21.11 Exercises

Exercise *2.1* [⋆]

This exercise indicates the kind of facility with set theory needed for this book, and summarizes a few useful results in probability theory. Use set theory and the axioms defining a probability function to show that:

a. $P(A \cup B) = P(A) + P(B) - P(A \cap B)$ [the addition rule]

b. P(0) = 0

c. $P(\overline{A})$ = 1 -P(A)

d. $A \subseteq B \Rightarrow P(A) \leq P(B)$

e. $P(B - A) = P(B) - P(A \cap B)$

Exercise *2.2* [⋆]

Assume the following sample space:

(2.23) Ω = {is-noun, has-plural-s, is-adjective, is-verb]

and the function $f : 2^{\Omega} \rightarrow [0, 1]$ with the following values:

x	$f(x)$
{ is-noun }	0.45
{ has-plural-s }	0.2
{ is-adjective }	0.25
{ is-verb }	0.3

Can f be extended to all of 2^{Ω} such that it is a well-formed probability distribution? If not, how would you model these data probabilistically?

Exercise 2.3 [⋆]

Compute the probability of the event 'A period occurs after a three-letter word and this period indicates an abbreviation (not an end-of-sentence marker),' assuming the following probabilities.

(2.24) P(is-abbreviation] three-letter-word) = 0.8

(2.25) P(three-letter-word) = 0.0003

Exercise 2.4 [⋆]

Are X and Y as defined in the following table independently distributed?

x	0	0	1	1
Y	0	1	0	1
p(X = x, Y = y)	0.32	0.08	0.48	0.12

Exercise 2.5 [⋆]

In example 5, we worked out the expectation of the sum of two dice in terms of the expectation of rolling one die. Show that one gets the same result if one calculates the expectation for two dice directly.

Exercise 2.6 [⋆⋆]

Consider the set of grades you have received for courses taken in the last two years. Convert them to an appropriate numerical scale. What is the appropriate distribution for modeling them?

Exercise 2.7 [⋆⋆]

Find a linguistic phenomenon that the binomial distribution is a good model for. What is your best estimate for the parameter p?

Exercise 2.8 [⋆⋆]

For $i = 8$ and j = 2, confirm that the maximum of equation (2.15) is at 0.8, and that the maximum of equation (2.17) is 0.75. Suppose our prior belief had instead been captured by the equation:

$$P(\mu_m) = 30m^2(1-m)^2$$

What then would the MAP probability be after seeing a particular sequence of 8 heads and 2 tails? (Assume the theory μ_m and a prior belief that the coin is fair.)

2.2 Essential Information Theory

The field of information theory was developed in the 1940s by Claude Shannon, with the initial exposition reported in (Shannon 1948). Shannon was interested in the problem of maximizing the amount of information that you can transmit over an imperfect communication channel such as a noisy phone line (though actually many of his concerns stemmed from codebreaking in World War II). For any source of 'information' and any 'communication channel,' Shannon wanted to be able to determine theoretical maxima for (i) data compression – which turns out to be given by the Entropy H (or more fundamentally, by the Kolmogorov complexity K), and (ii) the transmission rate – which is given by the Channel Capacity C. Until Shannon, people had assumed that necessarily, if you send your message at a higher speed, then more errors must occur during the transmission. But Shannon showed that providing that you transmit the information in your message at a slower rate than the Channel Capacity, then you can make the probability of errors in the transmission of your message as small as you would like.

2.2.1 Entropy

ALPHABET

Let p(x) be the probability mass function of a random variable X, over a discrete set of symbols (or *alphabet*) X:

$$p(x) = P(X = x), \quad x \in x$$

For example, if we toss two coins and count the number of heads, we have a random variable: $p(0) = 1/4, p(1) = 1/2, p(2) = 1/4$.

ENTROPY
SELF-INFORMATION

The *entropy* (or self-information) is the average uncertainty of a single random variable:

(2.26) **Entropy** $$H(p) = H(X) = -\sum_{x \in X} p(x) \log_2 p(x)$$

Entropy measures the amount of information in a random variable. It is normally measured in bits (hence the log to the base 2), but using any other base yields only a linear scaling of results. For the rest of this book, an unadorned log should be read as log to the base 2. Also, for this definition to make sense, we define 0 log 0 = 0.

Example 7: Suppose you are reporting the result of rolling an 8-sided die. Then the entropy is:

$$H(X) = -\sum_{i=1}^{8} p(i) \log p(i) = -\sum_{i=1}^{8} \frac{1}{8} \log \frac{1}{8} = -\log \frac{1}{8} = \log 8 = 3 \text{ bits}$$

This result is what we would expect. Entropy, the amount of information in a random variable, can be thought of as the average length of the message needed to transmit an outcome of that variable. If we wish to send the result of rolling an eight-sided die, the most efficient way is to simply encode the result as a 3 digit binary message:

1	2	3	4	5	6	7	8
001	010	011	100	101	110	111	000

The transmission cost of each result is 3 bits, and there is no cleverer way of encoding the results with a lower average transmission cost. In general, an optimal code sends a message of probability p(i) in $\lceil -\log p(i) \rceil$ bits.

The minus sign at the start of the formula for entropy can be moved inside the logarithm, where it becomes a reciprocal:

(2.27) $$H(X) = \sum_{x \in X} p(x) \log \frac{1}{p(x)}$$

People without any statistics background often think about a formula like this as a sum of the quantity p(x) log(1/ p(x)) for each x. While this is mathematically impeccable, it is the wrong way to think about such equations. Rather you should think of $\sum_{x \in X}$ p(x) ... as an idiom. It says to take a weighted average of the rest of the formula (which will be a function of *x*), where the weighting depends on the probability of each x. Technically, this idiom defines an *expectation,* as we saw earlier. Indeed,

$$H(X) = E\left(\log \frac{1}{p(X)}\right) \tag{2.28}$$

Example 8: Simplified Polynesian Simplified Polynesian[3] appears to be just a random sequence of letters, with the letter frequencies as shown:

p	t	k	a	i	u
1/8	1/4	1/8	1/4	1/8	1/8

Then the per-letter entropy is:

$$\begin{aligned} H(P) &= -\sum_{i \in \{p,t,k,a,i,u\}} P(i) \log P(i) \\ &= -\left[4 \times \frac{1}{8} \log \frac{1}{8} + 2 \times \frac{1}{4} \log \frac{1}{4}\right] \\ &= 2\frac{1}{2} \text{ bits} \end{aligned}$$

This is supported by the fact that we can design a code that on average takes $2\frac{1}{2}$ bits to transmit a letter:

p	t	k	a	i	u
100	00	101	01	110	111

Note that this code has been designed so that fewer bits are used to send more frequent letters, but still so that it can be unambiguously decoded – if a code starts with a 0 then it is of length two, and if it starts with a 1 it is of length 3. There is much work in information theory on the design of such codes, but we will not further discuss them here.

Twenty Questions

One can also think of entropy in terms of the Twenty *Questions* game. If you can ask yes/no questions like 'Is it a *t* or an a?' or 'Is it a consonant?' then on average you will need to ask $2\frac{1}{2}$ questions to identify each letter with total certainty (assuming that you ask good questions!). In

3. Polynesian languages, such as Hawai'ian, are well known for their small alphabets.

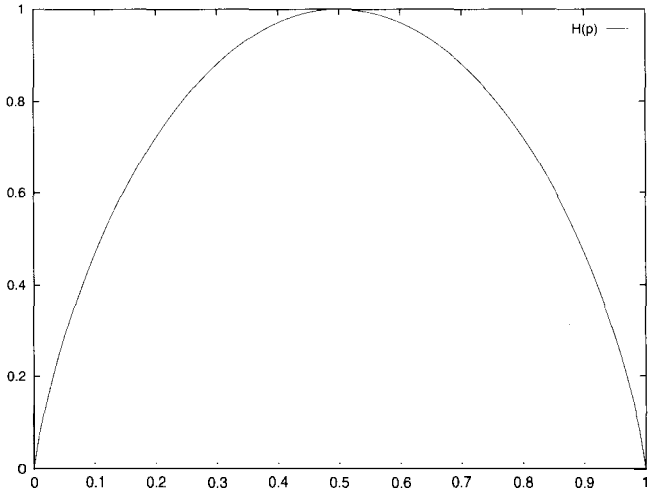

Figure 2.5 The entropy of a weighted coin. The horizontal axis shows the probability of a weighted coin to come up heads. The vertical axis shows the entropy of tossing the corresponding coin once.

other words, entropy can be interpreted as a measure of the size of the 'search space' consisting of the possible values of a random variable and its associated probabilities.

Note that: (i) $H(X) \geq 0$, (ii) $H(X) = 0$ only when the value of X is determinate, hence providing no new information, and that (iii) entropy increases with the message length. The information needed to transmit the results of tossing a possibly weighted coin depends on the probability p that it comes up heads, and on the number of tosses made. The entropy for a single toss is shown in figure 2.5. For multiple tosses, since each is independent, we would just multiply the number in the graph by the number of tosses.

2.2.2 Joint entropy and conditional entropy

The joint entropy of a pair of discrete random variables X, Y $\sim p(x, y)$ is the amount of information needed on average to specify both their values. It is defined as:

$$H(X, Y) = -\sum_{x \in X} \sum_{y \in Y} p(x, y) \log p(x, y) \tag{2.29}$$

The conditional entropy of a discrete random variable Y given another X, for X, Y $\sim p(x, y)$, expresses how much extra information you still need to supply on average to communicate Y given that the other party knows X:

$$
\begin{aligned}
(2.30)\quad H(Y|X) &= \sum_{x \in \mathcal{X}} p(x) H(Y|X = x) \\
&= \sum_{x \in \mathcal{X}} p(x) \left[- \sum_{y \in \mathcal{Y}} p(y|x) \log p(y|x) \right] \\
&= - \sum_{x \in \mathcal{X}} \sum_{\text{yty}} p(x, y) \log p(y|x)
\end{aligned}
$$

There is also a Chain rule for entropy:

$$
\begin{aligned}
(2.31)\quad H(X, Y) &= H(X) + H(Y|X) \\
H(X_1, \ldots, X_n) &= H(X_1) + H(X_2|X_1) + \ldots + H(X_n|X_1, \ldots, X_{n-1})
\end{aligned}
$$

The products in the chain rules for probabilities here become sums because of the log:

$$
\begin{aligned}
H(X, Y) &= -E_{p(x,y)} (\log p(x, y)) \\
&= -E_{p(x,y)} (\log(p(x) p(y|x))) \\
&= -E_{p(x,y)} (\log p(x) + \log p(y|x)) \\
&= -E_{p(x)} (\log p(x)) - E_{p(x,y)} (\log p(y|x)) \\
&= H(X) + H(Y|X)
\end{aligned}
$$

Example 9: Simplified Polynesian revisited An important scientific idea is the distinction between a model and reality. Simplified Polynesian isn't a random variable, but we approximated it (or modeled it) as one. But now let's learn a bit more about the language. Further fieldwork has revealed that Simplified Polynesian has syllable structure. Indeed, it turns out that all words consist of sequences of CV (consonant-vowel) syllables. This suggests a better model in terms of two random variables C for the consonant of a syllable, and V for the vowel, whose joint distribution $P(C, V)$ and marginal distributions $P(C, \cdot)$ and $P(\cdot, V)$ are as follows:

(2.32)

	p	t	k	
a	$\frac{1}{16}$	$\frac{3}{8}$	$\frac{1}{16}$	$\frac{1}{2}$
i	$\frac{1}{16}$	$\frac{3}{16}$	0	$\frac{1}{4}$
u	0	$\frac{3}{16}$	$\frac{1}{16}$	$\frac{1}{4}$
	$\frac{1}{8}$	$\frac{3}{4}$	$\frac{1}{8}$	

Note that here the marginal probabilities are on a per-syllable basis, and are therefore double the probabilities of the letters on a per-letter basis, which would be:

(2.33)

p	t	k	a	i	u
1/16	3/8	1/16	1/4	1/8	1/8

We can work out the entropy of the joint distribution, in more than one way. Let us use the chain rule:[4]

$$\begin{aligned} H(C) &= 2 \times \frac{1}{8} \times 3 + \frac{3}{4}(2 - \log 3) \\ &= \frac{9}{4} - \frac{3}{4}\log 3 \text{ bits} \approx 1.061 \text{ bits} \end{aligned}$$

$$\begin{aligned} H(V|C) &= \sum_{c=p,t,k} p(C = c) H(V|C = c) \\ &= \frac{1}{8}H\left(\frac{1}{2}, \frac{1}{2}, 0\right) + \frac{3}{4}H\left(\frac{1}{2}, \frac{1}{4}, \frac{1}{4}\right) + \frac{1}{8}H\left(\frac{1}{2}, 0, \frac{1}{2}\right) \\ &= 2 \times \frac{1}{8} \times 1 + \frac{3}{4}\left[\frac{1}{2} \times 1 + 2 \times \frac{1}{4} \times 2\right] \\ &= \frac{1}{4} + \frac{3}{4} \times \frac{3}{2} \\ &= \frac{11}{8} \text{ bits} = 1.375 \text{ bits} \end{aligned}$$

$$\begin{aligned} H(C, V) &= H(C) + H(V|C) \\ &= \frac{9}{4} - \frac{3}{4}\log 3 + \frac{11}{8} \\ &= \frac{29}{8} - \frac{3}{4}\log 3 \approx 2.44 \text{ bits} \end{aligned}$$

4. Within the calculation, we use an informal, but convenient, notation of expressing a finite-valued distribution as a sequence of probabilities, which we can calculate the entropy of.

Note that those 2.44 bits are now the entropy for a whole syllable (which was 2 x $2\frac{1}{2}$ = 5 for the original Simplified Polynesian example). Our better understanding of the language means that we are now much less uncertain, and hence less surprised by what we see on average than before.

Because the amount of information contained in a message depends on the length of the message, we normally want to talk in terms of the per-letter or per-word entropy. For a message of length n, the per-letter/word entropy, also known as the *entropy rate,* is:[5]

ENTROPY RATE

$$(2.34)\quad H_{\text{rate}} = \frac{1}{n}H(X_{1n}) = -\frac{1}{n}\sum_{x_{1n}} p(x_{1n}) \log p(x_{1n})$$

If we then assume that a language is a stochastic process consisting of a sequence of tokens L = (Xi), for example a transcription of every word you utter in your life, or a corpus comprising everything that is sent down the newswire to your local paper, then we can define the entropy of a human language L as the entropy rate for that stochastic process:

$$(2.35)\quad H_{\text{rate}}(L) = \lim_{n\to\infty} \frac{1}{n}H(X_1, X_2, \ldots, X_n)$$

We take the entropy rate of a language to be the limit of the entropy rate of a sample of the language as the sample gets longer and longer.

2.2.3 Mutual information

By the chain rule for entropy,

$$H(X,Y) = \text{H(X)} + H(Y|X) = \text{H(Y)} + H(X|Y)$$

Therefore,

$$\text{H(X)} - H(X|Y) = \text{H(Y)} - H(Y|X)$$

MUTUAL INFORMATION

This difference is called the *mutual information* between X and Y. It is the reduction in uncertainty of one random variable due to knowing about another, or in other words, the amount of information one random variable contains about another. A diagram illustrating the definition of mutual information and its relationship to entropy is shown in figure 2.6 (adapted from Cover and Thomas (1991: 20)).

5. Commonly throughout this book we use two subscripts on something to indicate a subsequence. So, here, we use X_{ij} to represent the sequence of random variables (Xi,. . . , X_j) and similarly $x_{ij} = (x_i, \ldots, x_j)$. This notation is slightly unusual, but very convenient when sequences are a major part of the domain of discourse. So the reader should remember this convention and be on the lookout for it.

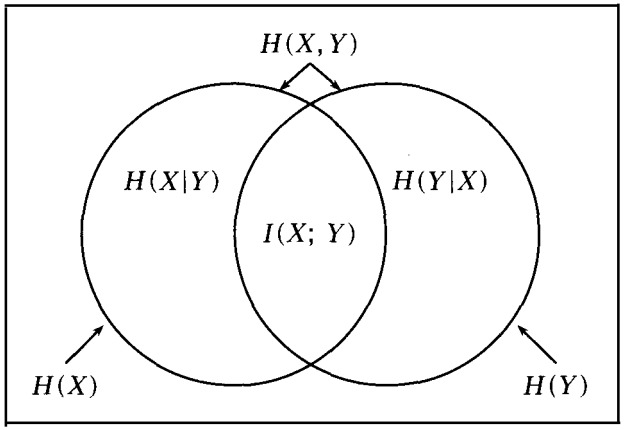

Figure 2.6 The relationship between mutual information I and entropy H.

Mutual information is a symmetric, non-negative measure of the common information in the two variables. People thus often think of mutual information as a measure of dependence between variables. However, it is actually better to think of it as a measure of independence because:

- It is 0 only when two variables are independent, but
- For two dependent variables, mutual information grows not only with the degree of dependence, but also according to the entropy of the variables.

Simple arithmetic gives us the following formulas for mutual information $I(X;Y)$:[6]

$$
\begin{aligned}
(2.36)\quad I(X;Y) &= \mathrm{H}(\mathrm{X}) - H(X|Y) \\
&= \mathrm{H}(\mathrm{X}) + \mathrm{H}(\mathrm{Y}) - H(X,Y) \\
&= \sum_{x} \mathrm{p}(x) \log \frac{1}{\mathrm{p}(x)} + \sum_{y} \mathrm{p}(y) \log \frac{1}{\mathrm{p}(y)} + \sum_{x,y} \mathrm{p}(x,y) \log \mathrm{p}(x,y) \\
&= \sum_{x,y} \mathrm{p}(x,y) \log \frac{\mathrm{p}(x,y)}{\mathrm{p}(x)\,\mathrm{p}(y)}.
\end{aligned}
$$

Since $H(X|X) = 0$, note that:

$$\mathrm{H}(\mathrm{X}) = \mathrm{H}(\mathrm{X}) - H(X|X) = I(X;X)$$

6. Mutual information is conventionally written with a semi-colon separating the two arguments. We are unsure why.

This illustrates both why entropy is also called self-information, and how the mutual information between two totally dependent variables is not constant but depends on their entropy.

We can also derive conditional mutual information and a chain rule:

$$I(X;Y|Z) = I((X;Y)|Z) = H(X|Z) - H(X|Y,Z) \tag{2.37}$$

$$\begin{aligned} I(X_{1n};\mathrm{Y}) &= I(X_1;Y) + \ldots + I(X_n;Y|X_1,\ldots.X_{n-1}) \\ &= \sum_{i=1}^{n} I(X_i;\ Y|X_1,\ldots,X_{i-1}) \end{aligned} \tag{2.38}$$

POINTWISE MUTUAL INFORMATION

In this section we have defined the mutual information between two random variables. Sometimes people talk about the ***pointwise mutual information*** between two particular points in those distributions:

$$I(x,y) = \log \frac{\mathrm{p}(x,\mathrm{Y})}{\mathrm{p}(x)\ \mathrm{P}(\mathrm{Y})}$$

This has sometimes been used as a measure of association between elements, but there are problems with using this measure, as we will discuss in section 5.4.

▼ Mutual information has been used many times in Statistical NLP, such as for clustering words (section 14.1.3). It also turns up in word sense disambiguation (section 7.2.2).

2.2.4 The noisy channel model

Using information theory, Shannon modeled the goal of communicating down a telephone line – or in general across any channel – in the following way: The aim is to optimize in terms of throughput and accuracy the communication of messages in the presence of noise in the channel. It is assumed that the output of the channel depends probabilistically on the input. In general, there is a duality between ***compression,*** which is achieved by removing all redundancy, and transmission accuracy, which is achieved by adding controlled redundancy so that the input can be recovered even in the presence of noise. The goal is to encode the message in such a way that it occupies minimal space while still containing enough redundancy to be able to detect and correct errors. On receipt, the message is then decoded to give what was most likely the original message. This process is shown in figure 2.7.

COMPRESSION

REDUNDANCY

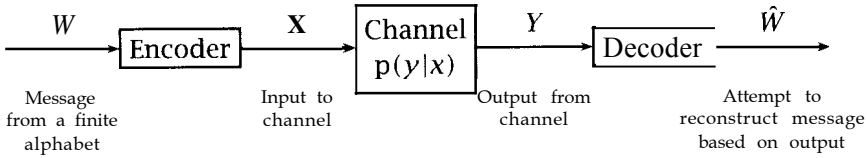

Figure 2.7 The noisy channel model.

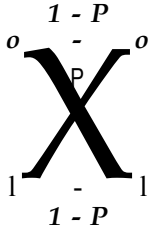

Figure 2.8 A binary symmetric channel. A 1 or a 0 in the input gets flipped on transmission with probability p.

CAPACITY

The central concept that characterizes a channel in information theory is its ***capacity.*** The channel capacity describes the rate at which one can transmit information through the channel with an arbitrarily low probability of being unable to recover the input from the output. For a memoryless channel, Shannon's second theorem states that the channel capacity can be determined in terms of mutual information as follows:

$$C = \max_{p(X)} I(X;Y) \tag{2.39}$$

According to this definition, we reach a channel's capacity if we manage to design an input code X whose distribution maximizes the mutual information between the input and the output over all possible input distributions p(X) .

As an example, consider the binary symmetric channel in figure 2.8. Each input symbol is either a 1 or a 0, and noise in the channel causes each symbol to be flipped in the output with probability ***p.*** We find that:

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

Therefore,

$$\max_{p(X)} I(X;Y) = 1 - H(p)$$

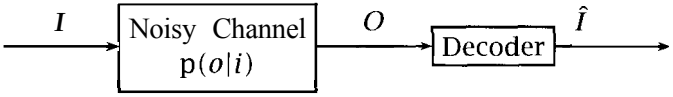

Figure 2.9 The noisy channel model in linguistics.

This last line follows because the mutual information is maximized by maximizing the entropy in the codes, which is done by making the input and hence the output distribution uniform, so their entropy is 1 bit. Since entropy is non-negative, $C \leq 1$. The channel capacity is 1 bit only if the entropy is zero, that is if $p = 0$ and the channel reliably transmits a 0 as 0 and a 1 as 1, or if $p = 1$ and it always flips bits. A completely noisy binary channel which transmits both 0s and 1s with equal probability as 0s and 1s (i.e., $p = \frac{1}{2}$) has capacity $C = 0$, since in this case there is no mutual information between X and Y. Such a channel is useless for communication.

It was one of the early triumphs of information theory that Shannon was able to show two important properties of channels. First, channel capacity is a well-defined notion. In other words, for each channel there is a smallest upper bound of $I(X;Y)$ over possible distributions p(X). Second, in many practical applications it is easy to get close to the optimal channel capacity. We can design a code appropriate for the channel that will transmit information at a rate that is optimal or very close to optimal. The concept of capacity eliminates a good part of the guesswork that was involved in designing communications systems before Shannon. One can precisely evaluate how good a code is for a communication line and design systems with optimal or near-optimal performance.

The noisy channel model is important in Statistical NLP because a simplified version of it was at the heart of the renaissance of quantitative natural language processing in the 1970s. In the first large quantitative project after the early quantitative NLP work in the 1950s and 60s, researchers at IBM's T. J. Watson research center cast both speech recognition and machine translation as a noisy channel problem.

Doing linguistics via the noisy channel model, we do not get to control the encoding phase. We simply want to decode the output to give the most likely input, and so we work with the channel shown in figure 2.9. Many problems in NLP can be construed as an attempt to determine the most likely input given a certain output. We can determine

Application	Input	output	$p(i)$	$p(o\|i)$
Machine Translation	L_1 word sequences	L_2 word sequences	$p(L_1)$ *in a* language model	translation model
Optical Character Recognition (OCR)	actual text	text with mistakes	prob of language text	model of OCR errors
Part Of Speech (POS) tagging	POS tag sequences	English words	prob of POS sequences	$p(w\|t)$
Speech recognition	word sequences	speech signal	prob of word sequences	acoustic model

Table 2.2 Statistical NLP problems as decoding problems.

this as follows, by using Bayes' theorem, and then noting that the output probability is a constant:

$$\hat{I} = \arg\max_i p(i|o) = \arg\max_i \frac{p(i)\,p(o|i)}{p(o)} = \arg\max_i p(i)\,p(o|i) \tag{2.40}$$

LANGUAGE MODEL

Here we have two probability distributions to consider: p(i) is the language ***model,*** the distribution of sequences of 'words' in the input language, and $p(o|i)$ is the channel probability.

CHANNEL PROBABILITY

As an example, suppose we want to translate a text from English to French. The noisy channel model for translation assumes that the true text is in French, but that, unfortunately, when it was transmitted to us, it went through a noisy communication channel and came out as English. So the word cow we see in the text was really ***vuche,*** garbled by the noisy channel to cow. All we need to do in order to translate is to recover the original French – or to ***decode*** the English to get the French.[7]

DECODE

The validity of the noisy channel model for translation is still giving rise to many a heated debate among NLP researchers, but there is no doubt that it is an elegant mathematical framework that has inspired a significant amount of important research. We will discuss the model in more detail in chapter 13. Other problems in Statistical NLP can also be seen as instantiations of the decoding problem. A selection is shown in table 2.2.

7. The French reader may be sympathetic with the view that English is really a form of garbled French that makes the language of *clarté* unnecessarily ambiguous!

2.2.5 Relative entropy or Kullback-Leibler divergence

RELATIVE ENTROPY

For two probability mass functions, p(x), q(x) their relative *entropy* is given by:

$$D(\mathrm{p} \,\|\, \mathrm{q}) = \sum_{x \in \mathcal{X}} \mathrm{p}(x) \log \frac{\mathrm{p}(x)}{\mathrm{q}(x)} \tag{2.41}$$

KULLBACK-LEIBLER DIVERGENCE

where again we define $0 \log \frac{0}{\mathrm{q}} = 0$ and otherwise $\mathrm{p} \log \frac{\mathrm{p}}{0} = \infty$. The relative entropy, also known as the *Kullback-Leibler* divergence, is a measure of how different two probability distributions (over the same event space) are. Expressed as an expectation, we have:

$$D(\mathrm{p} \,\|\, \mathrm{q}) = E_{\mathrm{p}} \left(\log \frac{\mathrm{p}(X)}{\mathrm{q}(X)} \right) \tag{2.42}$$

Thus, the KL divergence between p and q is the average number of bits that are wasted by encoding events from a distribution p with a code based on a not-quite-right distribution q.

This quantity is always non-negative, and $D(\mathrm{p} \| \mathrm{q}) = 0$ iff p = q. For these reasons, some authors use the name 'KL distance,' but note that relative entropy is not a metric (in the sense in which the term is used in mathematics): it is not symmetric in p and q (see exercise 2.12), and

TRIANGLE INEQUALITY

it does not satisfy the *triangle inequality*.[8] Hence we will use the name 'KL divergence,' but nevertheless, informally, people often think about the relative entropy as the 'distance' between two probability distributions: it gives us a measure of how close two pmfs are.

Mutual information is actually just a measure of how far a joint distribution is from independence:

$$I(X;Y) = D(\mathrm{p}(x,y) \| \mathrm{p}(x)\,\mathrm{p}(y)) \tag{2.43}$$

We can also derive conditional relative entropy and a chain rule for relative entropy (Cover and Thomas 1991: 23):

$$D(\mathrm{p}(y|x) \,\|\, \mathrm{q}(y|x)) = \sum_{x} \mathrm{p}(x) \sum_{y} \mathrm{p}(y|x) \log \frac{\mathrm{p}(y|x)}{\mathrm{q}(y|x)} \tag{2.44}$$

$$D(\mathrm{p}(x,y) \| \mathrm{q}(x,y)) = D(\mathrm{p}(x) \| \mathrm{q}(x)) + D(\mathrm{p}(y|x) \| \mathrm{q}(y|x)) \tag{2.45}$$

8. The triangle inequality is that for any three points x, y, z:

$$d(x,y) \leq d(x,z) + d(z,y)$$

▼ KL divergence is used for measuring selectional preferences in section 8.4.

2.2.6 The relation to language: Cross entropy

So far we have examined the notion of entropy, and seen roughly how it is a guide to determining efficient codes for sending messages, but how does this relate to understanding language? The secret to this is to return to the idea that entropy is a measure of our uncertainty. The more we know about something, the lower the entropy will be because we are less surprised by the outcome of a trial.

We can illustrate this with the examples used above. Consider again Simplified Polynesian from examples 8 and 9. This language has 6 letters. The simplest code is to use 3 bits for each letter of the language. This is equivalent to assuming that a good model of the language (where our 'model' is simply a probability distribution) is a uniform model. However, we noticed that not all the letters occurred equally often, and, noting these frequencies, produced a zeroth order model of the language. This had a lower entropy of 2.5 bits per letter (and we showed how this observation could be used to produce a more efficient code for transmitting the language). Thereafter, we noticed the syllable structure of the language, and developed an even better model that incorporated that syllable structure into it. The resulting model had an even lower entropy of 1.22 bits per letter. The essential point here is that if a model captures more of the structure of a language, then the entropy of the model should be lower. In other words, we can use entropy as a measure of the quality of our models.

Alternately, we can think of entropy as a matter of how surprised we will be. Suppose that we are trying to predict the next word in a Simplified Polynesian text. That is, we are examining $P(w|h)$, where w is the next word and h is the history of words seen so far. A measure of our surprise on seeing the next word can be derived in terms of the conditional probability assigned to w by our model m of the distribution of Simplified Polynesian words. Surprise can be measured by what we might term the *pointwise entropy* $H(w|h) = -log\, \mathrm{m}(w|h)$. If the predictor is certain that word w follows a given history h and it is correct, then the information supplied to the predictor on seeing w is $-\log 1 = 0$. In other words, the predictor does not experience any surprise at all. On the other hand, if the model thinks that w cannot follow h, then $\mathrm{m}(w|h) = 0$ and

SURPRISE

POINTWISE ENTROPY

so the information supplied to the predictor is infinite ($-\log 0 = \infty$). In this case our model is infinitely surprised, which is normally a very bad thing. Usually our models will predict a probability between these two extremes for each event and so the model will gain some information, or alternatively, be somewhat surprised, when it sees the next word, and the goal is to keep that level of surprise as low as possible. Summing over the surprise of the predictor at each word gives an expression for our total surprise:

$$\begin{aligned} H_{\text{total}} &= -\sum_{j=1}^{n} \log_2 \mathrm{m}(w_j | w_1, w_2, \ldots, w_{j-1}) \\ &= -\log_2 \mathrm{m}(w_1, w_2, \ldots, w_n) \end{aligned}$$

The second line above follows from the chain rule. Normally, we would want to normalize this measure by the length of the text so our notion of surprise is not dependent on the size of the text. This normalized measure gives the average surprise of the predictor per word.

So far this discussion has been rather informal, but we can formalize it through the notion of relative entropy. Suppose that we have some empirical phenomenon, in Statistical NLP usually utterances in a certain language. Assuming some mapping to numbers, we can represent it via a random variable X. Then we assume that there is some probability distribution over the utterances – for instance, you hear ***Thank you*** much more often than ***On you. So we*** take $X \sim p(x)$.

Now, unfortunately we do not know what $p(\cdot)$ is for empirical phenomena. But by looking at instances, for example by looking at a corpus of utterances, we can estimate roughly what p seems to be like. In other words, we can produce a model m of the real distribution, based on our best estimates. In making this model, what we want to do is to minimize $D(\mathrm{p} \,\|\, \mathrm{m})$ – to have as accurate a probabilistic model as possible. Unfortunately, we normally cannot calculate this relative entropy – again, because we do not know what p is. However, there is a related quantity, the cross entropy, which we fortunately can get a handle on.

CROSS ENTROPY

The ***cross entropy*** between a random variable X with true probability distribution p(x) and another pmf q (normally a model of p) is given by:

$$\begin{aligned} H(X, \mathrm{q}) &= H(X) + D(\mathrm{p} \,\|\, \mathrm{q}) \\ &= -\sum_{x} \mathrm{p}(x) \log \mathrm{q}(x) \end{aligned} \tag{2.46}$$

(2.47) $$= E_p\left(\log \frac{1}{q(x)}\right)$$

(Proof of this is left to the reader as exercise 2.13.)

Just as we defined the entropy of a language in section 2.2.2, we can define the cross entropy of a language $L = (Xi) \sim p(x)$ according to a model m by:

(2.48) $$H(L, m) = -\lim_{n\to\infty} \frac{1}{n} \sum_{x_{1n}} p(x_{1n}) \log m(x_{1n})$$

We do not seem to be making much progress, because it still seems that we cannot calculate this quantity without knowing p. But if we make certain assumptions that the language is 'nice,' then the cross entropy for the language can be calculated as:

(2.49) $$H(L, m) = -\lim_{n\to\infty} \frac{1}{n} \log m(x_{1n})$$

Using this second form, we can calculate the cross entropy based only on knowing our probability model and having a large body of utterances available. That is, we do not actually attempt to calculate the limit, but approximate it by calculating for a sufficiently large n:

(2.50) $$H(L, m) \approx -\frac{1}{n} \log m(x_{1n})$$

This measure is just the figure for our average surprise. Our goal will be to try to minimize this number. Because H(X) is fixed (if unknown), this is equivalent to minimizing the relative entropy, which is a measure of how much our probability distribution departs from actual language use. The only additional requirement is that the text that we use to test the model must be an independent test set, and not part of the training corpus that we used to estimate the parameters of the model. Cross entropy is inversely related to the average probability a model assigns to words in test data. Lower model cross entropy normally leads to better performance in applications, but it need not do so if it is just a matter of improving the magnitude of probability estimates, but not their relative ordering. (See section 6.2.3 for more practical details on calculating the cross entropy of models.)

But what justifies going from equation (2.48) to equation (2.49)? The formula for language cross entropy has an expectation embedded within it:

(2.51) $$H(L, m) = \lim_{n\to\infty} \frac{1}{n} E\left(\log \frac{1}{m(X_{1n})}\right)$$

Recall that the expectation is a weighted average over all possible sequences. But in the above formula we are using a limit and looking at longer and longer sequences of language use. Intuitively, the idea is then that if we have seen a huge amount of the language, what we have seen is 'typical.' We no longer need to average over all samples of the language; the value for the entropy rate given by this particular sample will be roughly right.

The formal version of this is to say that if we assume that $L = (\mathrm{Xi})$ is a stationary ergodic process, then we can prove the above result. This is a consequence of the Shannon-McMillan-Breiman theorem, also known as the Asymptotic Equipartition Property:

Theorem: If H_{rate} is the entropy rate of a finite-valued stationary ergodic process (X_n), then:

$$-\frac{1}{n}\log \mathrm{p}(X_1, \ldots, X_n) \rightarrow H, \quad \text{with probability 1}$$

We will not prove this theorem; see Cover and Thomas (1991: ch. 3, 15).

ERGODIC An *ergodic* process is one that, roughly, cannot get into different substates that it will not escape from. An example of a non-ergodic process is one that in the beginning chooses one of two states: one in which it generates 0 forever, one in which it generates 1 forever. If a process is not ergodic, then even looking at one very long sequence will not necessarily tell us what its typical behavior is (for example, what is likely to happen when it gets restarted).

STATIONARY A stationary process is one that does not change over time. This is clearly wrong for language: new expressions regularly enter the language while others die out. And so, it is not exactly correct to use this result to allow the calculation of a value for cross entropy for language applications. Nevertheless, for a snapshot of text from a certain period (such as one year's newswire), we can assume that the language is near enough to unchanging, and so this is an acceptable approximation to truth. At any rate, this is the method regularly used.

2.2.7 The entropy of English

As noted above, English in general is not a stationary ergodic process. But we can nevertheless model it with various stochastic approximations. In

n-GRAM MODELS particular, we can model English with what are known as ***n-gram models***

MARKOVCHAINS

MARKOVASSUMPTION

or *Markov chains.* These models, which we discuss in detail in chapters 6 and 9, are ones where we assume a limited memory. We assume that the probability of the next word depends only on the previous k words in the input. This gives a k^{th} order *Markov* approximation:

$$P(X_n = x_n | X_{n-1} = x_{n-1}, \ldots, X_1 = x_1) = P(X_n = x_n | X_{n-1} = x_{n-1}, \ldots, X_{n-k} = x_{n-k})$$

If we are working on a character basis, for example, we are trying to guess what the next character in a text will be given the preceding k characters. Because of the redundancy of English, this is normally fairly easy. For instance, a generation of students have proved this by being able to make do with photocopies of articles that are missing the last character or two of every line.

By adding up counts of letters, letter digraphs (that is, sequences of two letters), and so on in English, one can produce upper bounds for the entropy of English." We assume some such simplified model of English and compute its cross entropy against a text and this gives us an upper bound for the true entropy of English – since $D(\mathrm{p}||\mathrm{m}) \geq 0$, $H(X, \mathrm{m}) \geq \mathrm{H(X)}$. Shannon did this, assuming that English consisted of just 27 symbols (the 26 letters of the alphabet and SPACE – he ignored case distinctions and punctuation). The estimates he derived were:

(2.52)

Model	Cross entropy (bits)	
zeroth order	4.76	(uniform model, so log 2 7)
first order	4.03	
second order	2.8	
Shannon's experiment	1.3 (1.34)	(Cover and Thomas 1991: 140)

The first three lines show that as the order of the model increases, that is, as information about the frequencies of letters (first order) and digraphs (second order) is used, our model of English improves and the calculated cross entropy drops. Shannon wanted a tighter upper bound on the entropy of English, and derived one by human experiments – finding out how good at guessing the next letter in a text a human being was. This gave a much lower entropy bound for English. (A later experiment with

9. More strictly, one produces an estimate for the text on which the counts are based, and these counts are good for 'English only to the extent that the text used is representative of English as a whole. Working at the character level, this is not too severe a problem, but it becomes quite important when working at the word level, as discussed in chapter 4.

more subjects on the same text that Shannon used produced the figure in parentheses, 1.34.)

Of course, the real entropy of English must be lower still: there are doubtless patterns in people's speech that humans do not pick up on (although maybe not that many!). But at present, the statistical language models that we can construct are much worse than human beings, and so the current goal is to produce models that are as good as English speakers at knowing which English utterances sound normal or common and which sound abnormal or marked.

▼ We return to n-gram models in chapter 6.

2.2.8 Perplexity

PERPLEXITY In the speech recognition community, people tend to refer to ***perplexity*** rather than cross entropy. The relationship between the two is simple:

$$\text{perplexity}(x_{1n}, \text{m}) = 2^{H(x_{1n},\text{m})} \tag{2.53}$$

$$= \text{m}(x_{1n})^{\frac{1}{n}} \tag{2.54}$$

We suspect that speech recognition people prefer to report the larger non-logarithmic numbers given by perplexity mainly because it is much easier to impress funding bodies by saying that "we've managed to reduce perplexity from 950 to only 540" than by saying that "we've reduced cross entropy from 9.9 to 9.1 bits." However, perplexity does also have an intuitive reading: a perplexity of k means that you are as surprised on average as you would have been if you had had to guess between k equiprobable choices at each step.

2.2.9 Exercises

Exercise 2.9 [⋆]

Take a (short) piece of text and compute the relative frequencies of the letters in the text. Assume these are the true probabilities. What is the entropy of this distribution?

Exercise 2.10 [⋆]

Take another piece of text and compute a second probability distribution over letters by the same method. What is the KL divergence between the two distributions? (You will need to 'smooth' the second distribution and replace any zero with a small quantity E.)