Chapter 20 Information Theory

In this chapter we introduce the theoretical concepts behind the security of a cryptosystem. The basic question is the following: If Eve observes a piece of ciphertext, does she gain any new information about the encryption key that she did not already have? To address this issue, we need a mathematical definition of information. This involves probability and the use of a very important measure called entropy.

Many of the ideas in this chapter originated with Claude Shannon in the 1940s.

Before we start, let's consider an example. Roll a standard six-sided die. Let A be the event that the number of dots is odd, and let B be the event that the number of dots is at least 3. If someone tells you that the roll belongs to the event $A \cap B$, then you know that there are only two possibilities for what the roll is. In this sense, $A \cap B$ tells you more about the value of the roll than just the event A, or just the event B. In this sense, the information contained in the event $A \cap B$ is larger than the information just in A or just in B.

The idea of information is closely linked with the idea of uncertainty. Going back to the example of the die, if you are told that the event $A \cap B$ happened, you become less uncertain about what the value of the roll was than if you are simply told that event A occurred. Thus the information increased while the uncertainty decreased. Entropy provides a measure of the increase in information or the decrease in uncertainty provided by the outcome of an experiment.

20.1 Probability Review

In this section we briefly introduce the concepts from probability needed for what follows. An understanding of probability and the various identities that arise is essential for the development of entropy.

Consider an experiment X with possible outcomes in a finite set X. For example, X could be flipping a coin and $X = \{\text{heads}, \text{tails}\}$. We assume each outcome is assigned a probability. In the present example, $p(X = \text{heads}) = 1/2$ and $p(X = \text{tails}) = 1/2$. Often, the outcome X of an experiment is called a **random variable**.

In general, for each $x \in X$, denote the probability that $X = x$ by

$$p_X(x) = p_x = p(X = x).$$

Note that $\sum_{x \in X} p_x = 1$. If $A \subseteq X$, let

$$p(A) = \sum_{x \in A} p_x,$$

which is the probability that X takes a value in A.

Often one performs an experiment where one is measuring several different events. These events may or may not be related, but they may be lumped together to form a new random event. For example, if we have two random events X and Y with possible outcomes X and Y, respectively, then we may create a new random event $Z = (X, Y)$ that groups the two events together. In this case, the new event Z has a set of possible outcomes $Z = X \times Y$, and Z is sometimes called a joint random variable.

Example

Draw a card from a standard deck. Let X be the suit of the card, so $X = \{\text{clubs, diamonds, hearts, spades}\}$. Let Y be the value of the card, so $Y = \{\text{two, three, } \ldots \text{, ace}\}$. Then Z gives the 52 possibilities for the card. Note that if $x \in X$ and $y \in Y$, then $p((X, Y) = (x, y)) = p(X = x, Y = y)$ is simply the probability that the card drawn has suit x and value y. Since all cards are equally probable, this probability is 1/52, which is the probability that $X = x$ (namely 1/4) times the probability that $Y = y$ (namely 1/13). As we discuss later, this means X and Y are independent.

Example

Roll a die. Suppose we are interested in two things: whether the number of dots is odd and whether the number is at least 2. Let $X = 0$ if the number of dots is even and $X = 1$ if the number of dots is odd. Let $Y = 0$ if the number of dots is less than 2 and $Y = 1$ if the number of dots is at least 2. Then $Z = (X, Y)$ gives us the results of both experiments together. Note that the probability that the number of dots is odd and less than 2 is $p(Z = (1, 0)) = 1/6$. This is not equal to $p(X = 0) \cdot p(Y = 0)$, which is $(1/2)(1/6) = 1/12$. This means that X and Y are not independent. As we'll see, this is closely related to the fact that knowing X gives us information about Y.

We denote

$$p_{X,Y}(x, y) = p(X = x, Y = y).$$

Note that we can recover the probability that $X = x$ as

$$p_X(x) = \sum_{y \in Y} p_{X,Y}(x, y).$$

We say that two random events X and Y are **independent** if

$$p_{X,Y}(x, y) = p_X(x)p_Y(y)$$

for all $x \in X$ and all $y \in Y$. In the preceding example, the suit of a card and the value of the card were independent.

We are also interested in the probabilities for Y given that $X = x$ has occurred. If $p_X(x) > 0$, define the **conditional probability** of $Y = y$ given that $X = x$ to be

$$p_Y(y|x) = \frac{p_{X,Y}(x, y)}{p_X(x)}.$$

One way to think of this is that we have restricted to the set where $X = x$. This has total probability $p_X(x) = \sum_y p_{X,Y}(x, y)$. The fraction of this sum that comes from $Y = y$ is $p_Y(y|x)$.

Note that X and Y are independent if and only if

$$p_Y(y|x) = p_Y(y)$$

for all x, y. In other words, the probability of y is unaffected by what happens with X.

There is a nice way to go from the conditional probability of Y given X to the conditional probability of X given Y .

Bayes’s Theorem

If $p_X(x) > 0$ and $p_Y(y) > 0$, then

$$p_X(x|y) = \frac{p_X(x)p_Y(y|x)}{p_Y(y)}.$$

The proof consists of simply writing the conditional probabilities in terms of their definitions.

20.2 Entropy

Roll a six-sided die and a ten-sided die. Which experiment has more uncertainty? If you make a guess at the outcome of each roll, you are more likely to be wrong with the ten-sided die than with the six-sided die. Therefore, the ten-sided die has more uncertainty. Similarly, compare a fair coin toss in which heads and tails are equally likely with a coin toss in which heads occur 90% of the time. Which has more uncertainty? The fair coin toss does, again because there is more randomness in its possibilities.

In our definition of uncertainty, we want to make sure that two random variables X and Y that have same probability distribution have the same uncertainty. In order to do this, the measure of uncertainty must be a function only of the probability distributions and not of the names chosen for the outcomes.

We require the measure of uncertainty to satisfy the following properties:

1. To each set of nonnegative numbers $p_1, \ldots, p_n$ with $p_1 + \cdots + p_n = 1$, the uncertainty is given by a number $H(p_1, \ldots, p_n)$.

2. H should be a continuous function of the probability distribution, so a small change in the probability distribution should not drastically change the uncertainty.

3. $H\left(\frac{1}{n}, \ldots, \frac{1}{n}\right) \leq H\left(\frac{1}{n+1}, \ldots, \frac{1}{n+1}\right)$ for all $n > 0$. In other words, in situations where all outcomes are equally likely, the uncertainty increases when there are more possible outcomes.

4. If $0 < q < 1$, then

$$H(p_1, \ldots, qp_j, (1-q)p_j, \ldots, p_n) = H(p_1, \ldots, p_j, \ldots, p_n) + p_j H(q, 1-q).$$

What this means is that if the jth outcome is broken into two suboutcomes, with probabilities qp_j and $(1-q)p_j$, then the total uncertainty is increased by the uncertainty caused by the choice between the two suboutcomes, multiplied by the probability p_j that we are in this case to begin with. For example, if we roll a six-sided die, we can record two outcomes: *even* and *odd*. This has uncertainty $H\left(\frac{1}{2}, \frac{1}{2}\right)$. Now suppose we break the outcome *even* into the suboutcomes 2 and $\{4, 6\}$. Then we have three possible outcomes: 2, $\{4, 6\}$, and *odd*. We have

$$H\left(\frac{1}{6}, \frac{1}{3}, \frac{1}{2}\right) = H\left(\frac{1}{2}, \frac{1}{2}\right) + \frac{1}{2} H\left(\frac{2}{3}, \frac{1}{3}\right).$$

The first term is the uncertainty caused by *even* versus *odd*. The second term is the uncertainty added by splitting *even* into two suboutcomes.

Starting from these basic assumptions, Shannon [Shannon2] showed the following:

Theorem

Let $H(X)$ be a function satisfying properties (1)–(4). In other words, for each random variable X with outcomes $X = \{x_1, \ldots, x_n\}$ having probabilities $p_1, \ldots, p_n$, the function H assigns a number $H(X)$ subject to the conditions (1)–(4). Then H must be of the form

$$H\left(p_1, \cdots, p_n\right) = -\lambda \sum_k p_k \log_2 p_k$$

where λ is a nonnegative constant and where the sum is taken over those k such that $p_k > 0$.

Because of the theorem, we define the **entropy** of the variable X to be

$$H(X) = -\sum_{x \in X} p(x) \log_2 p(x).$$

The entropy $H(X)$ is a measure of the uncertainty in the outcome of X. Note that since $\log_2 p(x) \leq 0$, we have $H(X) \geq 0$, so there is no such thing as negative uncertainty.

The observant reader might notice that there are problems when we have elements $x \in X$ that have probability 0. In this case we define $0 \log_2 0 = 0$, which is justified by looking at the limit of $x \log_2 x$ as $x \to 0$. It is typical convention that the logarithm is taken base 2, in which case entropy is measured in bits. The entropy of X may also be interpreted as the expected value of $-\log_2 p(X)$ (recall that $E[g(X)] = \sum_x g(x)p(x)$).

We now look at some examples.

Example

Consider a fair coin toss. There are two outcomes, each with probability 1/2. The entropy of this random event is

$$-\left(\frac{1}{2}\log_2\frac{1}{2} + \frac{1}{2}\log_2\frac{1}{2}\right) = 1.$$

This means that the result of the coin flip gives us 1 bit of information, or that the uncertainty in the outcome of the coin flip is 1 bit.

Example

Consider a nonfair coin toss X with probability p of getting heads and probability $1 - p$ of getting tails (where $0 < p < 1$). The entropy of this event is

$$H(X) = -p\log_2 p - (1-p)\log_2 (1-p).$$

If one considers $H(X)$ as a function of p, one sees that the entropy is a maximum when $p = \frac{1}{2}$. (For a more general statement, see Exercise 14.)

Example

Consider an n-sided fair die. There are n outcomes, each with probability $1/n$. The entropy is

$$-\frac{1}{n}\log_2(1/n) - \ldots - \frac{1}{n}\log_2(1/n) = \log_2(n).$$

There is a relationship between entropy and the number of yes-no questions needed to determine accurately the outcome of a random event. If one considers a totally nonfair coin toss where $p(1) = 1$, then $H(X) = 0$. This result can be interpreted as not requiring any questions to determine what the value of the event was. If someone rolls a four-sided die, then it takes two yes-no questions to find out the outcome. For example, is the number less than 3? Is the number odd?

A slightly more subtle example is obtained by flipping two coins. Let X be the number of heads, so the possible outcomes are $\{0, 1, 2\}$. The probabilities are $1/4,\ 1/2,\ 1/4$ and the entropy is

$$-\frac{1}{4}\log_2(1/4) - \frac{1}{2}\log_2(1/2) - \frac{1}{4}\log_2(1/4) = \frac{3}{2}.$$

Note that we can average 3/2 questions to determine the outcome. For example, the first question could be “Is there exactly one head?” Half of the time, this will suffice to determine the outcome. The other half of the time a second question is needed, for example, “Are there two heads?” So the average number of questions equals the entropy.

Another way of looking at $H(X)$ is that it measures the number of bits of information that we obtain when we are given the outcome of X. For example, suppose the outcome of X is a random 4-bit number, where each possibility has probability 1/16. As computed previously, the entropy is $H(X) = 4$, which says we have received four bits of information when we are told the value of X.

In a similar vein, entropy relates to the minimal amount of bits necessary to represent an event on a computer (which is a binary device). See Section 20.3. There is no sense recording events whose outcomes can be predicted with 100% certainty; it would be a waste of space. In storing information, one wants to code just the uncertain parts because that is where the real information is.

If we have two random variables X and Y, the joint entropy $H(X, Y)$ is defined as

$$H(X, Y) = -\sum_{x \in X} \sum_{y \in Y} p_{X,Y}(x, y) \log_2 p_{X,Y}(x, y).$$

This is just the entropy of the joint random variable $Z = (X, Y)$ discussed in Section 20.1.

In a cryptosystem, we might want to know the uncertainty in a key, given knowledge of the ciphertext. This leads us to the concept of **conditional entropy**, which is the amount of uncertainty in Y, given X. It is defined to be

$$\begin{aligned} H(Y|X) &= \sum_x p_X(x) H(Y|X=x) \\ &= -\sum_x p_X(x) \left(\sum_y p_Y(y|x) \log_2 p_Y(y|x) \right) \\ &= -\sum_x \sum_y p_{X,Y}(x, y) \log_2 p_Y(y|x). \end{aligned}$$

The last equality follows from the relationship $p_{X,Y}(x, y) = p_Y(y|x) p_X(x)$. The quantity $H(Y|X = x)$ is the uncertainty in Y given the information that $X = x$. It is defined in terms of conditional probabilities by the expression in parentheses on the second line. We calculate $H(Y|X)$ by forming a weighted sum of these uncertainties to get the total uncertainty in Y given that we know the value of X.

Remark

The preceding definition of conditional entropy uses the weighted average, over the various $x \in X$, of the entropy of Y given $X = x$. Note that $H(Y|X) \neq -\sum_{x,\, y} p_Y(y|x) \log_2 (p_Y(y|x))$. This sum does not have properties that information or uncertainty should have. For example, if X and Y are independent, then this definition would imply that the uncertainty of Y given X is greater than the uncertainty of Y (see Exercise 15). This clearly should not be the case.

We now derive an important tool, the chain rule for entropies. It will be useful in Section 20.4.

Theorem

(Chain Rule). $H(X, Y) = H(X) + H(Y|X)$.

Proof

$$\begin{aligned} H(X,\ Y\ ) &= -\sum_{x\in X}\sum_{y\in Y} p_{X,\ Y}\,(x,\ y) \log_2 p_{X,\ Y}\,(x,\ y) \\ &= -\sum_{x\in X}\sum_{y\in Y} p_{X,\ Y}\,(x,\ y) \log_2 p_X\,(x)\; py\,(y|x) \\ &= -\sum_{x\in X}\sum_{y\in Y} p_{X,\ Y}\,(x,\ y) \log_2 p_X\,(x) - \sum_{x\in X}\sum_{y\in Y} p_{X,\ Y}\,(x,\ y) \log_2 p_Y\,(y|x) \\ &= \left(\sum_x \log_2 p_X(x) \sum_Y p_{X,\ Y}\,(x,\ y)\right) + H(Y|X)) \\ &= \sum_x p_X(x) \log_2 p_X(x) + H(Y|X) \quad (\text{since } \sum_y p_{X,Y}\,(x,\ y)\ =\ p_X(x)) \\ &= H(X)\ +\ H(Y|X). \end{aligned}$$

What does the chain rule tell us? It says that the uncertainty of the joint event (X, Y) is equal to the uncertainty of event X + uncertainty of event Y given that event X has happened.

We now state three more results about entropy.

Theorem

1. $H(X) \leq \log_2 |X|$, where $|X|$ denotes the number of elements in X. We have equality if and only if all elements of X are equally likely.
2. $H(X, Y) \leq H(X) + H(Y)$.
3. (Conditioning reduces entropy) $H(Y|X) \leq H(Y)$, with equality if and only if X and Y are independent.

The first result states that you are most uncertain when the probability distribution is uniform. Referring back to the example of the nonfair coin flip, the entropy was maximum for $p = \frac{1}{2}$. This extends to events with more possible outcomes. For a proof of (1), see [Welsh, p. 5].

The second result says that the information contained in the pair (X, Y) is at most the information contained in X plus the information contained in Y. The reason for the inequality is that possibly the information supplied by X and Y overlap (which is when X and Y are not independent). For a proof of (2), see [Stinson].

The third result is one of the most important results in information theory. Its interpretation is very simple. It says that the uncertainty one has in a random event Y given that event X occurred is less than the uncertainty in event Y alone. That is, X can only tell you information about event Y; it can't make you any more uncertain about Y.

The third result is an easy corollary of the second plus the chain rule:

$$H(X) + H(Y|X) = H(X, Y) \leq H(X) + H(Y).$$

20.3 Huffman Codes

Information theory originated in the late 1940s from the seminal papers by Claude Shannon. One of the primary motivations behind Shannon’s mathematical theory of information was the problem of finding a more compact way of representing data. In short, he was concerned with the problem of compression. In this section we briefly touch on the relationship between entropy and compression and introduce Huffman codes as a method for more succinctly representing data.

For more on how to compress data, see [Cover-Thomas] or [Nelson-Gailly].

Example

Suppose we have an alphabet with four letters a, b, c, d, and suppose these letters appear in a text with frequencies as follows.

a	b	c	d
$.5$	$.3$	$.1$	$.1$

We could represent a as the binary string 00, b as 01, c as 10, and d as 11. This means that the message would average two bits per letter. However, suppose we represent a as 1, b as 01, c as 001, and d as 000. Then the average number of bits per letter is

$$(1)(.5) + (2)(.3) + (3)(.1) + (3)(.1) = 1.7$$

(the number of bits for a times the frequency of a, plus the number of bits for b times the frequency of b, etc.). This encoding of the letters is therefore more efficient.

In general, we have a random variable with outputs in a set X. We want to represent the outputs in binary in an efficient way; namely, the average number of bits per output should be as small as possible.

An early example of such a procedure is Morse code, which represents letters as sequences of dots and dashes and was developed to send messages by telegraph. Morse asked printers which letters were used most, and made the more frequent letters have smaller representations. For example, e is represented as $\cdot$ and t as $-$. But x is $- \ \cdot \ \cdot -$ and z is $- \ - \ \cdot \ \cdot$.

A more recent method was developed by Huffman. The idea is to list all the outputs and their probabilities. The smallest two are assigned 1 and 0 and then combined to form an output with a larger probability. The same procedure is then applied to the new list, assigning 1 and 0 to the two smallest, then combining them to form a new list. This procedure is continued until there is only one output remaining. The binary strings are then obtained by reading backward through the procedure, recording the bits that have been assigned to a given output and to combinations containing it. This is best explained by an example.

Suppose we have outputs a, b, c, d with probabilities 0.5, 0.3, 0.1, 0.1, as in the preceding example. The diagram in Figure 20.1 gives the procedure.

Figure 20.1 An Example of Huffman Encoding

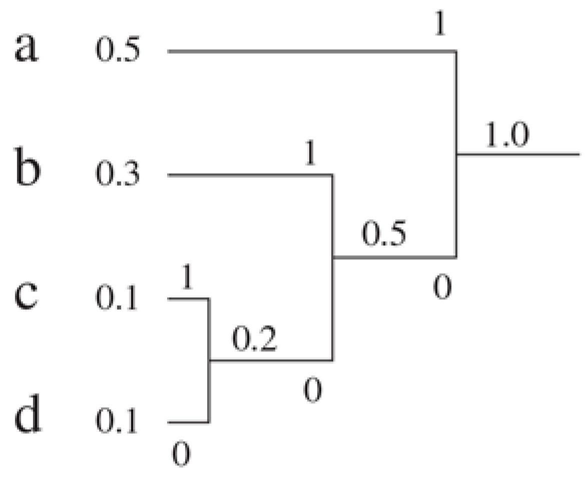

Figure 20.1 Full Alternative Text

Note that when there were two choices for the lowest, we made a random choice for which one received 0 and which one received 1. Tracing backward through the table, we see that a only received a 1, b received 01, c received 001, and d received 000. These are exactly the assignments made previously that gave a low number of bits per letter.

A useful feature of Huffman encoding is that it is possible to read a message one letter at a time. For example, the string 011000 can only be read as bad; moreover, as soon as we have read the first two bits 01, we know that the first letter is b.

Suppose instead that we wrote the bits assigned to letters in reverse order, so b is 10 and c is 001. Then the message 101000 cannot be determined until all bits have been read, since it potentially could start with bb or ba.

Even worse, suppose we had assigned 0 to a instead of 1. Then the messages aaa and d would be the same. It is possible to show that Huffman encoding avoids these two problems.

The average number of bits per output is closely related to the entropy.

Theorem

Let L be the average number of bits per output for Huffman encoding for the random variable X. Then

$$H(X) \leq L < H(X) + 1.$$

This result agrees with the interpretation that the entropy measures how many bits of information are contained in the output of X. We omit the proof. In our example, the entropy is

$$H(X) = -\big(.5\,\log_2\,(.5) + .3\,\log_2\,(.3) + .1\,\log\,(.1) + .1\,\log\,(.1)\big) \approx 1.685\,.$$

20.4 Perfect Secrecy

Intuitively, the one-time pad provides perfect secrecy. In Section 4.4, we gave a mathematical meaning to this statement. In the present section, we repeat some of the arguments of that section and phrase some of the ideas in terms of entropy.

Suppose we have a cipher system with possible plaintexts P, ciphertexts C, and keys K. Each plaintext in P has a certain probability of occurring; some are more likely than others. The choice of a key in K is always assumed to be independent of the choice of plaintext. The possible ciphertexts in C have various probabilities, depending on the probabilities for P and K.

If Eve intercepts a ciphertext, how much information does she obtain for the key? In other words, what is $H(K|C)$? Initially, the uncertainty in the key was $H(K)$. Has the knowledge of the ciphertext decreased the uncertainty?

Example

Suppose we have three possible plaintexts: $a,\ b,\ c$ with probabilities .5, .3, .2 and two keys $k_1,\ k_2$ with probabilities .5 and .5. Suppose the possible ciphertexts are $U,\ V,\ W$. Let e_k be the encryption function for the key k. Suppose

$$\begin{gathered}e_{k_1}(a)=U,\ e_{k_1}(b)=V,\ e_{k_1}(c)=W\\ e_{k_2}(a)=U,\ e_{k_2}(b)=W,\ e_{k_2}(c)=V.\end{gathered}$$

Let $p_P(a)$ denote the probability that the plaintext is a, etc. The probability that the ciphertext is U is

$$\begin{aligned}p_C(U) &= p_K(k_1)p_P(a) + p_K(k_2)p_P(a)\\ &= (.5)(.5) + (.5)(.5) = .50.\end{aligned}$$

Similarly, we calculate $p_C(V) = .25$ and $p_C(W) = .25$.

Suppose someone intercepts a ciphertext. This gives some information on the plaintext. For example, if the ciphertext is U, then it can be deduced immediately that the plaintext was a. If the ciphertext is V, the plaintext was either b or c.

We can even say more: The probability that a ciphertext is V is .25, so the conditional probability that the plaintext was b, given that the ciphertext is V is

$$p(b|V) = \frac{p_{(P,\,C)}(b,\,V)}{p_C(V)} = \frac{p_{(P,\,K)}(b,\,k_1)}{p_C(V)} = \frac{(.3)(.5)}{.25} = .6.$$

Similarly, $p(c|V) = .4$ and $p(a|V) = 0$. We can also calculate

$$p(a|W) = 0, \quad p(b|W) = .6, \quad p(c|W) = .4.$$

Note that the original probabilities of the plaintexts were .5, .3, and .2; knowledge of the ciphertext allows us to revise the probabilities. Therefore, the ciphertext gives us information about the plaintext. We can quantify this via the concept of conditional entropy. First, the entropy of the plaintext is

$$H(P) = -\big(.5\log_2(.5) + .3\log_2(.3) + .2\log_2(.2)\big) = 1.485.$$

The conditional entropy of P given C is

$$H(P|C) = -\sum_{x\in\{a,\,b,\,c\}}\ \sum_{Y\in\{U,\,V,\,W\}} p(Y)p(x|Y)\log_2(p(x|Y)) = .485.$$

Therefore, in the present example, the uncertainty for the plaintext decreases when the ciphertext is known.

On the other hand, we suspect that for the one-time pad the ciphertext yields no information about the plaintext that was not known before. In other words, the uncertainty for the plaintext should equal the uncertainty for the plaintext given the ciphertext. This leads us to the following definition and theorem.

Definition

A cryptosystem has **perfect secrecy** if $H(P|C) = H(P)$.

Theorem

The one-time pad has perfect secrecy.

Proof. Recall that the basic setup is the following: There is an alphabet with Z letters (for example, Z could be 2 or 26). The possible plaintexts consist of strings of characters of length L. The ciphertexts are strings of characters of length L. There are Z^L keys, each consisting of a sequence of length L denoting the various shifts to be used. The keys are chosen randomly, so each occurs with probability $1/Z^L$.

Let $c \in C$ be a possible ciphertext. As before, we calculate the probability that c occurs:

$$p_C(c) = \sum_{\substack{x \in P,\, k \in K \\ e_k(x) = c}} p_P(x) p_K(k).$$

Here $e_k(x)$ denotes the ciphertext obtained by encrypting x using the key k. The sum is over those pairs x, k such that k encrypts x to c. Note that we have used the independence of P and K to write joint probability $p_{(P,\,K)}(x,\,k)$ as the product of the individual probabilities.

In the one-time pad, every key has equal probability $1/Z^L$, so we can replace $p_K(k)$ in the above sum by $1/Z^L$. We obtain

$$p_C(c) = \frac{1}{Z^L} \sum_{\substack{x \in P,\, k \in K \\ e_k(x)=c}} p_P(x).$$

We now use another important feature of the one-time pad: For each plaintext x and each ciphertext c, there is exactly one key k such that $e_k(x) = c$. Therefore, every $x \in P$ occurs exactly once in the preceding sum, so we have $Z^{-L} \sum_{x \in P} p_P(x)$. But the sum of the probabilities of all possible plaintexts is 1, so we obtain

$$p_C(c) = \frac{1}{Z^L}.$$

This confirms what we already suspected: Every ciphertext occurs with equal probability.

Now let's calculate some entropies. Since K and C each have equal probabilities for all Z^L possibilities, we have

$$H(K) = H(C) = \log_2 \left(Z^L\right).$$

We now calculate $H(P, K, C)$ in two different ways. Since knowing (P, K, C) is the same as knowing (P, K), we have

$$H(P, K, C) = H(P, K) = H(P) + H(K).$$

The last equality is because P and K are independent. Also, knowing (P, K, C) is the same as knowing (P, C) since C and P determine K for the one-time pad. Therefore,

$$H(P, K, C) = H(P, C) = H(P|C) + H(C).$$

The last equality is the chain rule. Equating the two expressions, and using the fact that $H(K) = H(C)$, we obtain $H(P|C) = H(P)$. This proves that the one-time pad has perfect secrecy.

The preceding proof yields the following more general result. Let $\#K$ denote the number of possible keys, etc.

Theorem

Consider a cryptosystem such that

1. Every key has probability $1/\#K$.
2. For each $x \in P$ and $c \in C$ there is exactly one $k \in K$ such that $e_k(x) = c$.

Then this cryptosystem has perfect secrecy.

It is easy to deduce from condition (2) that $\#C = \#K$. Conversely, it can be shown that if $\#P = \#C = \#K$ and the system has perfect secrecy, then (1) and (2) hold (see [Stinson, Theorem 2.4]).

It is natural to ask how the preceding concepts apply to RSA. The possibly surprising answer is that $H(P|C) = 0$; namely, the ciphertext determines the plaintext. The reason is that entropy does not take into account computation time. The fact that it might take billions of years to factor n is irrelevant. What counts is that all the information needed to recover the plaintext is contained in the knowledge of n, e, and c.

The more relevant concept for RSA is the computational complexity of breaking the system.

20.5 The Entropy of English

In an English text, how much information is obtained per letter? If we had a random sequence of letters, each appearing with probability 1/26, then the entropy would be $\log_2 (26) = 4.70$; so each letter would contain 4.7 bits of information. If we include spaces, we get $\log_2 (27) = 4.75$. But the letters are not equally likely: a has frequency .082, b has frequency .015, etc. (see Section 2.3). Therefore, we consider

$$-\left(.082 \log_2 .082 + .015 \log_2 .015 + \cdots\right) = 4.18.$$

However, this doesn't tell the whole story. Suppose we have the sequence of letters *we are studyin*. There is very little uncertainty as to what the last letter is; it is easy to guess that it is g. Similarly, if we see the letter q, it is extremely likely that the next letter is u. Therefore, the existing letters often give information about the next letter, which means that there is not as much additional information carried by that letter. This says that the entropy calculated previously is still too high. If we use tables of the frequencies of the $26^2 = 676$ digrams (a digram is a two-letter combination), we can calculate the conditional entropy of one letter, given the preceding letter, to be 3.56. Using trigram frequencies, we find that the conditional entropy of a letter, given the preceding two letters, is approximately 3.3. This means that, on the average, if we know two consecutive letters in a text, the following letter carries 3.3 bits of additional information. Therefore, if we have a long text, we should expect to be able to compress it at least by a factor of around $3.3/4.7 = .7$.

Let L represent the letters of English. Let L^N denote the N-gram combinations. Define the entropy of English to be

$$H_{\text{English}} = \lim_{N\to\infty} \frac{H(L^N)}{N},$$

where $H(L^N)$ denotes the entropy of N-grams. This gives the average amount of information per letter in a long text, and it also represents the average amount of uncertainty in guessing the next letter, if we already know a lot of the text. If the letters were all independent of each other, so the probability of the digram qu equaled the probability of q times the probability of u, then we would have $H(L^N) = N \cdot H(L)$, and the limit would be $H(L)$, which is the entropy for one-letter frequencies. But the interactions of letters, as noticed in the frequencies for digrams and trigrams, lower the value of $H(L^N)$.

How do we compute $H(L^N)$? Calculating 100-gram frequencies is impossible. Even tabulating the most common of them and getting an approximation would be difficult. Shannon proposed the following idea.

Suppose we have a machine that is an optimal predictor, in the sense that, given a long string of text, it can calculate the probabilities for the letter that will occur next. It then guesses the letter with highest probability. If correct, it notes the letter and writes down a 1. If incorrect, it guesses the second most likely letter. If correct, it writes down a 2, etc. In this way, we obtain a sequence of numbers. For example, consider the text $itissunnytoday$. Suppose the predictor says that t is the most likely for the 1st letter, and it is wrong; its second guess is i, which is correct, so we write the i and put 2 below it. The predictor then predicts that t is the next letter, which is correct. We put 1 beneath the t. Continuing, suppose it finds i on its 1st guess, etc. We obtain a situation like the following:

$$\begin{array}{cccccccccccccc} i & t & i & s & s & u & n & n & y & t & o & d & a & y \\ 2 & 1 & 1 & 1 & 4 & 3 & 2 & 1 & 4 & 1 & 1 & 1 & 1 & 1 \end{array}$$

Using the prediction machine, we can reconstruct the text. The prediction machine says that its second guess for the first letter will be i, so we know the 1st letter is i. The predictor says that its first guess for the next letter is t, so we know that's next. The first guess for the next is i, etc.

What this means is that if we have a machine for predicting, we can change a text into a string of numbers without losing any information, because we can reconstruct the text. Of course, we could attempt to write a computer program to do the predicting, but Shannon suggested that the best predictor is a person who speaks English. Of course, a person is unlikely to be as deterministic as a machine, and repeating the experiment (assuming the person forgets the text from the first time) might not yield an identical result. So reconstructing the text might present a slight difficulty. But it is still a reasonable assumption that a person approximates an optimal predictor.

Given a sequence of integers corresponding to a text, we can count the frequency of each number. Let

$$q_i = \text{ frequency of the number } i.$$

Since the text and the sequence of numbers can be reconstructed from each other, their entropies must be the same. The largest the entropy can be for the sequence of numbers is when these numbers are independent. In this case, the entropy is $-\sum_{i=1}^{26} q_i \log_2 (q_i)$. However, the numbers are probably not independent. For example, if there are a couple consecutive 1s, then perhaps the predictor has guessed the rest of the word, which means that there will be a few more 1s. However, we get an upper bound for the entropy, which is usually better than the one we obtain using frequencies of letters. Moreover, Shannon also found a lower bound for the entropy. His results are

$$\sum_{1=1}^{26} i(q_i - q_{i+1}) \log_2 (i) \le H_{\text{English}} \le -\sum_{i=1}^{26} q_i \log_2 (q_i).$$

Actually, these are only approximate upper and lower bounds, since there is experimental error, and we are really considering a limit as $N \to \infty$.

These results allow an experimental estimation of the entropy of English. Alice chooses a text and Bob guesses the first letter, continuing until the correct guess is made. Alice records the number of guesses. Bob then tries to guess the second letter, and the number of guesses is again recorded. Continuing in this way, Bob tries to guess each letter. When he is correct, Alice tells him and records the number of guesses. Shannon gave Table 20.1 as a typical result of an experiment. Note that he included spaces, but ignored punctuation, so he had 27 possibilities: There are 102 symbols. There are seventy-nine 1s, eight 2s, three 3s, etc. This gives

$$q_1 = 79/102, \quad q_2 = 8/102, \quad q_3 = 3/102, \quad q_4 = q_5 = 2/102,$$
$$q_6 = 3/102, \quad q_7 = q_8 = q_{11} = q_{15} = q_{17} = 1/102.$$

Table 20.1 Shannon’s Experiment on the Entropy of English

t h e r e i s n o r e v e r s e

1 1 1 5 1 1 2 1 1 2 1 1 15 1 17 1 1 1 2 1

o n a m o t o r c y c l e a

3 2 1 2 2 7 1 1 1 1 4 1 1 1 1 1 3 1

f r i e n d o f m i n e f o u n d

8 6 1 3 1 1 1 1 1 1 1 1 1 1 6 2 1 1 1 1

t h i s o u t r a t h e r

1 1 2 1 1 1 1 1 1 4 1 1 1 1 1 1

d r a m a t i c a l l y t h e

11 5 1 1 1 1 1 1 1 1 1 1 1 6 1 1 1

o t h e r d a y

1 1 1 1 1 1 1 1 1 1

Table 20.1 Full Alternative Text

The upper bound for the entropy is therefore

$$-(\frac{79}{102}\log_2\frac{79}{102}+\cdots+\frac{1}{102}\log_2\frac{1}{102})\approx 1.42.$$

Note that since we are using $0\ \log_2\ 0=0$, the terms with $q_i=0$ can be omitted. The lower bound is

$$1 \cdot \left(\frac{79}{102} - \frac{8}{102}\right) \log_2(1) + 2 \cdot \left(\frac{8}{102} - \frac{3}{102}\right) \log_2(2) + \ldots \approx .72.$$

A reasonable estimate is therefore that the entropy of English is near 1, maybe slightly more than 1.

If we want to send a long English text, we could write each letter (and the space) as a string of five bits. This would mean that a text of length 102, such as the preceding, would require 510 bits. It would be necessary to use something like this method if the letters were independent and equally likely. However, suppose we do a Huffman encoding of the message $1, 1, 1, 5, 1, 1, 2, \ldots$ from Table 20.1. Let

$$\begin{gathered} 1 \leftrightarrow 1 \quad 2 \leftrightarrow 110 \quad 3 \leftrightarrow 1010 \quad 4 \leftrightarrow 0100 \\ 5 \leftrightarrow 11100 \quad 6 \leftrightarrow 0010 \quad 7 \leftrightarrow 01100 \quad 8 \leftrightarrow 11000 \\ 11 \leftrightarrow 01000 \quad 15 \leftrightarrow 10000 \quad 17 \leftrightarrow 100000. \end{gathered}$$

All other numbers up to 27 can be represented by various combinations of six or more bits. To send the message requires

$$79 \cdot 1 + 8 \cdot 3 + 3 \cdot 4 + 2 \cdot 4 + \cdots + 1 \cdot 6 = 171 \text{ bits},$$

which is 1.68 bits per letter.

Note that five bits per letter is only slightly more than the “random” entropy 4.75, and 1.68 bits per letter is slightly more than our estimate of the entropy of English. These agree with the result that entropy differs from the average length of a Huffman encoding by at most 1.

One way to look at the preceding entropy calculations is to say that English is around 75% redundant. Namely, if we send a long message in standard written English, compared to the optimally compressed text, the ratio is approximately 4 to 1 (that is, the random entropy 4.75 divided by the entropy of English, which is around 1). In our example, we were close, obtaining a ratio near 3 to 1 (namely 4.75/1.68).

Define the **redundancy** of English to be

$$R = 1 - \frac{H_{\text{English}}}{\log_2{(26)}}.$$

Then R is approximately 0.75, which is the 75% redundancy mentioned previously.

20.5.1 Unicity Distance

Suppose we have a ciphertext. How many keys will decrypt it to something meaningful? If the text is long enough, we suspect that there is a unique key and a unique corresponding plaintext. The unicity distance n_0 for a cryptosystem is the length of ciphertext at which one expects that there is a unique meaningful plaintext. A rough estimate for the unicity distance is

$$n_0 = \frac{\log_2{|K|}}{R \log_2{|L|}},$$

where $|K|$ is the number of possible keys, $|L|$ is the number of letters or symbols, and R is the redundancy (see [Stinson]). We'll take $R = .75$ (whether we include spaces in our language or not; the difference is small).

For example, consider the substitution cipher, which has $26!$ keys. We have

$$n_0 = \frac{\log_2{26!}}{.75 \log_2{26}} \approx 25.1.$$

This means that if a ciphertext has length 25 or more, we expect that usually there is only one possible meaningful plaintext. Of course, if we have a ciphertext of length 25, there are probably several letters that have not appeared. Therefore, there could be several possible keys, all of which decrypt the ciphertext to the same plaintext.

As another example, consider the affine cipher. There are 312 keys, so

$$n_0 = \frac{\log_2{312}}{.75 \log_2{26}} \approx 2.35.$$

This should be regarded as only a very rough approximation. Clearly it should take a few more letters to get a unique decryption. But the estimate of 2.35 indicates that very few letters suffice to yield a unique decryption in most cases for the affine cipher.

Finally, consider the one-time pad for a message of length N. The encryption is a separate shift mod 26 for each letter, so there are 26^N keys. We obtain the estimate

$$n_0 \approx \frac{\log_2 26^N}{.75 \log_2 26} = 1.33N.$$

In this case, it says we need more letters than the entire ciphertext to get a unique decryption. This reflects the fact that all plaintexts are possible for any ciphertext.

20.6 Exercises

1. Let X_1 and X_2 be two independent tosses of a fair coin. Find the entropy $H(X_1)$ and the joint entropy $H(X_1, X_2)$. Why is $H(X_1, X_2) = H(X_1) + H(X_2)$?

2. Consider an unfair coin where the two outcomes, heads and tails, have probabilities $p(heads) = p$ and $p(tails) = 1 - p$.

    1. If the coin is flipped two times, what are the possible outcomes along with their respective probabilities?

    2. Show that the entropy in part (a) is $-2p \log_2(p) - 2(1-p)\log_2(1-p)$. How could this have been predicted without calculating the probabilities in part (a)?

3. A random variable X takes the values $1, 2, \cdots, n, \cdots$ with probabilities $\frac{1}{2}, \frac{1}{2^2}, \cdots, \frac{1}{2^n}, \cdots$. Calculate the entropy $H(X)$.

4. Let X be a random variable taking on integer values. The probability is 1/2 that X is in the range $[0, 2^8 - 1]$, with all such values being equally likely, and the probability is 1/2 that the value is in the range $[2^8, 2^{32} - 1]$, with all such values being equally likely. Compute $H(X)$.

5. Let X be a random event taking on the values $-2, -1, 0, 1, 2$, all with positive probability. What is the general inequality/equality between $H(X)$ and $H(Y)$, where Y is the following?

    1. $Y = 2^X$

    2. $Y = X^2$

6.
    1. In this problem we explore the relationship between the entropy of a random variable X and the entropy of a function $f(X)$ of the random variable. The following is a short proof that shows $H(f(X)) \le H(X)$. Explain what principles are used in each of the steps.

    $$H(X, f(X)) = H(X) + H(f(X)|X) = H(X),$$
    $$H(X, f(X)) = H(f(X)) + H(X|f(X)) \ge H(f(X)).$$

    2. Letting X take on the values ± 1 and letting $f(x) = x^2$, show that it is possible to have $H(f(X)) < H(X)$.

3. In part (a), show that you have equality if and only if f is a one-to-one function (more precisely, f is one-to-one on the set of outputs of X that have nonzero probability).

4. The preceding results can be used to study the behavior of the run length coding of a sequence. Run length coding is a technique that is commonly used in data compression. Suppose that $X_1, X_2, \cdots, X_n$ are random variables that take the values 0 or 1. This sequence of random variables can be thought of as representing the output of a binary source. The run length coding of $X_1, X_2, \cdots, X_n$ is a sequence $\mathbf{L} = (L_1, L_2, \cdots, L_k)$ that represents the lengths of consecutive symbols with the same value. For example, the sequence 110000100111 has a run length sequence of $\mathbf{L} = (2, 4, 1, 2, 3)$. Observe that **L** is a function of $X_1, X_2, \cdots, X_n$. Show that **L** and X_1 uniquely determine $X_1, X_2, \cdots, X_n$. Do **L** and X_n determine $X_1, X_2, \cdots, X_n$? Using these observations and the preceding results, compare $H(X_1, X_2, \cdots, X_n)$, $H(\mathbf{L})$, and $H(\mathbf{L}, X_1)$.

7. A bag contains five red balls, three white balls, and two black balls that are identical to each other in every manner except color.

   1. Choose two balls from the bag with replacement. What is the entropy of this experiment?

   2. What is the entropy of choosing two balls without replacement? (Note: In both parts, the order matters; i.e., red then white is not the same as white then red.)

8. We often run into situations where we have a sequence of n random events. For example, a piece of text is a long sequence of letters. We are concerned with the rate of growth of the joint entropy as n increases. Define the entropy rate of a sequence $\mathbf{X} = \{X_k\}$ of random events as

$$H_\infty(\mathbf{X}) = \lim_{n\to\infty} \frac{1}{n} H(X_1, X_2, \cdots X_n).$$

   1. A very crude model for a language is to assume that subsequent letters in a piece of text are independent and come from identical probability distributions. Using this, show that the entropy rate equals $H(X_1)$.

   2. In general, there is dependence among the random variables. Assume that $X_1, X_2, \cdots X_n$ have the same probability distribution but are somehow dependent on each other (for example, if I give you the letters TH you can guess that the next letter is E). Show that

$$H(X_1, X_2, \cdots X_n) \le \sum_k H(X_k)$$

and thus that

$$H_\infty(\mathrm{X}) \le H(X_1)$$

(if the limit defining H_∞ exists).

9. Suppose we have a cryptosystem with only two possible plaintexts. The plaintext a occurs with probability $1/3$ and b occurs with probability $2/3$. There are two keys, k_1 and k_2, and each is used with probability $1/2$. Key k_1 encrypts a to A and b to B. Key k_2 encrypts a to B and b to A.

    1. Calculate $H(P)$, the entropy for the plaintext.

    2. Calculate $H(P|C)$, the conditional entropy for the plaintext given the ciphertext. (*Optional hint:* This can be done with no additional calculation by matching up this system with another well-known system.)

10. Consider a cryptosystem $\{P, K, C\}$.

    1. Explain why $H(P, K) = H(C, P, K) = H(P) + H(K)$.

    2. Suppose the system has perfect secrecy. Show that

    $$H(C, P) = H(C) + H(P)$$

    and

    $$H(C) = H(K) - H(K|C, P).$$

    3. Suppose the system has perfect secrecy and, for each pair of plaintext and ciphertext, there is at most one corresponding key that does the encryption. Show that $H(C) = H(K)$.

11. Prove that for a cryptosystem $\{P, K, C\}$ we have

$$H(C|P) = H(P, K, C) - H(P) - H(K|C, P) = H(K) - H(K|C, P).$$

12. Consider a Shamir secret sharing scheme where any five people of a set of 20 can determine the secret K, but no fewer can do so. Let $H(K)$ be the entropy of the choice of K, and let $H(K|S_1)$ be the conditional entropy of K, given the information supplied to the first person. What are the relative sizes of $H(K)$ and $H(K|S_1)$? (Larger, smaller, equal?)

13. Let X be a random event taking on the values $1, 2, 3, \ldots, 36$, all with equal probability.

1. What is the entropy $H(X)$?

2. Let $Y = X^{36} (\mathrm{mod} 37)$. What is $H(Y)$?

14.

1. Show that the maximum of $-p \log_2 p - (1-p) \log_2 (1-p)$ for $0 \le p \le 1$ occurs when $p = 1/2$.

2. Let $p_i \ge 0$ for $1 \le i \le n$. Show that the maximum of

$$-\sum_i p_i \log_2 p_i,$$

subject to the constraint $\sum_i p_i = 1$, occurs when $p_1 = \cdots = p_n$. (Hint: Lagrange multipliers could be useful in this problem.)

15.

1. Suppose we define $\tilde{H}(Y|X) = -\sum_{x,\,y} p_Y(y|x) \log_2 p_Y(y|x)$. Show that if X and Y are independent, and X has $|X|$ possible outputs, then $\tilde{H}(Y|X) = |X| H(Y) \ge H(Y)$
.

2. Use (a) to show that $\tilde{H}(Y|X)$ is not a good description of the uncertainty of Y given X.