X_N$, where X_i is the ensemble associated with the i-th experiment, and joint the probability distribution is given by:

$$p_{X_1 \dots X_N}(x_1, x_2, \dots, x_n) = \prod_{i=1}^{n} p_X(x_i) \tag{1.5}$$

■

A random variable V on the ensemble X is a numerical function from the elements of $\mathscr{A}_X$ to (typically) the real line. That is, a function $V : \mathscr{A}_X \to \mathscr{A}_V$, where $\mathscr{A}_V$ is a finite subset of the reals. The random variable V induces an ensemble with alphabet $\mathscr{A}_V$ and probability distribution p_V where p_V is given by:

$$p_V(v) = \sum_{x \in \mathscr{A}_X : V(x) = v} p_X(x) \tag{1.6}$$

for all $v \in \mathscr{A}_V$.

The mean or expectation of a random variable is given by:

$$\mathbb{E}[V] = \sum_{x \in \mathscr{A}_X} p_X(x) V(x) = \sum_{v \in \mathscr{A}_V} p_V v \tag{1.7}$$

1.3 Axiomatic derivation of entropy

Let us now try to understand what type of functions can quantify information in a satisfactory way. Let us make this investigation more precise. In particular, suppose that given some ensemble X we observe the occurrence of an event $x \in \mathscr{A}_X$. As we informally argued in the introduction, the information we gain seems to be related to the likelihood of the event we observed. But how can we make this intuition quantitative?

A function that quantifies information will be a function from a subset of $\mathscr{A}_X$ to the reals. Let us call this function h. Then given some event x, $h(x)$ will be some number that will quantify the information we learn. Let us discuss what properties an ideal information quantifier should have.

- The measure should be non-negative, that is, an event gives either none or some information, but it can not give negative information. That is, for all events $x \in \mathscr{A}_X$ we require:

$$h(x) \geq 0 \tag{1.8}$$

- Suppose that we buy two lottery tickets in two different lottery games, event x is: "our first ticket wins a prize", event y is: "our second ticket does not win a prize". We expect these two events to be independent and the information content of knowing both events should be the sum of the information of the individual events. The occurrence of two independent events should yield the same information that the occurrence of the single events would provide an observer. If we let h be an information measuring function

$$p_X(x \text{ and } y) = p_X(x)p_X(y) \Rightarrow h(x \text{ and } y) = h(x) + h(y) \tag{1.9}$$

- Following our discussion about information and surprise, we want h to quantify less probable events with a larger value than more probable events. For any two ensembles X, Y and events $x \in \mathscr{A}_X$ and $y \in \mathscr{A}_Y$, we require:

$$p_X(x) < p_Y(y) \Rightarrow h(x) > h(y) \tag{1.10}$$

- The final condition is that we don't want that arbitrarily small changes in probability lead to a change in the information quantity, i.e. h should be a continuous function.

It turns out that there is a very limited set of functions that verify these properties. Given some ensemble X, the unique family of functions is of the form:

$$h(x) = -\log_\lambda p_X(x) \tag{1.11}$$

where $x \in \mathscr{A}_X$ and with $\lambda > 1$ for the measure to be positive. Choosing different values of λ allows us to measure information with different units.

There are some common choices of λ that give rise to well known units of information: if we let $\lambda = 2$, the unit of information is called bit. When $\lambda = 3$ information is measured in trits, for $\lambda = 10$ the unit is called a digit and when $\lambda = e$ nat. Unless stated otherwise, in the following we will assume that $\lambda = 2$ and will let $\log = \log_2$.

Definition 1.3.1 Given an ensemble X the information measured in bits of an event $S \subset \mathscr{A}_X$ is given by:

$$h(\mathscr{S}) = -\log p_X(\mathscr{S}) \tag{1.12}$$

Exercise 1.5 Let X be an ensemble modelling a fair coin, that is with alphabet $\mathscr{A}_X = \{\text{heads}, \text{tails}\}$ and with p_X the uniform distribution. What is the information of the event heads and of the event tails? ■

Let us end this section by checking that all our desired conditions hold. First since the log function is continuous and monotonically increasing in the range $(0, 1]$ it holds that h is also continuous and monotonically decreasing in the range. Finally, if two events a, b are independent,

$p(a \text{ and } b) = p(a)p(b)$ and in consequence

$$\begin{aligned} h(a \text{ and } b) &= -\log\left(p(a \text{ and } b)\right) && (1.13)\\ &= -\log\left(p(a)p(b)\right) && (1.14)\\ &= -\log(p(a)) - \log(p(b)) && (1.15)\\ &= h(a) + h(b) && (1.16) \end{aligned}$$

1.4 Entropy

We define the entropy of an ensemble as the average information content it provides:

Definition 1.4.1 Let X be an ensemble, the entropy of the ensemble is defined as:

$$H(X) = -\sum_x p(x)\log p(x) \tag{1.17}$$

where we take the convention that $0\log 0 = 0$, i.e. adding a zero-probability event to a probability distribution does not affect its entropy.

We can rewrite the definition of entropy as the expectation of the random variable $h(X)$. That is a random variable that associated each event with the negative logarithm of its probability:

$$H(X) = -\sum_x p(x)\log p(x) = E(-\log p(X)) \tag{1.18}$$

Note that entropy only depends on the values of the probabilities. In the following we will sometimes be interested in the entropy a probability distribution independently of an ensemble. We will use the notation $H(p_1,\ldots,p_n)$ to indicate the probability distribution. Let us now investigate some basic properties of entropy that we will use through this course.

Exercise 1.6 Show that entropy can not be negative.

$$H(X) \geq 0$$

■

Definition 1.4.2 A function $f(x) : (a,b) \mapsto \mathbb{R}$ is concave if any two points $x_1, x_2 \in (a,b)$ and any $p \in [0,1]$ verify:

$$f(px_1 + (1-p)x_2) \geq pf(x_1) + (1-p)f(x_2) \tag{1.19}$$

■ **Example 1.5** Some examples of concave functions are $-x^2$, $-x^4$, cosine is concave in the interval $[-\pi/2, \pi/2]$ and the logarithm function. To prove the concavity of these functions, you might recall from your calculus course that if a function is twice differentiable in the interval of interest, then it is concave if and only if the second derivative is non-negative. ■

The following is known as Jensen's inequality and will be of use in the following.

Theorem 1.4.1 — Jensen's inequality. Let $f(x) : (a,b) \mapsto \mathbb{R}$ be a concave function. Then for any set of points $\{x_i\}_{i=1}^n \in (a,b)$ and for any set of positive real numbers $\{p_i\}_{i=1}^n$ such that

$\sum_{i=1}^{n} p_i = 1$:

$$f\left(\sum_{i=1}^{n} p_i x_i\right) \geq \sum_{i=1}^{n} p_i f(x_i)$$

Proof. If $n = 2$, the proof follows by the definition of concavity. We will complete the proof by induction. Let us suppose that it holds for $n = m$:

$$f\left(\sum_{i=1}^{m} p_i x_i\right) \geq \sum_{i=1}^{m} p_i f(x_i) \tag{1.20}$$

and let us show that it implies that it also holds for $n = m+1$. Let

$$x' = \sum_{i=1}^{m} \frac{p_i}{1 - p_{m+1}} x_i \tag{1.21}$$

Then we have from the definition of concavity that:

$$f\left(\sum_{i=1}^{m+1} p_i x_i\right) = f((1 - p_{m+1} x' + p_{m+1} x_{m+1}) \tag{1.22}$$

$$\geq (1 - p_{m+1}) f(x') + p_{m+1} f(x_{m+1}) \tag{1.23}$$

Finally from the induction hypothesis (1.20) we have that:

$$f(x') = f\left(\sum_{i=1}^{m} \frac{p_i}{1 - p_{m+1}} x_i\right) \tag{1.24}$$

$$\geq \sum_{i=1}^{m} \frac{p_i}{1 - p_{m+1}} f(x_i) \tag{1.25}$$

which we can plug in back in (1.23) to complete the proof. ■

Exercise 1.7 The distribution that maximizes entropy for any alphabet is the uniform distribution.

$$H(p_1, \ldots, p_n) \leq \log n$$

■

1.5 Joint entropy, conditional entropy and mutual information

We will now explore three information measures that derive from entropy as we defined it in the previous section. The first measure is joint entropy, which is a direct application of the definition of entropy to a joint source.

Definition 1.5.1 Given two ensembles X and Y the entropy of the joint ensemble XY is given by:

$$H(XY) = -\sum_{x,y} p(x,y) \log p(x,y) \tag{1.26}$$

Exercise 1.3 suggests that the information content depends on the context. The second information measure that we introduce is conditional entropy. First, we can extend in a straightforward way the reasoning in Sec. 1.3 to define an information measure conditional on the knowledge of some event y. It can analogously be proved that a conditional information measure is of the form:

$$h(a|b) = -\log p(a|b) \tag{1.27}$$

Let XY be a joint ensemble, we can define the conditional entropy of X given the event y as the average conditional information:

$$H(X|y) = \sum_x p(x|y)h(x|y) \tag{1.28}$$

and the conditional entropy of X given ensemble Y:

$$H(X|Y) = \sum_y H(X|y) \tag{1.29}$$

Exercise 1.8 Show that $H(X|Y) = H(XY) - H(Y)$. ■

Let us investigate some basic properties of the conditional entropy.

Exercise 1.9 Show that the conditional entropy is non-negative.

$$H(X|Y) \geq 0$$

■

Exercise 1.10 Let X, Y be two random variables. Show that:

$$H(X|Y) \leq H(X)$$

■

Exercise 1.11 Given random variables X and Y if $X = f(Y)$:

$$H(X|Y) = 0$$

■

Exercise 1.12 Show that the following relation holds for any two ensembles XY:

$$H(XY) = H(X) + H(Y|X)$$

■

The third information measure that we introduce is the mutual information:

Definition 1.5.2 Given a joint ensemble XY, we define the mutual information between X and Y by:

$$I(X;Y) = H(X) + H(Y) - H(XY)$$

The mutual information $I(X;Y)$ is a measure of the information shared between the two variables X and Y. Let us make this intuition more precise:

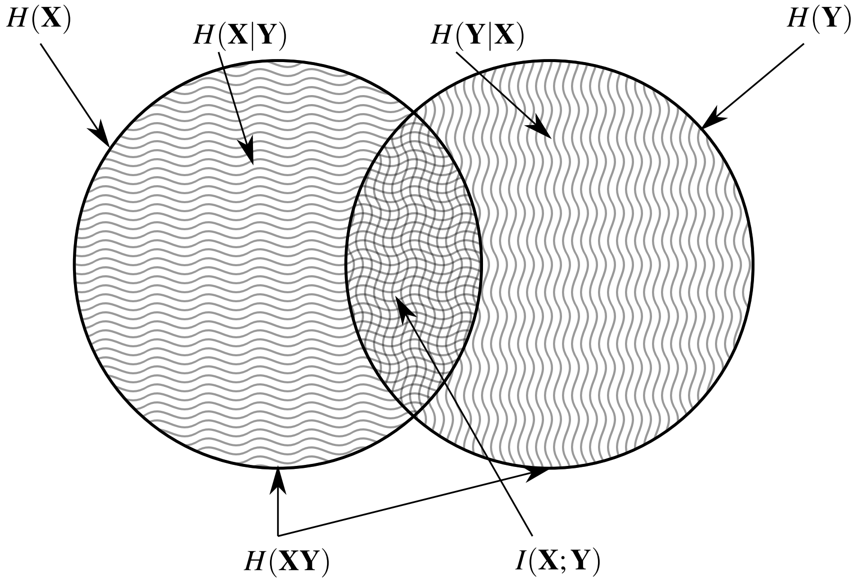

$H(\mathbf{X})$
$H(\mathbf{X}|\mathbf{Y})$
$H(\mathbf{Y}|\mathbf{X})$
$H(\mathbf{Y})$
$H(\mathbf{XY})$
$I(\mathbf{X};\mathbf{Y})$

Figure 1.1: Graphical representation of the information measures.

Exercise 1.13 Show that for any ensemble X: $I(X;X) = H(X)$. ■

Exercise 1.14 Show that $I(X;Y) = 0$ if and only if X and Y are independent. ■

Exercise 1.15 Show that $I(X;Y) \geq 0$ ■

Fig. 1.1 shows the relationship between the four measures that we have defined: entropy, joint entropy, conditional entropy and mutual information.

$$\begin{aligned} I(X;Y) &= H(Y) - H(Y|X) \\ &= H(X) - H(X|Y) \\ &= I(Y;X) \end{aligned} \tag{1.30}$$

1.6 Exercises

Exercise 1.16 Let X be a random variable with $H(X) > 0$ and let $Y = f(X)$.

1. Give one function such that $H(Y) = H(X)$
2. Give one function such that $0 < H(Y) < H(X)$
3. Give one function such that $H(Y) = 0$

■

Exercise 1.17 A classic logical problem (also classic in information theory texts!) states that you receive 12 coins one of which is a counterfeit. The counterfeit is either lighter or heavier than the normal coins, you do not know which is the case. Fortunately, you have access to a two-plate scale that can compare weights.

1. Give a non-trivial bound on the minimum number of weighings that could give the answer.

2. Give an strategy that solves the problem if you know tha the counterfeit coin is heavier than normal coins.
3. Give an strategy that solves the general problem. (difficult)

■

Exercise 1.18 Let X_1, X_2 be two independent random variables modelling two different but otherwise indistinguishable bent coins. Let Y be a binary random variable that models a process where we throw X_1 with probability t and X_2 with probability $1-t$. Show that $H(Y) \geq tH(X_1) + (1-t)H(X_2)$. ■

Exercise 1.19 Let X_1, X_2 be two independent random variables modelling two different bent coins from two different countries, i.e. the outcomes can be clearly distinguished. Let Y be a binary random variable that models a process where we throw X_1 with probability t and X_2 with probability $1-t$. Show that $H(Y) = H(t, 1-t) + tH(X_1) + (1-t)H(X_2)$. ■

1.7 Solutions to selected exercises

Solution. [Exercise 1.5] As p_X is uniform, we have that $p(\text{heads}) = p(\text{tails}) = 1/2$. Hence:

$$h(\text{heads}) = -\log(1/2) = 1 \text{ bit}$$

and

$$h(\text{heads}) = -\log(1/2) = 1 \text{ bit} .$$

■

Solution. [Exercise 1.6]

$$0 \leq p(x) \leq 1 \Rightarrow -\log p(x) \geq 0 \Rightarrow H(X) \geq 0 \tag{1.31}$$

■

Solution. [Exercise 1.7]

$$\begin{aligned} H(p_1, \ldots, p_n) - \log n &= \sum_{i=1}^{n} p_i \log \frac{1}{p_i} - \sum_{i=1}^{n} \frac{1}{n} \log n \\ &= \sum_{i=1}^{n} p_i \log \frac{1}{p_i} - \log n \sum_{i=1}^{n} \frac{1}{n} \\ &= \sum_{i=1}^{n} p_i \log \frac{1}{p_i} - \log n \sum_{i=1}^{n} p_i \\ &= \sum_{i=1}^{n} p_i \log \frac{1}{p_i} - \sum_{i=1}^{n} p_i \log n \\ &= \sum_{i=1}^{n} p_i \log \frac{1}{n p_i} \\ &\leq \log \sum_{i=1}^{n} \frac{1}{n} = 0 \end{aligned} \tag{1.32}$$

where the second equality follows from the fact that a probability distribution adds up to one and the last inequality holds from log being a concave function and applying Jensen's inequality. ■

Solution. [Exercise 1.9] $H(X|Y)$ is a sum of entropies, which are positive as we proved in Exercise 1.6, weighed by the probabilities of each event which are also positive. ■

Solution. [Exercise 1.10]

$$\begin{aligned}
H(X|Y)-H(X) &= \sum_y p(y)\sum_x p(x|y)\log\frac{1}{p(x|y)} - \sum_x p(x)\log\frac{1}{p(x)} \\
&= \sum_y\sum_x p(x,y)\log\frac{1}{p(x|y)} + \sum_{x,y} p(x,y)\log p(x) \\
&= \sum_{x,y} p(x,y)\log\frac{p(x)}{p(x|y)} \\
&= \sum_{x,y} p(x,y)\log\frac{p(x)p(y)}{p(x,y)} \\
&\leq \log\sum_{x,y} p(x)p(y) = 0 \qquad (1.33)
\end{aligned}$$

■

Solution. [Exercise 1.11]

If $X=f(Y)$, then given Y we know X with absolute certainty, in other words, given Y there is just one possible outcome.

$$\begin{aligned}
H(X|Y) &= \sum_y p(y)H(X|y) \\
&= 0 \qquad (1.34)
\end{aligned}$$

■

Solution. [Exercise 1.12]

$$\begin{aligned}
H(XY) &= -\sum_{x,y} p(x,y)\log p(x,y) \\
&= -\sum_x p(x)\sum_y p(y|x)\log p(x)p(y|x) \\
&= -\sum_x p(x)\log p(x)\sum_y p(y|x) \\
&\quad -\sum_x p(x)\sum_y p(y|x)\log p(y|x) \\
&= H(X)+H(Y|X) \qquad (1.35)
\end{aligned}$$

■

Solution. [Exercise 1.19] Since X_1 is chosen with probability t we have that for a symbol $x_1\in\mathscr{X}_1$, the probability $p_Y(x_1)=tp_{X_1}(x_1)$. Similarly for X_2, we have that for a symbol $x_2\in\mathscr{X}_2$ the probability $p_Y(x_2)=(1-t)p_{X_2}(x_2)$. With this observation, let us expand $H(Y)$:

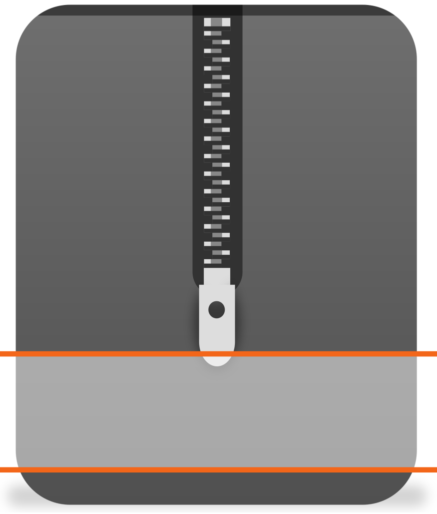
$$\begin{aligned}
H(Y) &= -\sum_y p_Y(y)\log p_Y(y) \\
&= -\sum_{x_1\in\mathscr{X}_1} tp_{X_1}(x_1)\log(tp_{X_1}(x_1)) - \sum_{x_2\in\mathscr{X}_2} tp_{X_2}(x_2)\log(tp_{X_2}(x_2)) \\
&= -t\sum_{x_1\in\mathscr{X}_1} p_{X_1}(x_1)\left(\log(t)+\log(p_{X_1}(x_1))\right) - (1-t)\sum_{x_2\in\mathscr{X}_2} p_{X_2}(x_2)\left(\log(1-t)+\log(p_{X_2}(x_2))\right) \\
&= -t\log t + tH(X_1) - (1-t)\log(1-t) + (1-t)H(X_2) \\
&= H(t,1-t) + tH(X_1) + (1-t)H(X_2)
\end{aligned}$$

■

1.8 Further reading

The mathematical foundations of information theory were to a certain extent developped single handedly by Claude Shannon. His original paper [12] developped the framework and also solved some of the most important problems. The text has not aged with time and remains a greatly written and accessible introduction to the field. A second excellent source for digging deeper into the material is the book of Cover and Thomas [3], it is the reference of the field and widely used in most introductory courses on information theory. Similar to Cover and Thomas but with a more informal treatment, the book of MacKay [8] is also a recommended source. Chapter 2 in both [3] and [8] develop in depth the material of this chapter.

In section 1.3 we sketched an axiomatic derivation of entropy. For a complete discussion on axiomatic derivations of entropy and information please refer to [1, 2, 4, 5].



2. Data compression

In the previous chapter we posed a series of conditions that information measures should possess. We built on top of those conditions and found a series of information measures satisfying them.

In this chapter we will begin a journey to show that not only entropy is a good measure for information according to our desired properties, but also that it carries a strong operational meaning. In fact, we will show that matching our intuition, if an ensemble has a certain entropy, then the length of a message that can communicate the content of the ensemble can not be smaller the entropy of then ensemble.

2.1 Codes

Definition 2.1.1 A symbol code is a map $C : \mathscr{A} \mapsto \mathscr{C}^*$. We associate every symbol a from the alphabet $\mathscr{A}$ with $C(a)$ a sequence of symbols from the code alphabet $\mathscr{C}$. We call $C(a)$ the codeword of a and denote by $|C(a)|$ its length.

■ **Example 2.1** Let X be an ensemble modelling a fair coin. We can consider the codes C_1, C_2 on the ensemble with:
$C_1(\text{tails}) = 00$ and $C_1(\text{heads}) = 111$ $C_2(\text{tails}) = 0$ and $C_2(\text{heads}) = 0$ $C_3(\text{tails}) = 00$ and $C_2(\text{heads}) = 0$ ■

You have probably noticed that the code C_2 in the previous example is not very useful. This consideration motivates the following definitions.

Definition 2.1.2 A code $C : \mathscr{X} \mapsto \mathscr{Y}$ is non-singular if $\forall x, y \in \mathscr{X}$ with $x \neq y$ $C(x) \neq C(y)$.

If we receive a single symbol encoded with a non-singular code, correct decoding is guaranteed, all codewords are different. However, we might consider the use of a code for sending a sequence of symbols from some ensemble. Let X be an ensemble, C be a code on the ensemble and $x = (x_1, \dots, x_n) \in \mathscr{X}^*$ a sequence of elements of $\mathscr{X}$ with finite length. We can extend the definition of C and define its action on x as follows: $C(x) = C(x_1)C(x_2)\dots C(x_n)$. That is, the word associated with x is the concatenation of the codewords for x_1, x_2 until x_n. Under this extended definition, some non-singular codes can lead to erroneous decodings. For instance, consider the code C_3 in the previous example and the word 000. It can be decoded both as (tails,heads) or as (heads,tails).

Definition 2.1.3 A code $C : \mathscr{X} \mapsto \mathscr{Y}$ is uniquely decodable if $\forall x, y \in \mathscr{X}^*$ with $x \neq y$ $C(x) \neq C(y)$.

If we think in the usefulness of a code, unique decodability is a basic requirement. A more practical requirement is that it should be possible to decode symbols as one reads a word instead of waiting until the end of the transmission. A code is called instantaneous if it can be decoded symbol by symbol from left to right without regarding future symbols.

A convenient family of instantaneous codes are so called prefix codes. Before defining them, let $w_1, w_2 \in D^*$, w_1 is a prefix of w_2 if there exist $t \in D^*$ such that w_1 concatenated with t equals w_2. That is: $w_1, t = w_2$.

Definition 2.1.4 A prefix code is a code where no codeword is the prefix of any other codeword.

It is easy to see that prefix codes are instantaneous, indeed as soon as a complete codeword is seen, it is possible to decode the associated symbol. Since the codeword can not be prefix of a subsequent codeword, there is no possibility of confusion. Moreover, it turns out that all instantaneous codes are also prefix, i.e. both concepts are equivalent. In order to prove this, we can argue the contrapositive.

Let C be a non-prefix code and let's see that it implies it is not instantaneous. If C is non prefix there exist codewords w_1, w_2 and t such that $w_1, t = w_2$. Hence, if we receive w_1 we can not decode until additional symbols are received that allow to discriminate between w_1 and w_2.

Exercise 2.1 Let $w_1, w_2 \in D^*$, w_1 is a suffix of w_2 if there exist $t \in D^*$ such that t concatenated with w_1 equals w_2. We call a code a suffix code if no codeword is suffix of any other code word.

1. Are all suffix codes also prefix codes? If yes, prove it. If not, give a counterexample.
2. Are suffix codes uniquely decodable?
3. Are suffix codes instantaneous?

■

In the previous example, it was fairly easy to spot that the code was not uniquely decodable. However, for some codes it is not so simple.

Exercise 2.2 Let $C_1 = \{01, 100, 1101, 0111\}$ and $C_2 = \{01, 100, 1101, 10111, 01011\}$. Are C_1, C_2 uniquely decodable? (Hint: if you can not find the solution, continue reading and return to this exercise once you have understood the Sardinas-Patterson method) ■

Let us now investigate a method to find if a code is uniquely decodable or not. This method was proposed by Sardinas and Patterson [11] and is also known as the method of the dangling suffixes.

The method works as follows:

1. Let C_0 be the set of codewords in the code
2. Let $C_1 = \{w \in \mathscr{A}^* : uw = v,$ where $u, v \in C_0\}$.
3. For $n \geq 2$ do:
   (a) Let $C_n = \{w \in \mathscr{A}^* : uw = v,$ where $u \in C_0, v \in C_{n-1}$ or $u \in C_{n-1}, v \in C_0\}$.
   (b) If C_n is empty, the code is uniquely decodable
   (c) If there is $2 \leq m < n$ such that $C_m = C_n$, i.e. if C_n is a repeated set, then the code is uniquely decodable.
   (d) If the intersection between C_0 and C_n is non-empty. That is if there is some codeword in C_n. Then the code is not uniquely decodable.
   (e) Else we increase n.

This method is a little bit magical. Let us informally investigate some of its properties. One might wonder about two things. First, does the algorithm end for all codes? This first question is relatively straight forward if one observes that the sets C_n for $n \geq 2$ are always composed of suffixes of C_0. Since the number of possible suffixes is finite, the number of sets of suffixes is also finite. Hence at some point the algorithm will end. The second question is whether or not the output of the algorithm is correct. This is a little more complicated and beyond the scope of the course see [10] for more information.

2.2 Code length and fundamental limits

Definition 2.2.1 Given an ensemble X and a code C for this ensemble, we define the average length of the code as follows:

$$l(C) = \sum_{x\in\mathscr{X}} p(x)|C(x)| \tag{2.1}$$

where $|C(x)|$ is the length of the codeword of event x.

■ **Example 2.2** Consider the codes C_1, C_2 for a fair coin. $C_1(\text{tails}) = 0$, $C_1(\text{heads}) = 10$, $C_2(\text{tails}) = 0$, $C_2(\text{tails}) = 1$. The average length of the codes is:

$$L(C_1) = \frac{1}{2}1 + \frac{1}{2}2 = 1.5 \tag{2.2}$$

$$L(C_2) = \frac{1}{2}1 + \frac{1}{2}1 = 1 \tag{2.3}$$

■

Intuitively, it seems that we can not do better than C_2, how do we prove it? In the following we prove a fundamental relation between the length of a code and the entropy of the associated ensemble. Previous to that we need to prove Kraft-MacMillan's inequality. This inequality was first proved by McMillan [9] however the following proof is a simpler version by Karush [3, 7].

Theorem 2.2.1 The length of a uniquely decodable code C for a random variable X taking values in alphabet $\mathscr{Y}$ verifies:

$$\sum_x \frac{1}{|\mathscr{Y}|^{l(x)}} \leq 1$$

Proof. Let $c(x_1,x_2,\ldots,x_k)$ be a concatenation of codewords of aggregated length $l(x_1,x_2,\ldots,x_k) = \sum_{i=1}^k l(x_i)$. Since C is uniquely decodable for any aggregated length k, no more than $|\mathscr{Y}|^k$ different concatenation of codewords can be generated.

We can consider the related expression on the aggregated length:

$$\begin{aligned}
\left(\sum_x \frac{1}{|\mathscr{Y}|^{l(x)}}\right)^n &= \sum_{x_1}\frac{1}{|\mathscr{Y}|^{l(x_1)}}\sum_{x_2}\frac{1}{|\mathscr{Y}|^{l(x_2)}}\cdots\sum_{x_n}\frac{1}{|\mathscr{Y}|^{l(x_n)}} \\
&= \sum_{x_1,x_2,\ldots,x_n}\frac{1}{|\mathscr{Y}|^{l(x_1)+l(x_2)+\cdots+l(x_n)}} \\
&= \sum_{x_1,x_2,\ldots,x_n}\frac{1}{|\mathscr{Y}|^{l(x_1+x_2+\cdots+x_n)}} \qquad (2.4)
\end{aligned}$$

which can also be written as the sum for all possible lengths i of the number T_i of concatenation of n codewords:

$$\begin{aligned}\left(\sum_x \frac{1}{|\mathscr{Y}|^{l(x)}}\right)^n &= \sum_{i=1}^{nl_{max}} \frac{T_i}{|\mathscr{Y}|^i} \\ &\leq \sum_{i=1}^{nl_{max}} \frac{|\mathscr{Y}|^i}{|\mathscr{Y}|^i} \\ &\leq nl_{max} \end{aligned} \tag{2.5}$$

where $l_{max} = \max_x l(x)$. And taking the n-th root in both sides:

$$\sum_x \frac{1}{|\mathscr{Y}|^{l(x)}} \leq (nl_{max})^{1/n} \tag{2.6}$$

now since the limit $\lim_{n\to\infty}(nl_{max})^{1/n} = 1$ and the result holds for all n:

$$\sum_x \frac{1}{|\mathscr{Y}|^{l(x)}} \leq 1 \tag{2.7}$$

■

The Kraft-MacMillan inequality is in a sense an if and only if condition. Given a set of lengths satisfying the inequality, there exists a code C with the same lengths that is a prefix code. Let us see how to construct this associated prefix code.

We first observe that any prefix code can be represented by a tree with each codeword represented by a leave of the tree. For example, consider the code $C = \{0, 10, 1100, 1101, 1110, 1111\}$ and its representation in figure 2.1.

Second, consider a set of lengths $\{l_1, \dots, l_n\}$ which we assume are ordered i.e. $l_1 \leq \dots \leq l_n$ and let us assume that this set verifies the Kraft-MacMillan inequality. We will construct the code by assigning to each codeword a number of leaves from a full binary tree of depth l_n. There are then a total of 2^{l_n} leaves to be assigned. The strategy will be to assign to codeword i $2^{l_n - l_i}$ leaves. We should verify that the number of leaves assigned is not larger than the total number of leaves in the tree. That is:

$$\sum_i 2^{l_n - l_i} \leq 2^{l_n}. \tag{2.8}$$

The inequality holds since if we divide both sides by 2^{l_n} we recover Kraft-MacMillan's inequality.

Finally, we proceed to assign the leaves in the tree to each codeword consecutively and from left to right. The assignment is done in order of length from the shortest to the longest codeword.

We can now show that the length of a uniquely decodable code is lower bounded by the entropy of the random variable.

Theorem 2.2.2 The length of a uniquely decodable code taking values from finite alphabet $\mathscr{Y}$ for random variable X is lower bounded by the entropy of X.

$$L \geq H_{|\mathscr{Y}|}(X)$$

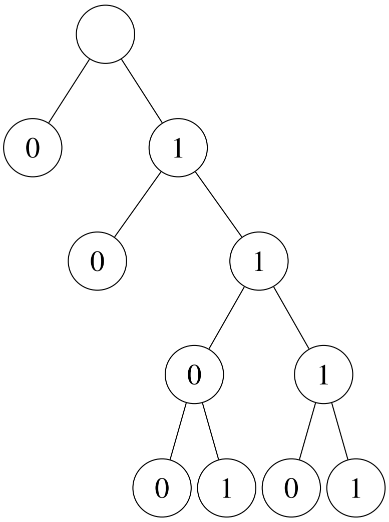


Figure 2.1: Binary tree representing the code $C = \{0, 10, 1100, 1101, 1110, 1111\}$.

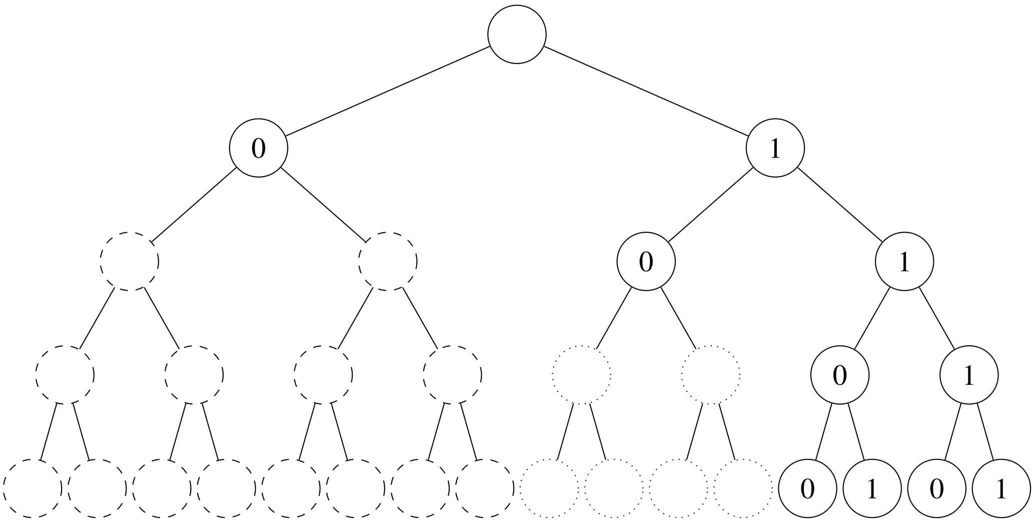


Figure 2.2: Construction of a prefix code from the set of lengths: $1, 2, 4, 4, 4, 4$.

Proof.

$$\begin{aligned}
H_{|\mathscr{Y}|}(X) - L &= \sum_x p(x)\log_{|\mathscr{Y}|}\frac{1}{p(x)} - \sum_x p(x)l(x) \\
&= \sum_x p(x)\log_{|\mathscr{Y}|}\frac{1}{p(x)} - \sum_x p(x)\log_{|\mathscr{Y}|}|\mathscr{Y}|^{-l(x)} \\
&= \sum_x p(x)\log_{|\mathscr{Y}|}\frac{|\mathscr{Y}|^{-l(x)}}{p(x)} \\
&\leq \log_{|\mathscr{Y}|}\sum_x |\mathscr{Y}|^{-l(x)} \\
&\leq \log_{|\mathscr{Y}|} 1 = 0 \qquad (2.9)
\end{aligned}$$

where the first inequality is again an application of Jensen's result Th. 1.4.1 and the second one results from applying McMillan's Th. 2.2.1. ■

One relevant question is: how far is the optimal code from the bound? In exercise 2.4 you will show that at most one bit! Let us first introduce some notation.

Definition 2.2.2 We use the notation $\lceil\rceil$ and $\lfloor\rfloor$ to indicate the rounding "up" and "down" of a real number to the closest integer value. More precisely, let $x \in \mathbb{R}$:

$$\lceil x\rceil = \min\{n \in \mathbb{N} : n \geq x\} \qquad (2.10)$$

$$\lfloor x\rfloor = \max\{n \in \mathbb{N} : n \leq x\} \qquad (2.11)$$

■ **Example 2.3** $\lceil 0.3\rceil = 1, \lfloor 0.3\rfloor = 0, \lceil -0.3\rceil = 0$ and $\lfloor -0.3\rfloor = -1$. ■

Exercise 2.3 Let $p \in (0,1)$, show that $-\lceil\log(1/p)\rceil = \lfloor\log p\rfloor$. ■

Exercise 2.4 Let X be an ensemble and consider a binary code C with lengths $l(C(x)) = \lceil\log 1/p_X(x)\rceil$ for all symbols $x \in \mathscr{X}$. Show that there exist a code C with the indicated lengths that satisfies:

$$H(X) \leq L(C) \leq H(X) + 1 \qquad (2.12)$$

■

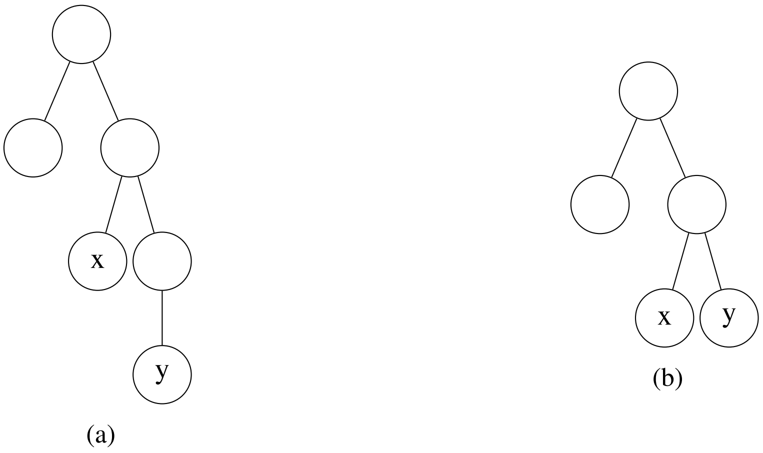
2.3 Huffman codes and their optimality

We will now investigate a scheme that achieves the optimal encoding rate of an ensemble. The algorithm is remarkably simple and was discovered by Huffman [6] while a master student. Let us describe the algorithm.

Given an ensemble X and let C denote the Huffman code of X.

1. Let a, b be two symbols with smallest probability. Create ensemble X' identical to X but replacing a, b with a new symbol c that has probability $p_{X'}(c) = p_X(a) + p_X(b)$.
2. Let $C(a) = C(c)0$ and $C(b) = C(c)1$.
3. If X' has an alphabet with more than one symbol go back to 1.

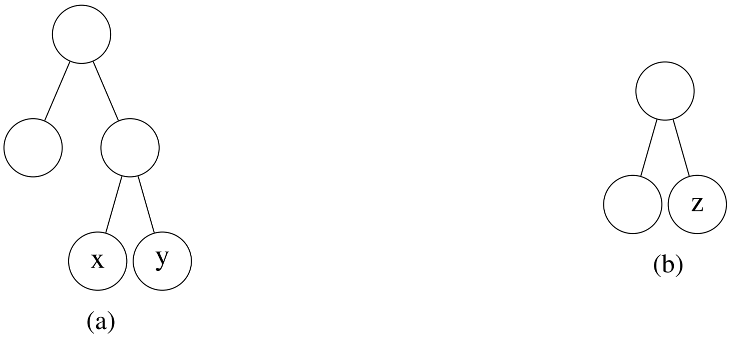
A first observation on the algorithm is that computationally it is very simple. Since at each iteration there is one symbol less it will terminate in $|\mathscr{A}_X|$ iterations.



(a)

(b)

Figure 2.3: On the left we have an abstract representation of a code C where a,b, the two symbols with the longest codewords have different lengths. On the right we have the representation of a new code C' with the longest codeword trimmed.



(a)

(b)

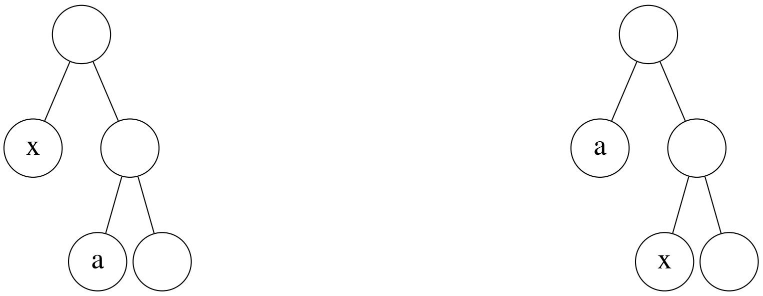
Figure 2.4: On the left we have an abstract representation of a code C where x,y, the two symbols with smallest probability have equal and maximum lengths differing only in the last bit value. On the right we have the representation of a new code C' without symbols x,y and with a new symbol z.

Exercise 2.5 Find the Huffman code of an ensemble X with symbols $\mathscr{A}_X = \{x_1,x_2,x_3,x_4,x_5,x_6\}$ and corresponding probabilities $p_X = \{0.3,0.2,0.18,0.15,0.12,0.05\}$. ■

The prove of optimality of Huffman codes is indirect. What we will prove is that no other code can have lower length based on some properties of the trees associated with Huffman codes. Our discussion is limited to binary trees, for the full discussion, please see the references in Section 2.6.

Lemma 2.3.1 If x,y are the two symbols with smallest probability, there exists an optimal code C where $C\left(x\right),C(y)$ are the longest codewords, have the same size and differ only in the last bit.

Proof. Consider the tree representation of any code C where the two longest words are of unequal size, i.e. $|C(a)| > |C(b)|$ where a,b are the symbols with longest codewords. Then we could construct a new code where we remove the last bits of codeword $C(b)$ until it has length $|C(a)$. The new code would be shorter. See Fig. 2.3 for a graphical representation. Let us then assume that the longest codewords are of the same length, but that at least one of a,b is different from x,y. Suppose that a is neither x or y and consider a new code C' where we swap $C(a)$ with $C(x)$. That is: $C'(a) = C(x)$ and $C'(x) = C(a)$. Now, we will show that the average length of the new code can



(a) (b)

Figure 2.5: On the left we have an abstract representation of a code C where x, one of the two symbols with smallest probability has not maximum length while a which is not one of the two symbols with smallest probability has a maximum length codeword. On the right we have the representation of a new code C' with the codewords of symbols x and a swapped.

not be larger than the average length of C.

$$\begin{aligned} L(C)-L(C') &= \sum_{t\in\mathscr{X}} p_X(t)|C(t)| - \sum_{t\in\mathscr{X}} p_X(t)|C'(t)| && (2.13)\\ &= p_X(x)|C(x)| + p_X(a)|C(a)| - p_X(x)|C'(x)| - p_X(a)|C'(a)| && (2.14)\\ &= p_X(x)(|C(x)|-|C(a)|) - p_X(a)(|C(x)|-|C(a)|) && (2.15)\\ &= (p_X(x)-p_X(a))(|C(x)|-|C(a)|) && (2.16)\end{aligned}$$

The proof is complete since $p_X(x) \leq p_X(a)$ and $|C(x)| \leq |C(a)|$, which implies that $L(C)-L(C')$ is greater than zero. ■

Lemma 2.3.2 For any code C satisfying that the two symbols with smallest probability have codewords of equal length differing only in the last bit and the code $C\prime$ that results from removing symbols x, y and adding symbol z with $p(z) = p(x) + p(y)$.

$$L(C) = L\left(C'\right) + p(z) \qquad (2.17)$$

Proof. Recall that from the statement we have that $|C(x)| = |C(y)| = |C(z)| - 1$ and consider the following chain of equalities:

$$\begin{aligned} L(C)-L(C') &= \sum_t p(t)|C(t)| - \sum_t p(t)|C'(t)| && (2.18)\\ &= p(x)|C(x)| + p(y)|C(y)| - p(z)|C'(z)| && (2.19)\\ &= p(z) && (2.20)\end{aligned}$$

■

The previous two lemmas strongly point to Huffman codes being optimal. We will not complete the proof here. If you are interested, please look at the references provided in Section 2.6

Theorem 2.3.3 Given an ensemble X, the Huffman code C produces an optimal encoding for the ensemble. That is, for any othe uniquely decodable code C' it holds that $L(C) \leq L(C')$.

You might have noticed, that while optimal, Huffman codes are not satisfactory in extreme scenarios. For instance, you could think about the Huffman code for a binary ensemble where one of the symbols have almost unit probability. While the average length is less than one bit from the entropy, if we take the ratio between the average length and the entropy it diverges when one of the elements of the ensemble approaches unit probability.

If we want to transmit not one, but several symbols of some ensemble X, one solution to the situation we described above is to instead of encoding the symbols of the ensemble one by one encode n symbols together. We can model this communication scenario with a joint ensemble that we denote by X^n. The alphabet of the joint ensemble is the cartesian product of the alphabets of the original ensemble and its probability distribution is given by:

$$p_{X^n}(x_1,\dots,x_n) = \prod_{i=1}^{n} p_X(x_i) \tag{2.21}$$

under the assumption that all symbols are drawn independently and identically from the distribution p_X.

■ **Example 2.4** Let X be a binary ensemble with symbols $\{a,b\}$ that occur with probabilities $\{0.7, 0.3\}$. The alphabet of X^2 is $\{a,b\} \times \{a,b\} = \{aa, ab, ba, bb\}$ and the probability of each word is:

$$p_{X^2}(aa) = p_X(a)p_X(a) = 0.49 \tag{2.22}$$
$$p_{X^2}(ab) = p_X(a)p_X(b) = 0.21 \tag{2.23}$$
$$p_{X^2}(ba) = p_X(b)p_X(a) = 0.21 \tag{2.24}$$
$$p_{X^2}(bb) = p_X(b)p_X(b) = 0.09 \tag{2.25}$$

■

Exercise 2.6 Prove that if X, Y are two independent ensembles, then $H(XY) = H(X) + H(Y)$. ■

Exercise 2.7 Prove that $H(X^n) = nH(X)$. ■

Hence, we have on the one hand that $H(X^n) = nH(X)$ and on the other hand that for each extended ensemble there exists a code C_n with length at most one bit from entropy. That is: $L(C_n) \leq H(X^n) + 1 = nH(X) + 1$. If we normalize by n we obtain the following relation:

$$H(X) \leq \frac{L(C_n)}{n} \leq H(X) + \frac{1}{n} \tag{2.26}$$

In consequence, by extending any ensemble enough we can ensure the existence of codes with normalized average length arbitrarily close to the entropy of the original ensemble.

2.4 Exercises

Exercise 2.8 Consider a binary ensemble X with probabilities $\{0.1, 0.9\}$.

1. Find the Huffman code of X and of the ensemble extensions X^2 and X^3.
2. Find the average length of the three codes.
3. Compare the average lengths to the entropy of the corresponding ensemble.

■

Exercise 2.9 A dyadic distribution, is a probability distribution where all symbols have probability 2^{-n} for some integer n. Show that ensembles with dyadic distributions have Huffman codes with average length equal to the entropy of the ensemble. ■

2.5 Solutions to selected exercises

Solution. [Exercise 2.2] We will solve the problem for the first code.
We let $C_0 = \{01, 100, 1101, 0111\}$.
We construct $C_1 = \{11\}$ the set of dangling suffixes from the codewords.
For the next step we take the union of the suffixes in C_1 with respect to the codewords and the suffixes in the set of codewords with respect to the words in C_1. This set is $C_2 = \{01\}$.
The algorithm ends here because C_2 contains a codeword. Hence the code is not uniquely decodable.
■

Solution. [Exercise 2.4] Let us first verify that the lengths given satisfy the Kraft-MacMillan's inequality:

$$\begin{aligned}
\sum_{x\in\mathscr{X}} 2^{-\lceil \log 1/p_X(x)\rceil} &= \sum_{x\in\mathscr{X}} 2^{\lfloor \log p_X(x)\rfloor} && (2.27)\\
&\leq \sum_{x\in\mathscr{X}} 2^{\log p_X(x)} && (2.28)\\
&= \sum_{x\in\mathscr{X}} p_X(x) && (2.29)\\
&= 1 && (2.30)
\end{aligned}$$

Since the inequality is verified, this implies the existence of a prefix code with the given lengths satisfying $H(X) \leq L(C)$.

Let us now estimate the average length of the code:

$$\begin{aligned}
L(C) &= \sum_{x\in\mathscr{X}} p_X(x)|C(x)| && (2.31)\\
&= \sum_{x\in\mathscr{X}} p_X(x)\lceil \log 1/p_X(x)\rceil && (2.32)\\
&\leq \sum_{x\in\mathscr{X}} p_X(x)(\log 1/p_X(x)+1) && (2.33)\\
&\leq H(X)+1 && (2.34)
\end{aligned}$$

■

2.6 Further reading

Both Shannon's original paper [12] and Cover and Thomas [3] (chapter 5) provide further detail into the topics discussed here. A popular introduction to data compression is provided by xkcd (the comic stripe) [13].

An important remark on our exposition is that we have only covered lossless codes. That is codes where no information is lost. While we showed that it is not possible to reliably code at rates below entropy, in many practical applications a certain amount of loss or unreliability is acceptable if in exchange one can increase the compression efficiency. The different theoretical trade-offs and schemes go beyond the scope of this course. For the interested reader we can point to Salomon's encyclopedic treatise on the topic [10].

3. Data transmission

In the previous chapter we investigated the data compression problem. That is, given a source and some noiseless means of communication, the problem is to encode the source in such a way that we minimize the usage of the noiseless communications channel while we allow the receiver to recover the message. Here we will investigate a dual problem, the problem of transmitting a source over a noisy channel. This problem, is the problem that your mobile phone faces each time that it wants to exchange information with the nearest base station. It is also the same problem that your computer faces when it wants to store information on a disk in such a way that it can be recovered at a later time.

3.1 The communications problem

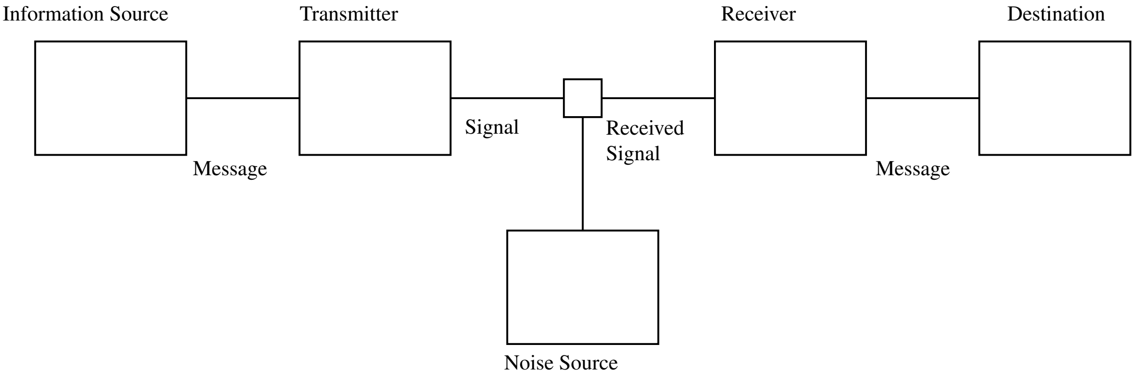


Figure 3.1: This figure reproduces the communications system diagram introduced by Shannon [12].

Let us first of all, depict the building blocks of an idealized communications problem. Our description parallels the one of Shannon [12], see in Fig. 3.1 a graphical representation. The figure shows five entities: an information source, a transmitter, a noise source, a receiver, and a destination. The communications scheme works as follows:

First the information source generates a message m from a set of possible messages M. Then, the transmitter takes m and encodes it into n channel symbols. We define the coding rate R as:

$$R = \frac{\log M}{n} \tag{3.1}$$

The channel is a physical medium of transmission. Mathematically, we can model it as a system taking symbols from input alphabet $\mathscr{X}$ to symbols of output alphabet $\mathscr{Y}$ and characterized by a transition probability matrix that maps the probability of every symbol y if symbol x is sent. The receiver tries to undo the encoding given the noisy received signal and at the end of the scheme the destination receives the $\hat{m}$ possibly identical to m.

3.2 Detection, correction and minimum distance

Let us recall our original example of transmitting the weather forecast. Let us recall that the set of messages is binary : rain and sun. If we are interested in transmitting one of these messages through a noiseless channel, it should be clear that unless one of the two messages has zero probability the encoding with minimum average length will assign to rain the codeword 0 and to sun 1 or viceversa.

Let us now imagine that we want to transmit the weather forecast through a noisy channel. For instance a channel that takes a binary symbol as input and outputs the same symbol with probability $1-p$ or flips it with probability p. In this new scenario, unless we change the encoding, the messages transmitted will be erroneous with probability p. The most obvious way of protecting against error is repeating the message. The repetition code of length 3 is $C = \{000, 111\}$.

Let us suppose that we use the repetition code to encode the weather forecast, for instance $C(\text{rain}) = 000, C(\text{sun}) = 111$, we send the codeword corresponding to rain through the noisy channel described above and the channel flips the last symbol. We would receive the word 001, and we could guess that the most likely codeword corresponding to the received vector is 000. In the following we will introduce the necessary definitions to understand quantitatively this example.

First, we introduce a distance function between vectors that will allow to justify the choice of decoding 001 to the codeword 000.

Definition 3.2.1 The Hamming distance between two vectors $x, y \in D^n$ is given by:

$$d(x,y) = |i : x_i \neq y_i, 1 \leq i \leq n| \tag{3.2}$$

An useful way of interpreting the Hamming distance is as the minimum number of positions that it is necessary to change in x to transform it to y.

■ **Example 3.1** The Hamming distance between vectors $x = (1,2,0,1,2)$ and $y = (2,1,0,1,2)$ is two because they differ in the first two entries. Alternatively, from the definition

$$d(x,y) = |i : x_i \neq y_i, 1 \leq i \leq n| = |1,2| = 2 \tag{3.3}$$

■

A function $d : \mathscr{X} \times \mathscr{X} \mapsto \mathbb{R}$ is a distance function if it verifies the following properties:

1. Identity: $d(x,y) = 0$ if and only if $x = y$
2. Symmetry: $d(y,x) = d(x,y)$
3. Triangle inequality: $d(x,y) \leq d(x,z) + d(z,y)$

Exercise 3.1 Show that the Hamming distance is a valid distance function ■

In the following whenever we refer to a distance, we will refer to the Hamming distance unless explicitly stated otherwise.

The decoding strategy that we described above is called nearest neighbor decoding or minimum distance decoding. A minimum distance decoder is a decoder that outputs the codeword closest in distance to the received vector y, or in the case that there are more than one it will choose from the set of closest codewords uniformly at random.

$$\hat{x} = \underset{x \in C}{\operatorname{argmin}} \, d(x,y) \tag{3.4}$$

■ **Example 3.2** A minimum distance decoder for the repetition code of length 3 will output 000 when it receives as input the word 001 since 000 has a smaller hamming distance to 001 than the other codeword in the code: 111. ■

Definition 3.2.2 A block code is a function $C : D^k \mapsto D^n$, where D is a finite set and $k \geq n$ are natural numbers.

Let us better understand the behavior of a minimum distance decoder by analyzing how it works quantitatively. Suppose that we send each of the symbols of a codeword $x = (x_1, \ldots, x_n)$ one by one through a channel that is potentially noisy. The output will be the vector $y = (y_1, \ldots, y_n)$ with $y_i = x_i + e_i$. We can understand the role of a decoder as that of guessing the added noise vector $e = (e_1, \ldots, e_n)$ as guessing e allows to undo the action of the channel.

We can now define the error rate of a word:

$$p_e(w) = \sum_{y \in D^n} p(y|c) p(\text{dec}(y) \neq x) \tag{3.5}$$

and the error rate of the transmission scheme:

$$p_e(C) = \sum_{w \in C} p(w) p_e(w) \tag{3.6}$$

Let us now study the effect of the repetition code in error rate on two important communications channels.

The first channel that we introduce is the binary symmetric channel.

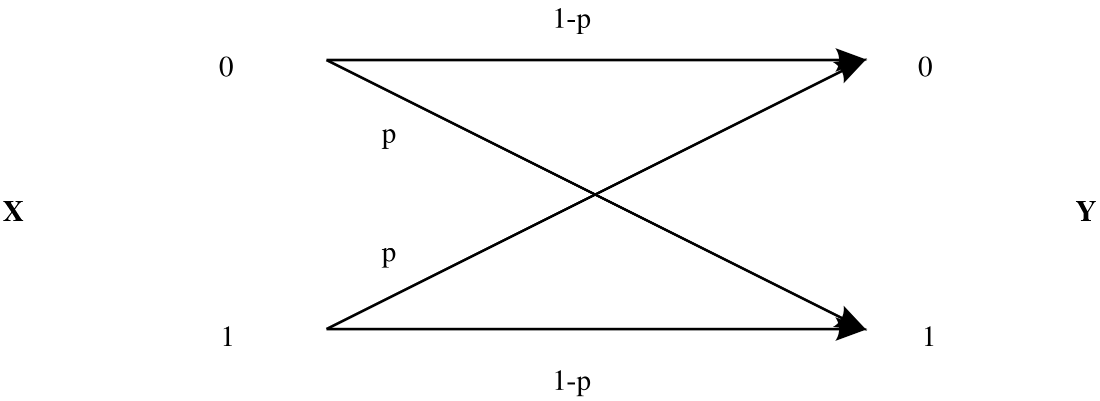


Figure 3.2: Binary Symmetric Channel.

Exercise 3.2 Consider $C = \{000, 111\}$, suppose that we send each symbol of the word 000 through a BSC(p).

- What is the probability of having no errors?
- What is the probability of having one error?

- What is the probability of having two errors?
- What is the probability of having three errors?
- What is the error probability with a minimum distance decoder?

■

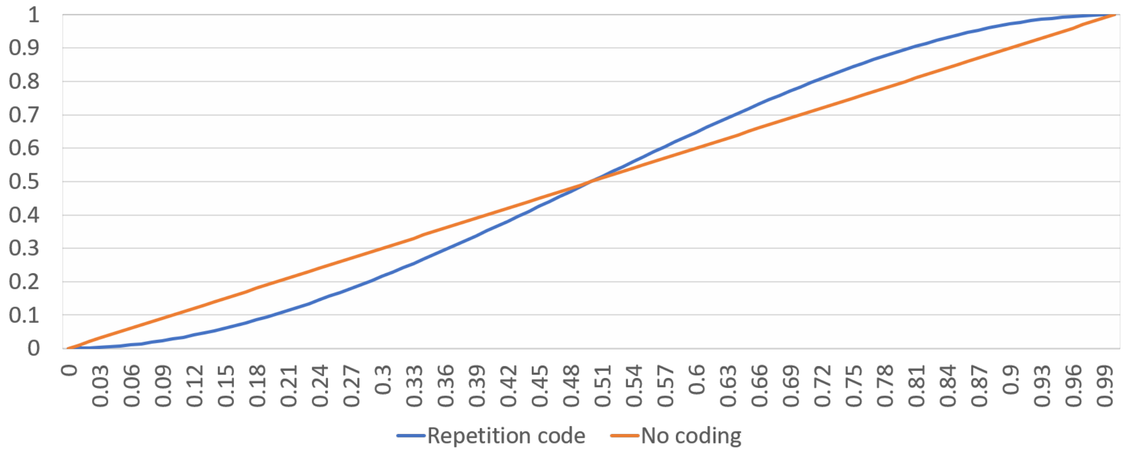


Figure 3.3: Decoding error vs crossover probability for the repetition code and for uncoded transmission.

Exercise 3.3 Consider $C = \{000, 111\}$, suppose that we send each symbol of the word 000 through a BEC(p).

- What is the probability of having no erasures?
- What is the probability of having one erasure?
- What is the probability of having two erasures?
- What is the probability of having three erasures?
- What is the error probability with a minimum distance decoder?

■

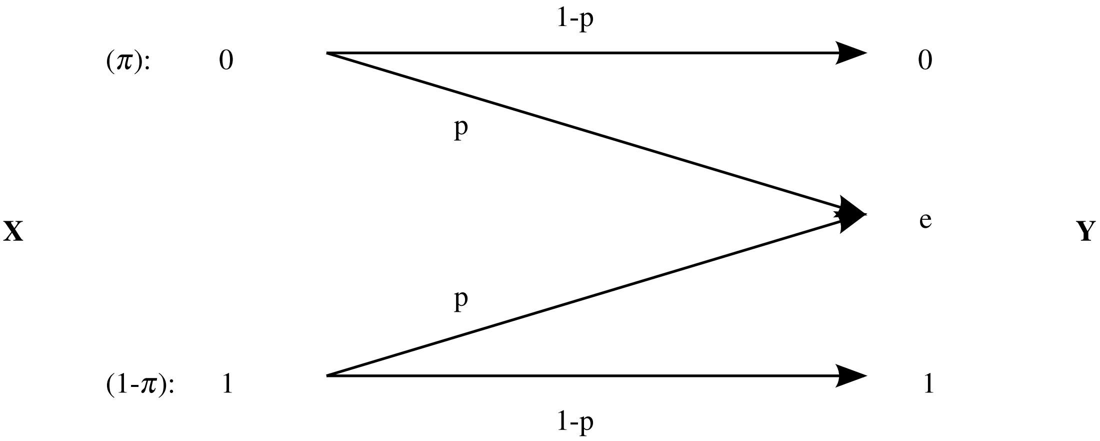


Figure 3.4: Binary Erasure Channel.

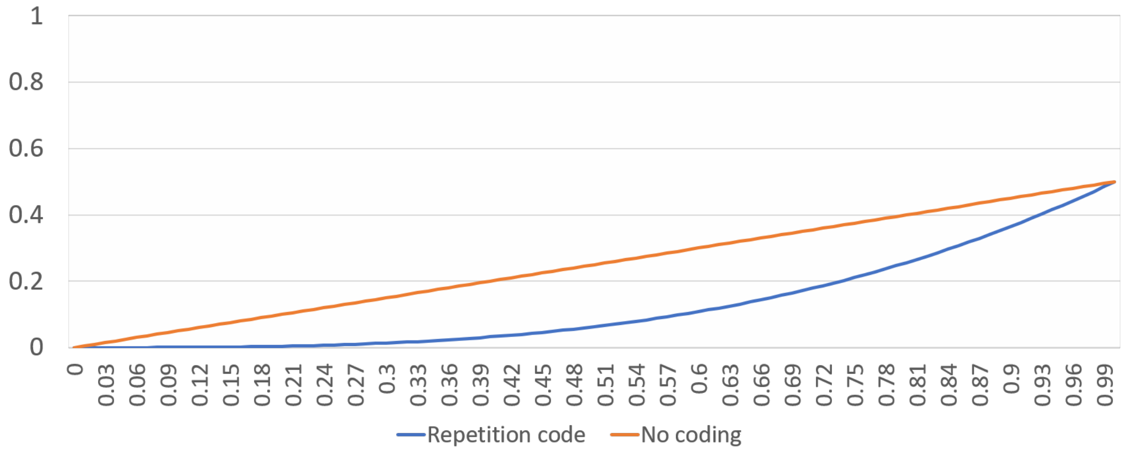


Figure 3.5: Decoding error vs erasure probability for the repetition code and for uncoded transmission.

Recall that the number of strings of length n with k ones is $\binom{n}{k}$.

Exercise 3.4 Show that a code C can detect all error patterns with s errors if and only if $d_{\min}(C) \geq s+1$ ■

Exercise 3.5 Show that a code C can correct all error patterns with t errors if and only if $d_{\min}(C) \geq 2t+1$ ■

Exercise 3.6 Show that a code C can detect all erasure patterns with e errors if and only if $d_{\min}(C) \geq e+1$ ■

Exercise 3.7 Given a code with minimum distance 12, find the maximum number of errors it can detect, the maximum number of errors it can correct and the maximum number of erasures it can correct. ■

Definition 3.2.3 An $[n,k,d]_q$ code is a code that encodes k symbols from a q-ary alphabet into n symbols of a q-ary alphabet and has minimum distance n.

■ **Example 3.3** The binary repetition code of length 3 is a $[3,1,3]_2$ code. ■

A typical coding theory problem is given two or three of the parameters in $[n,k,d]_q$ find the best code that matches those values. For instance:

- Given n,k,q find within the set of codes encoding k q-ary symbols into n, the code with the maximum minimum distance:

$$B_q(n,k) = \max \tag{3.7}$$

- Given n,k,q find within the set of codes encoding k q-ary symbols into n, the code with the maximum minimum distance:

$$A_q(n,k) = \max \tag{3.8}$$

- Given n, k, q find within the set of codes encoding k q-ary symbols into n, the code with the maximum minimum distance:

Exercise 3.8 Find $B_2(n,1)$

Exercise 3.9 Find $B_2(n,n)$

Definition 3.2.4 Two codes C and C' are equivalent if the set of codewords coincide up to a permutation in the position of the symbols, relabeling of the symbols or both.

Example 3.4 The codes $C = \{001, 110\}$ and $C' = \{100, 011\}$ are equivalent because the codewords coincide up to a permutation of the position of the symbols.

Example 3.5 The codes $C = \{001, 110\}$ and $C' = \{000, 111\}$ are equivalent because the codewords coincide up to a relabelling of the symbols.

The finite field F_2 is the set $\{0, 1\}$ together with the operations $+, \cdot$ defined as follows:

+	0	1
0	0	1
1	1	0

and

·	0	1
0	0	0
1	0	1

3.3 Bounds on codes

We will now introduce some notation about spheres on V_n that will allow us to bound the possible binary codes.

Definition 3.3.1 Given $x \in \{0,1\}^n$, and $r \in \mathbb{N}$ we define the sphere of radius r centered around x as the set of points with distance at most r: $S_r(x) = \{y : d(x,y) \leq r$.

Exercise 3.10 Find the set of points $S_1(x)$ with $x = (0, 1, 0)$.

Exercise 3.11 Let $x \in \{0,1\}^n$ and $r \in \mathbb{N}$, show that the number of elements in $S_r(x)$ is:

$$|S_r(x)| = \sum_{i=0}^{r} \binom{n}{i} \tag{3.9}$$

We now have the tools to prove to important bounds for the existence of codes. The first of these bounds is called the Hamming bound, from the mathematician that proved it, but also the sphere packing bound since it argues that a code can only correct all patterns of some weight t if it can fit as many spheres of radius t in V_n as the number of codewords.

Theorem 3.3.1 — Hamming bound. An $[n,k,d]$ code satisfies

$$\binom{2^k \sum_{i=0}^{t} n}{i \leq 2^n} \tag{3.10}$$

where $t = \lfloor \frac{d-1}{2} \rfloor$.

Proof. As shown in exercise **??**, a $S_t(x)$ sphere contains $\binom{\sum_{i=0}^{t} n}{i}$ words. For t to be the maximum weight of the error patterns the code can correct it needs to be possible to place a sphere of radius t around each of the 2^k codewords and these spheres need to be disjoint. This gives a total number of words of $\binom{2^k \sum_{i=0}^{t} n}{i \leq 2^n}$ which is only possible if this number is smaller than the total number of words in the space 2^n. ■

A code is called perfect if it attains the sphere packing bound with equality.

Now we will discuss a second bound, the Singleton bound, also based on dimensionality, but this time the argument stems from the distinguishability of codewords under erasure.

Theorem 3.3.2 — Singleton bound. An $[n,k,d]$-code satisfies $d \leq n-k+1$.

Proof. In a code with minimum distance d, if we erase $d-1$ positions of the code, all codewords need to be still different. However, the number of words of length $n-d+1$ is 2^{n-d+1} which can not be larger than the total number of words in the code 2^k, i.e. $2^{n-d+1} \leq 2^k$. The proof follows by taking the logarithm of both sides and solving for d. ■

A code that meets the Singleton bound with equality is called maximum distance separable code.

3.4 Refresher on linear algebra

A vector space that is going to be very useful in the following is the n-dimensional binary vector space that we will denote by V_n. This is the vector space of length n binary strings over $\mathbb{F}_2$.

Addition over V_n follows from addition in $\mathbb{F}_2$. That is, given $x, y \in \{0,1\}^n$.

$$x + y = (x_1 + y_1, \ldots, x_n + y_n) \tag{3.11}$$

where $x_i + y_i$ follows the rules from (**??**). Similarly scalar multiplication in V_n follows from the multiplication rules in $\mathbb{F}_2$, given $s \in \{0,1\}$ and $x \in \{0,1\}^n$:

$$s\dot{x} = (s \cdot x_1, \ldots, s \cdot x_n) \tag{3.12}$$

where $s \cdot x_i$ follows the rules from (**??**).

Definition 3.4.1 The Hamming weight $w : \{0,1\}^n \mapsto \mathbb{N}$ of a binary string is given by its number of ones. Given $x \in \{0,1\}^n$:

$$w(x) = \sum_{i=1}^{n} x_i \tag{3.13}$$

Exercise 3.12 Show that given $x, y \in \{0,1\}^n$, $d(x,y) = w(x+y)$. ■

In the following we state several important properties and definitions of vector spaces that will be of use in coding theory. Some of these, we state only for V_n for simplicity. If these notions are unfamiliar or not completely understood, please review your text on the matter.

Definition 3.4.2 U is a subspace of V if $U \subseteq V$ and U is a vector space.

Example 3.6 $\{000, 111\}$ is subspace of V_3.

Definition 3.4.3 A linear combination of the vectors $v^1, v^2, \dots, v^n \in \{0,1\}^n$ is a vector $s_1 \cdot v^1 + s_2 \cdot v^2 + \ldots + s_n \cdot v^n$ where $s_1, s_2, \dots, s_n \in \mathbb{F}_2$.

Example 3.7 $0 \cdot (0,0,1) + 1 \cdot (1,1,0) = (1,1,0)$ is a linear combination of the vectors $(0,0,1)$ and $(1,1,0)$.

Definition 3.4.4 The set of linear combinations of a set of vectors is called its span.

Exercise 3.13 Show that the span of a set of vectors is a vector space.

Definition 3.4.5 A set of vectors $v^1, v^2, \dots, v^k \in \{0,1\}^n$ is linearly dependent if there exist $s_1, s_2, \dots, s_k \in \{0,1\}^k$ different from $0, \ldots 0$ such that $s_1 \cdot v^1 + s_2 \cdot v^2 + \ldots + s_n \cdot v^n = 0$.

Definition 3.4.6 A set of vectors $v^1, v^2, \dots, v^k \in \{0,1\}^n$ is linearly independent if the $s_1, s_2, \dots, s_k \in \{0,1\}^k$ different from $0, \ldots 0$ such that $s_1 \cdot v^1 + s_2 \cdot v^2 + \ldots + s_n \cdot v^n = 0$.

3.5 Binary linear codes

A binary linear code of length n is a subspace of V_n.

Exercise 3.14 Show that the repetition code of length 3 is a subspace of V_3.

Exercise 3.15 If C is a binary linear code then:

$$\min_{x,y \in C} d(x,y) = \min_{x \in C \setminus \{0\}} w(x) \tag{3.14}$$

Definition 3.5.1 Given a $[n,k,d]$ linear code, a matrix with k rows and n columns where each row is an element of a basis of the code is called a generator matrix.

Example 3.8 A generator matrix for the length three repetition code is given by:

$$\big([ccc]1 \quad 1 \quad 1\big) \tag{3.15}$$

Exercise 3.16 Two generator matrices G_1, G_2 with coefficients in $\{0,1\}$ generate two equivalent binary codes if it is possible to transform G_1 into G_2 by a series of row permutations, column permutations and additions of one row into another.

Lemma 3.5.1 Any generator matrix G of an $[n,k,d]$ binary linear code can transformed to a generator matrix of the form $G' = (I_k | A)$ where I_k is the d-dimensional identity matrix and A is an arbitrary $n - k \times k$ matrix. This matrix form is called *standard form*.

Exercise 3.17 Take the following generator matrix to standard form:

$$G = \begin{pmatrix} [ccccc]0 & 1 & 0 & 0 & 1 \\ 1 & 1 & 0 & 0 & \\ 0 & 0 & 1 & 0 & 0 \end{pmatrix} \tag{3.16}$$

We define C, the capacity of a channel, as the maximum mutual information over all possible input distributions:

$$C = \max_{p(x)} I(X;Y) \tag{3.17}$$

3.6 Converse theorem for noisy channel coding

The converse statement follows from Fano's inequality [**Fano_61**]. The intuition behind this part is that if we think of an encoding that achieves a vanishing error probability, then necessarily $R < I(X;Y)$ [3].

The past sections have been devoted to schemes that achieve a compression rate slightly larger than the entropy. We have called these schemes optimal. We also discussed that codes that have no error have an average length bounded from below by the entropy. In this section we will investigate what happens if we can tolerate some error. Surprisingly, we can not do much better than entropy.

Let us first introduce Fano's inequality as it will be the main tool in our argument. Suppose that we want to guess the value of random variable X and we have access to random variable Y, let us call $\hat{X} = g(Y)$ the random variable that characterizes the guess. We denote by E the random variable that takes value 1 if $X = \hat{X}$ and 0 when $X \neq \hat{X}$. we can now state Fano's inequality:

Theorem 3.6.1 — Fano's inequality. Let X,Y be two random variables and $\hat{X} = g(Y)$ for some function $g : \mathscr{Y} \mapsto \mathscr{X}$ and let $E = I(X = \hat{X})$. Then:

$$H(X|Y) \leq H(E) + p_E(E=1)\log(|\mathscr{X} - 1|) \tag{3.18}$$

Proof. Let us first expand $H(X,E|Y)$ in two different ways applying the chain rule. First:

$$\begin{aligned} H(E,X|Y) &= H(E|Y) + H(X|E,Y) && (3.19)\\ &\leq H(E) + H(X|E,Y) && (3.20)\\ &= H(E) + p_E(0)H(X|E=0,Y) + p_E(1)H(X|E=1,Y) && (3.21)\\ &= H(E) + p_E(1)H(X|E=1,Y) && (3.22)\\ &\leq H(E) + p_E(1)\log(|\mathscr{X}|-1) && (3.23) \end{aligned}$$

where the first equality follows from the chain rule, the first inequality since conditioning reduces entropy, the second equality follows from application of the definition of conditional entropy and the second one since in the event that the error is zero, there is no uncertainty on the value of X, the last inequality follows since entropy can never be larger than the logarithm of the number of outcomes.

Let us note that $H(E,X|Y)$ can also be expanded as:

$$\begin{aligned} H(E,X|Y) &= H(X|Y) + H(E|X,Y) && (3.24)\\ &= H(X|Y) && (3.25) \end{aligned}$$

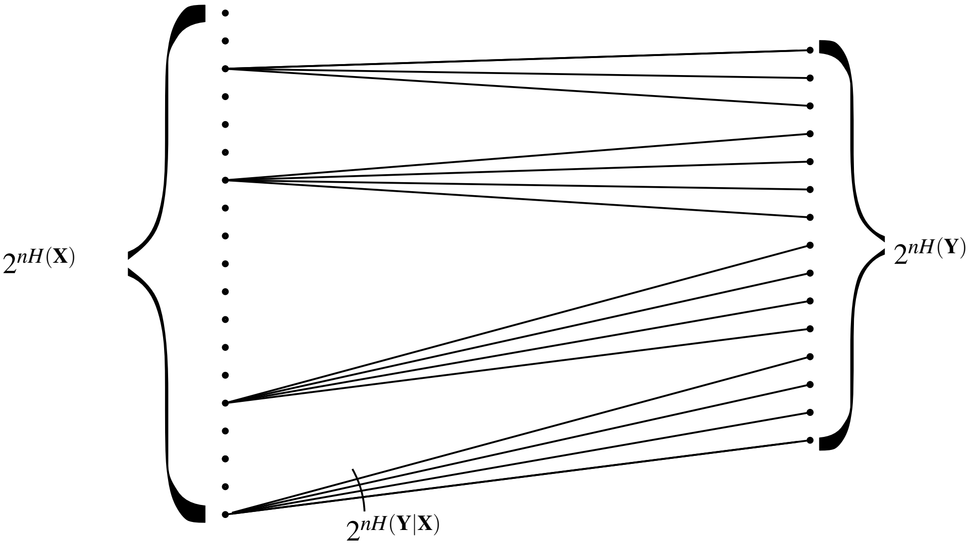


Figure 3.6: Graphical representation of the input and output typical sequences. A good encoding chooses as codewords a subset of the input typical sequences that produces disjoint sets of output typical sequences.

where the first equality follows from the chain rule and the second because if both X and Y are known, then E is also known.

The proof ends by joining (**??**) with (**??**) ■

Exercise 3.18 In (**??**), we have that $H(X|E=1,Y) \leq \log(|\mathscr{X}|-1)$. Why can we remove one outcome from the alphabet of X? ■

3.7 Hamming codes

3.8 Sketch of the noisy channel theorem

The capacity of a channel specifies the maximum rate at which a source can be reliably sent through a channel. On the other hand, no source with a rate over the capacity of the channel can be sent with a vanishing error probability.

A sketch of the proof would be as follows. Encoder and decoder share a code-book of 2^{nR} codewords chosen within the $2^{nH(X)}$ typical sequences [**Massey_77**]. The encoder sends a codeword x drawn with uniform probability. The decoder outputs a word $\hat{x}$ jointly typical with the received word y. It declares an error if x, y are not jointly typical and a decoding error can occur if there exists $x' \neq x$ jointly typical with y. We know by Eq. **??** that the probability of non-joint typicality for long enough n can be made as small as desired.

The intuition behind the achievability proof is simple. The decoder has access to two sets: the set of sequences jointly typical with y, and the set of codewords. If the intersection is to be a single word, every codeword has to be jointly typical with a disjoint set of typical output words.

Approximately, every codeword is jointly typical with $2^{nH(Y|X)}$ words. Then the number of jointly typical output words with input codewords is upper bounded by $2^{nR+nH(Y|X)}$, where R is the coding rate. This number should be much smaller than the total number of typical sequences $2^{nH(Y)}$:

$$2^{nR+nH(Y|X)} < 2^{nH(Y)}$$

which operating returns the expected result:

$$R < I(X;Y)$$

In conclusion, as long as the coding rate is below the mutual information between input and output for n long enough we can construct a code that allows the decoder to distinguish between codewords with a vanishing probability of error.

3.9 The capacity of some basic channels

In the BSC the binary elements or bits are either perfectly transmitted with probability $1-p$ or flipped with probability p.

Let us first find the mutual information between the input X and the output Y [3]::

$$\begin{aligned}
I(X;Y) &= H(Y) - H(Y|X) && (3.26)\\
&= H(Y) - \sum_x p(x)H(Y|x) && (3.27)\\
&= H(Y) - \sum_x p(x)H(p,1-p) && (3.28)\\
&= H(Y) - H(p,1-p)\sum_x p(x) && (3.29)\\
&\leq 1 - H(p,1-p) && (3.30)
\end{aligned}$$

We obtain the capacity by finding the maximum of the mutual information for all possible input distributions. It can be easily verified that the the uniform distribution reaches the upper bound in Eq. 3.30 and the capacity of the BSC is one minus the binary entropy of p.

The BEC was introduced by Elias in his famous paper "Coding for Two Noisy Channels" [**Elias_55**]. The BEC has two input elements while the output alphabet is composed of three elements: 0, 1, and e, which stands for an erasure in the channel. In this channel the bits are either correctly transmitted with probability $1-p$, or are erased with probability p.

We can first find $H(X|Y)$:

$$\begin{aligned}
H(X|Y) &= \pi(1-p)H(X|Y=0)\\
&\quad +(\pi p + (1-\pi)p)H(X|Y=e)\\
&\quad +(1-\pi)(1-p)H(X|Y=1) && (3.31)\\
&= p && (3.32)
\end{aligned}$$

where $p(X=0)=\pi$.The second equality holds from $H(X|Y=1)=H(X|Y=0)=0$ and $H(X|Y=e)=1$. We can now plug Eq. 3.31 in Eq. 1.30 and bound from above the mutual information:

$$\begin{aligned}
I(X;Y) &= H(X) - H(X|Y) && (3.33)\\
&= H(\pi, 1-\pi) - p && (3.34)\\
&\leq 1 - p && (3.35)
\end{aligned}$$

equality in Eq. 3.35 is achieved again by the uniform distribution. That is, for $\pi = \frac{1}{2}$.

It might seem that the capacity of a BSC that flips bits with probability p is greater than the capacity of a BEC that erases bits with probability p. Fig. 3.7 shows that it is the opposite situation. On the range $p \in (0, 0.5)$, the capacity of the BEC is greater than the capacity of the BSC. Bits on the BEC are either perfectly known or perfectly unknown, however, it is not possible to distinguished flipped bits from correct bits in the BSC.

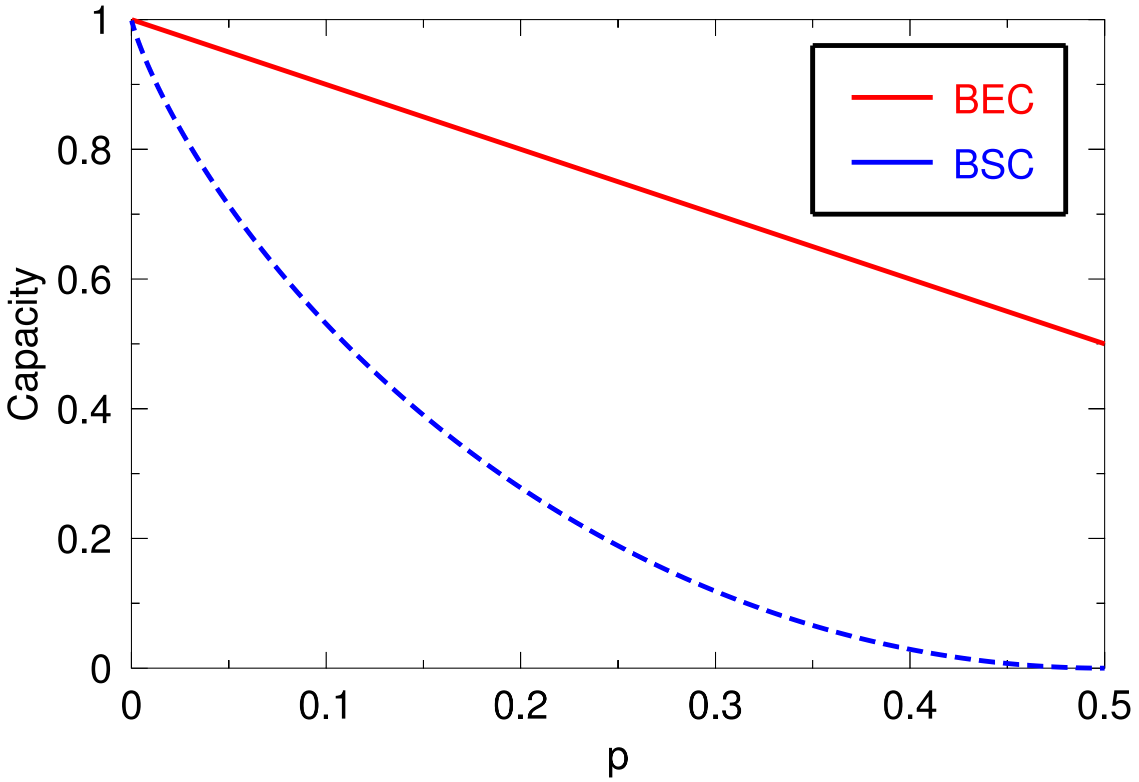


Figure 3.7: The capacity of the BEC and BSC.

3.10 Exercises

3.11 Further reading

Chapter 7 in [3].

Bibliography

[1] J. Aczel and Z. Daroczy. *On measures of information and their characterizations*. Academic Press, 1975 (cited on page 19).

[2] J. Aczel, B. Forte, and C. T. Ng. “Why the Shannon and Hartley Entropies Are Natural”. In: *Advances in Applied Probability* 6.1 (1974), pages 131–146. ISSN: 00018678 (cited on page 19).

[3] T. M. Cover and J. A. Thomas. *Elements of Information Theory*. Wiley-Interscience, Aug. 1991 (cited on pages 19, 23, 30, 39, 41, 42).

[4] I. Csiszár. “Axiomatic Characterizations of Information Measures”. In: *Entropy* 10 (2008), pages 261–273 (cited on page 19).

[5] A. Feinstein. *Foundations of Information Theory*. McGraw-Hill, 1958 (cited on page 19).

[6] David A Huffman. “A method for the construction of minimum-redundancy codes”. In: *Proceedings of the IRE* 40.9 (1952), pages 1098–1101 (cited on page 26).

[7] J. Karush. “A simple proof of an inequality of McMillan (Corresp.)” In: *IRE Transactions on Information Theory* 7.2 (Apr. 1961), page 118 (cited on page 23).

[8] David JC MacKay. *Information theory, inference and learning algorithms*. Cambridge university press, 2003 (cited on page 19).

[9] B. McMillan. “Two inequalities implied by unique decipherability”. In: *IRE Transactions on Information Theory* 2 (Dec. 1956), pages 115–116 (cited on page 23).

[10] David Salomon and Giovanni Motta. *Handbook of data compression*. Springer Science & Business Media, 2010 (cited on pages 23, 30).

[11] August Albert Sardinas and George W Patterson. “A necessary and sufficient condition for unique decomposition of coded messages”. In: *Proceedings of the Institute of Radio Engineers*. Volume 41. 3. 1953, pages 425–425 (cited on page 22).

[12] C. E. Shannon. “A mathematical theory of Communication”. In: *The Bell system technical journal* 27 (July 1948), pages 379–423 (cited on pages 19, 30, 31).

[13] *Twitter and Shannon*. `https://what-if.xkcd.com/34/`. Accessed: 2019-08-03 (cited on page 30).

Index