The Source Coding Theorem

Mário S. Alvim
(msalvim@dcc.ufmg.br)

Information Theory

DCC-UFMG
(2017/02)

The Source Coding Theorem - Introduction

- In this lecture we will define sensible measures of

  1. the information content of a random event, and
  2. the expected information content of a random experiment.

- We will also see how the measures above are connected with the compression of sources of information.

Definition of entropy and related functions

Ensembles

- Recall that an **ensemble** X is a triple $(x, \mathcal{A}_X, \mathcal{P}_X)$, where:
  - x is the outcome of a random variable,
  - $\mathcal{A}_X = \{a_1, a_2, \ldots, a_i, \ldots, a_I\}$ is the set of possible values for the random variable, and
  - $\mathcal{P}_X = \{p_1, p_2, \ldots, p_I\}$ are the probabilities of each value, with p_i standing for $p(x = a_i)$.

Introduction to Shannon entropy

- The **Shannon information content of an outcome** x is defined to be

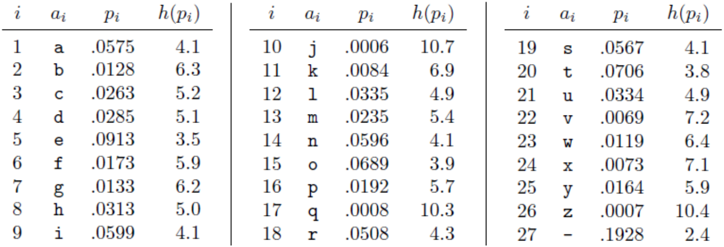
$$h(x) = \log_2 \frac{1}{p(x)},$$

  and it is measured in **bits**.

- **Convention.** From now on, $\log x$ stands for $\log_2 x$, unless otherwise stated.

Introduction to Shannon entropy

- Example 1 Frequency of letters in *"The Frequently Asked Questions Manual for Linux"*, and their entropy.



i	a_i	p_i	$h(p_i)$	i	a_i	p_i	$h(p_i)$	i	a_i	p_i	$h(p_i)$
1	a	.0575	4.1	10	j	.0006	10.7	19	s	.0567	4.1
2	b	.0128	6.3	11	k	.0084	6.9	20	t	.0706	3.8
3	c	.0263	5.2	12	l	.0335	4.9	21	u	.0334	4.9
4	d	.0285	5.1	13	m	.0235	5.4	22	v	.0069	7.2
5	e	.0913	3.5	14	n	.0596	4.1	23	w	.0119	6.4
6	f	.0173	5.9	15	o	.0689	3.9	24	x	.0073	7.1
7	g	.0133	6.2	16	p	.0192	5.7	25	y	.0164	5.9
8	h	.0313	5.0	17	q	.0008	10.3	26	z	.0007	10.4
9	i	.0599	4.1	18	r	.0508	4.3	27	-	.1928	2.4

Introduction to Shannon entropy

- The **entropy of an ensemble** X is defined to be the average Shannon information content of an outcome:

  $$H(X) = \sum_{x \in \mathcal{A}_X} p(x) \log_2 \frac{1}{p(x)},$$

  with the convention that for $p(x) = 0$ we have

  $$0 \cdot \log_2 1/0 = 0,$$

  since $\lim_{\theta \to 0} \theta \log_2 1/\theta = 0$.

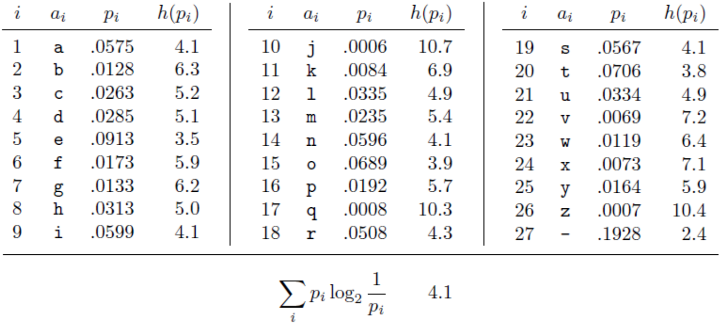
  When it is convenient we may write $H(X)$ as $H(\mathbf{p})$, where $\mathbf{p}$ is the vector $(p_1, p_2, \ldots, p_I)$.

  Another name for the entropy of X is the **uncertainty of** X.

Introduction to Shannon entropy

- Example 1 (Continued)

Frequency of letters in *"The Frequently Asked Questions Manual for Linux"*.



i	a_i	p_i	$h(p_i)$	i	a_i	p_i	$h(p_i)$	i	a_i	p_i	$h(p_i)$
1	a	.0575	4.1	10	j	.0006	10.7	19	s	.0567	4.1
2	b	.0128	6.3	11	k	.0084	6.9	20	t	.0706	3.8
3	c	.0263	5.2	12	l	.0335	4.9	21	u	.0334	4.9
4	d	.0285	5.1	13	m	.0235	5.4	22	v	.0069	7.2
5	e	.0913	3.5	14	n	.0596	4.1	23	w	.0119	6.4
6	f	.0173	5.9	15	o	.0689	3.9	24	x	.0073	7.1
7	g	.0133	6.2	16	p	.0192	5.7	25	y	.0164	5.9
8	h	.0313	5.0	17	q	.0008	10.3	26	z	.0007	10.4
9	i	.0599	4.1	18	r	.0508	4.3	27	-	.1928	2.4

$$\sum_i p_i \log_2 \frac{1}{p_i} \qquad 4.1$$

◁

Introduction to Shannon entropy

- Some properties of the entropy function:

  1. Entropy is always non-negative:

  $$H(X) \geq 0,$$

  with equality iff $p_i = 1$ for one i.

  2. Entropy is maximized when $\mathbf{p}$ is uniform:

  $$H(X) \leq \log_2 |\mathcal{A}_X|,$$

  with equality iff $p_i = 1/|\mathcal{A}_X|$ for all i.

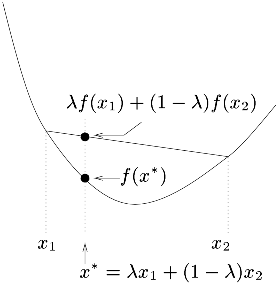
- Verifying these properties is part of your homework assignment: for that you'll need Jensen's inequality.

Jensen's inequality and convex functions

- A function $f(x)$ is **convex** $\smile$ over an interval (a, b) if every chord of the function lies above the function.

  That is, if for all $0 \leq \lambda \leq 1$:

  $$f(\lambda x_1 + (1-\lambda)x_2) \;\leq\; \lambda f(x_1) + (1-\lambda) f(x_2).$$



$\lambda f(x_1) + (1-\lambda) f(x_2)$
$f(x^*)$
x_1
x_2
$x^* = \lambda x_1 + (1-\lambda) x_2$

- **Jensen's inequality:** if f is a convex function and x is a random variable, then

  $$E[f(x)] \quad \geq \quad f(E[x]),$$

  where $E[\cdot]$ denotes expected value.

Introduction to Shannon entropy

- **The joint entropy of** X, Y is

$$H(X,Y) = \sum_{xy \in \mathcal{A}_X \mathcal{A}_Y} p(x,y) \log_2 \frac{1}{p(x,y)}.$$

Introduction to Shannon entropy

- **Theorem** Entropy is additive for independent random variables:

$$H(X,Y) = H(X) + H(Y),$$

iff $p(x,y) = p(x)p(y)$.

Proof.

We start by proving the following auxiliary result for when $p(x,y) = p(x)p(y)$:

$$\begin{aligned}
h(x,y) &= \log \frac{1}{p(x,y)} && \text{(by def. of } h(\cdot)\text{)} \\
&= \log \frac{1}{p(x)p(y)} && (p(x,y) = p(x)p(y)) \\
&= \log \left(\frac{1}{p(x)} \cdot \frac{1}{p(y)} \right) \\
&= \log \frac{1}{p(x)} + \log \frac{1}{p(y)} \\
&= h(x) + h(y) && \text{(by def. of } h(\cdot)\text{)}
\end{aligned}$$

Introduction to Shannon entropy

- **Proof.** (Continued)

  Then, we can show that:

$$
\begin{aligned}
H(X,Y) &= \sum_{x \in \mathcal{A}_X} \sum_{y \in \mathcal{A}_Y} p(x,y)h(x,y) && \text{(by definition)} \\
&= \sum_{x \in \mathcal{A}_X} \sum_{y \in \mathcal{A}_Y} p(x)p(y)\left[h(x)+h(y)\right] && \text{(x, y independent)} \\
&= \sum_{x \in \mathcal{A}_X} \sum_{y \in \mathcal{A}_Y} \left[p(x)p(y)h(x)+p(x)p(y)h(y)\right] && \text{(by distributivity)} \\
&= \sum_{x \in \mathcal{A}_X} \sum_{y \in \mathcal{A}_Y} p(x)p(y)h(x) + \sum_{x \in \mathcal{A}_X} \sum_{y \in \mathcal{A}_Y} p(x)p(y)h(y) && \text{(splitting the sums (*))}
\end{aligned}
$$

Joint Entropy - Properties

- **Proof.** (Continued)

  Note that the first term in the sum in Equation (*) can be written as

$$
\begin{aligned}
\sum_{x \in \mathcal{A}_X} \sum_{y \in \mathcal{A}_Y} p(x)p(y)h(x) &= \sum_{x \in \mathcal{A}_X} p(x)h(x) \sum_{y \in \mathcal{A}_Y} p(y) && \text{(moving out constants)} \\
&= \sum_{x \in \mathcal{A}_X} p(x)h(x) \cdot 1 && (\textstyle\sum_{y \in \mathcal{A}_Y} p(y) = 1) \\
&= H(X) && \text{(by definition (**))},
\end{aligned}
$$

  and the second term in the sum in Equation (*) can be written as

$$
\begin{aligned}
\sum_{x \in \mathcal{A}_X} \sum_{y \in \mathcal{A}_Y} p(x)p(y)h(x) &= \sum_{y \in \mathcal{A}_Y} p(y)h(y) \sum_{x \in \mathcal{A}_X} p(x) && \text{(moving out constants)} \\
&= \sum_{y \in \mathcal{A}_Y} p(y)h(y) \cdot 1 && (\textstyle\sum_{x \in \mathcal{A}_X} p(x) = 1) \\
&= H(Y) && \text{(by definition (***))}.
\end{aligned}
$$

Joint Entropy - Properties

- **Proof.** (Continued)

  Now we can substitute Equations (**) and (***) in Equation (*) to obtain

  $$H(X, Y) = H(X) + H(Y).$$

  The proof of the converse, i.e., that if $H(X, Y) = H(X) + H(Y)$ then X and Y are independent, is similar and is left as an exercise.

  □

The Source Coding Theorem

Ensembles

- Recall that an **ensemble** X is a triple $(x, \mathcal{A}_X, \mathcal{P}_X)$, where:
  - x is the outcome of a random variable,
  - $\mathcal{A}_X = \{a_1, a_2, \ldots, a_i, \ldots, a_I\}$ is the set of possible values for the random variable, and
  - $\mathcal{P}_X = \{p_1, p_2, \ldots, p_I\}$ are the probabilities of each value, with p_i standing for $p(x = a_i)$.

The Source Coding Theorem - Three claims

1. The **Shannon information content** (a.k.a. **surprisal**, or **self-information**)

   $$h(x = a_i) = \log_2 \frac{1}{p(x = a_i)}$$

   is a sensible measure of the information content of the outcome $x = a_i$.

2. The **entropy**

   $$H(X) = \sum_{x \in \mathcal{A}_X} p(x) \log_2 \frac{1}{p(x)}$$

   is a sensible measure of the expected information content of an ensemble X.

3. **Source coding theorem:** N outcomes from a source X can be compressed into roughly $N \cdot H(X)$ bits.

The Shannon information content of an outcome

- Our first claim is that the Shannon information content

$$h(x = a_i) = \log_2 \frac{1}{p(x = a_i)}$$

is a sensible measure of the information content of the outcome $x = a_i$.

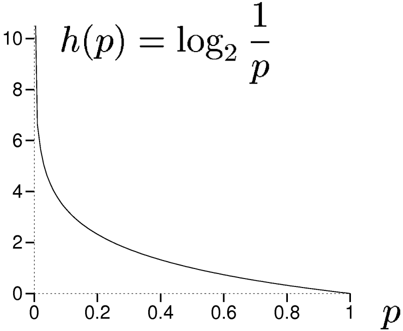
The Shannon information content of an outcome

- Some intuitive support for our first claim:

  a) The less probable an outcome is, the more informative is its happening.

  Or, the more "surprising" an outcome is, the more informative it is.

  The function $h(x)$ captures this intuition, since it grows as the probability of x diminishes:



p	$h(p)$
0.001	10.0
0.01	6.6
0.1	3.3
0.2	2.3
0.5	1.0

The Shannon information content of an outcome

- Some intuitive support for our first claim:

  b) If an outcome x happens with certainty (i.e., $p(x) = 1$), the occurrence of this outcome conveys no information:

  $$h(x) = \log_2 \frac{1}{p(x)} = \log_2 \frac{1}{1} = 0.$$

  If an outcome x happens is impossible (i.e., $p(x) = 0$), the occurrence of this outcome conveys an infinite amount of information:

  $$h(x) = \log_2 \frac{1}{p(x)} = \log_2 \frac{1}{0} = \infty.$$

The Shannon information content of an outcome

- Some intuitive support for our first claim:

  c) Independent events add up their surprises.

  If $p(x, y) = p(x)p(y)$ then

$$\begin{aligned} h(x, y) &= \log_2 \frac{1}{p(x, y)} \\ &= \log_2 \left(\frac{1}{p(x)} \cdot \frac{1}{p(y)} \right) \\ &= \log_2 \frac{1}{p(x)} + \log_2 \frac{1}{p(y)} \\ &= h(x) + h(y). \end{aligned}$$

The entropy of an ensemble

- Our second claim is that the entropy

  $$H(X) = \sum_{x \in \mathcal{A}_X} p(x) \log_2 \frac{1}{p(x)}$$

  is a sensible measure of the expected information content of an ensemble $X = (x, \mathcal{A}_X, \mathcal{P}_X)$.

The entropy of an ensemble

- Some intuitive support for our second claim:

  a) (**The weighing problem.**) You are given 12 balls, all equal in weight except for one that is either heavier or lighter.

     You are also given a two-pan balance to use.

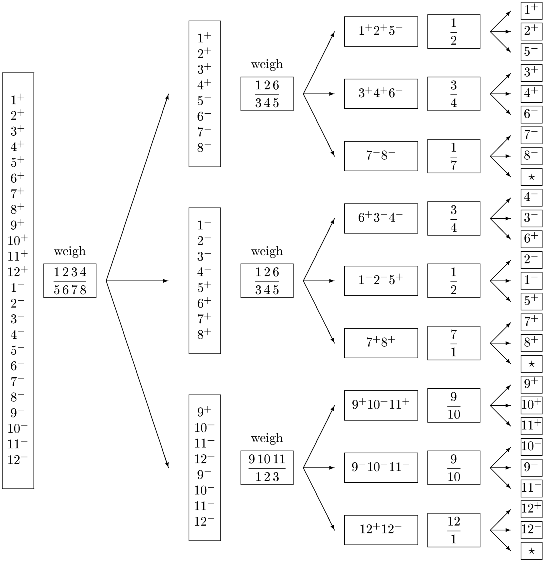
     In each use of the balance you may put any number of the 12 balls on the left pan, and the same number on the right pan, and push a button to initiate the weighing; there are three possible outcomes:

     1. either the weights are equal; or
     2. the balls on the left are heavier; or
     3. the balls on the left are lighter.

     Your task is to design a strategy to determine which is the odd ball and whether it is heavier or lighter than the others in as few uses of the balance as possible.

The entropy of an ensemble

- A possible optimal solution for the weighing problem.



1^+ 2^+ 3^+ 4^+ 5^+ 6^+ 7^+ 8^+ 9^+ 10^+ 11^+ 12^+ 1^- 2^- 3^- 4^- 5^- 6^- 7^- 8^- 9^- 10^- 11^- 12^-

weigh $\frac{1\,2\,3\,4}{5\,6\,7\,8}$

1^+ 2^+ 3^+ 4^+ 5^- 6^- 7^- 8^-

weigh $\frac{1\,2\,6}{3\,4\,5}$

$1^+2^+5^-$ $\frac{1}{2}$ 1^+ 2^+ 5^-

$3^+4^+6^-$ $\frac{3}{4}$ 3^+ 4^+ 6^-

7^-8^- $\frac{1}{7}$ 7^- 8^- $\star$

1^- 2^- 3^- 4^- 5^+ 6^+ 7^+ 8^+

weigh $\frac{1\,2\,6}{3\,4\,5}$

$6^+3^-4^-$ $\frac{3}{4}$ 4^- 3^- 6^+

$1^-2^-5^+$ $\frac{1}{2}$ 2^- 1^- 5^+

7^+8^+ $\frac{7}{1}$ 7^+ 8^+ $\star$

9^+ 10^+ 11^+ 12^+ 9^- 10^- 11^- 12^-

weigh $\frac{9\,10\,11}{1\,2\,3}$

$9^+10^+11^+$ $\frac{9}{10}$ 9^+ 10^+ 11^+

$9^-10^-11^-$ $\frac{9}{10}$ 10^- 9^- 11^-

12^+12^- $\frac{12}{1}$ 12^+ 12^- $\star$

The entropy of an ensemble

- Insights on “information” gained from the weighing problem:

  (i) The world may be in many different states, and you are uncertain about which is the real one.

  (ii) You have measurements (questions) that you can make (ask) to probe in what state the world is.

  (iii) Each measure (question) produces an observation (answer) that allows you to rule out some states of the world as not possible.

  (iv) At each time a subset of possible states is ruled out, you gain some information about the real state of the world.

  The information you have increases because your uncertainty about the real state of the world decreases.

The entropy of an ensemble

- Insights on "information" gained from the weighing problem:

  (v) The most efficient way of finding the actual state is to have every measurement (question) outcomes as close as possible to equally probable.

  If your measurement (question) allows for n different outcomes (types of answers), it is best to use them so to always split the set of still possible states of the world into n sets of probability $1/n$ each.

  (vi) The Shannon information content (in base 3) of the set of balls is

$$H(X) = \sum_{i=1}^{24} \frac{1}{24} \log_3 \frac{1}{1/24} = \log_3 24 = 2.89,$$

  which is just about the minimal number of measurements (3) needed in a best strategy.

The entropy of an ensemble

- Some intuitive support for our second claim:

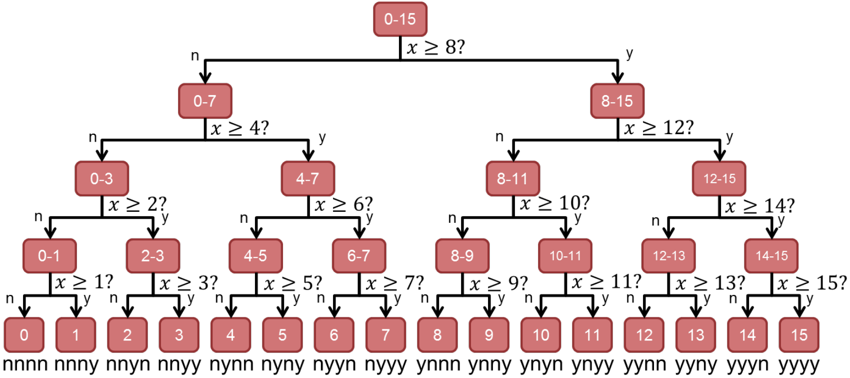
  b) (**The guessing game.**) Do you know the “20 questions” game? (If you don’t, Google it, it’s a fun game to while away the time with friends during a long trip.)

  In a dumber version of the game, a friend thinks of a number between 0 and 15 and you have to guess which number was selected using yes/no questions.

  What is the smallest number of questions needed to identify an integer between 0 and 15?

The entropy of an ensemble

- An optimal strategy of yes/no questions to identify an integer in 0-15:



The entropy of an ensemble

- Insights on "information" gained from the guessing game:

  (i) A series of answers to yes/no questions can be used to uniquely identify an object from a set.

  (That's why the questions are useful in the first place!)

  (ii) The optimal strategy to win the game corresponds to the shortest sequences of yes/no questions needed to identify objects in the set.

  (iii) If you map each yes/no answer to a 0/1 bit, you get a unique binary string that identifies each object in the set.

  (iv) Hence, the optimal strategy to win the game leads to the shortest binary description of objects in the set.

The entropy of an ensemble

- Insights on “information” gained from the guessing game:

  (v) Encoding information efficiently is related to asking the right questions.

  (vi) We saw that the number of yes/no questions needed to identify an integer between 0 and 15 is 4.

  Let us calculate the Shannon information of the set of integers between 0 and 15, assuming your friend can pick any number in the set with equal probability:

$$H(X) = \sum_{i=0}^{15} \frac{1}{16} \log_2 \frac{1}{1/16} = \log_2 16 = 4.$$

  Shannon entropy gave us the minimal number of questions necessary to win the game. Is this a coincidence?

The entropy of an ensemble

- Some intuitive support for our second claim:

  c) (**The game of submarine.**) In a boring version of the "game of battleships" called "game of submarine", each player hides just one submarine in one square of an eight-by-eight grid.

  At each round, the other player picks a square in the grid to shoot at.

  There two possible outcomes of a shot are $\mathtt{y}, \mathtt{n}$, corresponding to a hit and a miss, and their probabilities depend on the state of the board.

The entropy of an ensemble

- In the game of submarine, each shot made by a player defines an ensemble:
  - at the beginning:

    $$p(\mathtt{y}) = \frac{1}{64} \qquad \text{and} \qquad p(\mathtt{n}) = \frac{63}{64};$$

  - at the second shot, if the first shot missed:

    $$p(\mathtt{y} \mid \mathtt{n}) = \frac{1}{63} \qquad \text{and} \qquad p(\mathtt{n} \mid \mathtt{n}) = \frac{62}{63};$$

  - at the third shot, if the first two shots missed:

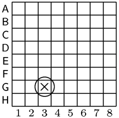
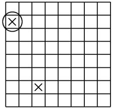
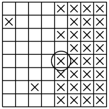
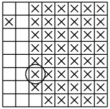
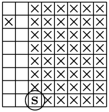
    $$p(\mathtt{y} \mid \mathtt{nn}) = \frac{1}{62} \qquad \text{and} \qquad p(\mathtt{n} \mid \mathtt{nn}) = \frac{61}{62};$$

  - ...
  - at the k^{th} shot (with $1 \leq k \leq 64$), if the first $k-1$ shots missed:

    $$p(\mathtt{y} \mid \mathtt{n}^{k-1}) = \frac{1}{(64-k+1)} \qquad \text{and} \qquad p(\mathtt{n} \mid \mathtt{n}^{k-1}) = \frac{(64-k)}{(64-k+1)}.$$

The entropy of an ensemble

- A game of submarine in which the submarine is hit on the 49^{th} attempt.

					
move #	1	2	32	48	49
question	G3	B1	E5	F3	H3
outcome	$x = \mathtt{n}$	$x = \mathtt{n}$	$x = \mathtt{n}$	$x = \mathtt{n}$	$x = \mathtt{y}$
$P(x)$	$\frac{63}{64}$	$\frac{62}{63}$	$\frac{32}{33}$	$\frac{16}{17}$	$\frac{1}{16}$
$h(x)$	0.0227	0.0230	0.0443	0.0874	4.0
Total info.	0.0227	0.0458	1.0	2.0	6.0

The entropy of an ensemble

- Insights on “information” gained from the game of submarine:

  (i) If we get a hit $\mathtt{y}$ on the k^{th} shot, the sequence of answers we get is a string

  $$x = \mathtt{n}^{k-1}\mathtt{y},$$

  in which the first $k-1$ symbols are $\mathtt{n}$ and the last one is $\mathtt{y}$.

  This string x is the outcome of the hit/miss experiment that uniquely identifies the square where the submarine is.

  For a fixed strategy, our game has 64 possible outcomes:

  - $\mathtt{y}$,
  - $\mathtt{ny}$,
  - $\mathtt{nny}$,
  - $\ldots$,
  - $\mathtt{nnnn}\ldots\mathtt{nny} = \mathtt{n}^{62}\mathtt{y}$,
  - $\mathtt{nnnn}\ldots\mathtt{nnny} = \mathtt{n}^{63}\mathtt{y}$.

The entropy of an ensemble

- Insights on "information" gained from the game of submarine:

  In particular, we can call $\mathtt{y}$ a bit 1 and $\mathtt{n}$ a bit 0, and encode each of the game's 64 possible outcomes as a binary string:

  - $\mathtt{y} = 1$ (which is a binary string of 1 bit),
  - $\mathtt{ny} = 01$ (which is a binary string of 2 bits)
  - $\mathtt{nny} = 001$ (which is a binary string of 3 bits),
  - $\ldots$,
  - $\mathtt{n}^{62}\mathtt{y} = 0^{62}1$ (which is a binary string of 63 bits),
  - $\mathtt{n}^{63}\mathtt{y} = 0^{63}1$ (which is a binary string of 64 bits).

- Note that the use of symbols $\{\mathtt{y}, \mathtt{n}\}$ or $\{0, 1\}$ makes little difference: each binary string uniquely identifies a result of the game.

  Note also that we have binary strings of many different sizes.

The entropy of an ensemble

- Insights on “information” gained from the game of submarine:

  (ii) Let us calculate the Shannon information content of an arbitrary string $x = \mathtt{n}^{k-1}\mathtt{y}$ for some k:

$$
\begin{aligned}
h(x = \mathtt{n}^{k-1}\mathtt{y}) &= \log_2 \frac{1}{p(x = \mathtt{n}^{k-1}\mathtt{y})} \\
&= \log_2 \frac{1}{\frac{63}{64} \cdot \frac{62}{63} \cdot \frac{61}{62} \cdot \ldots \cdot \frac{64-k+2}{64-k+3} \cdot \frac{64-k+1}{64-k+2} \cdot \frac{1}{64-k+1}} \\
&= \log_2 \frac{1}{1/64} \\
&= \log_2 64 \\
&= 6 \text{ bits.}
\end{aligned}
$$

The entropy of an ensemble

- Insights on “information” gained from the game of submarine:

  (iii) Every outcome $\mathtt{n}^{k-1}\mathtt{y}$, no matter k, conveys the same amount of information $h(\mathtt{n}^{k-1}\mathtt{y}) = 6$ bits.

  6 bits is exactly the number of bits necessary to uniquely identify a square in a set of 64!

  (iv) The information contained in a binary string representing an object is not necessarily the number of bits in the string: it is related to the object it identifies within a set!

  In our submarine example, all strings from $\mathtt{y}$ (which has size of 1 bit) to $\mathtt{n}^{63}\mathtt{y}$ (which has size of 64 bits) carry the same amount of bits of information: 6 bits each.

Entropy as the complexity of binary search

- Complementing the intuitions we got from the previous examples, in a future lecture we will be able to prove an operational interpretation of Shannon entropy in terms of search trees.

- The Shannon entropy $H(X)$ of a random variable distributed according to probability distribution p_X is the:

  - the expected number of comparisons needed
  - for an optimal binary-search algorithm
  - on a space of values $\mathcal{X}$ following distribution p_X.

- In this sense, a random variable with:

  - low entropy would be located "fast" using binary search, whereas
  - one with high entropy would need "more effort/time" to be located.

- Note that there could be other ways of measuring the information of a distribution: we'll disuss them at the end of this course.

Data compression

- Our third claim is that N outcomes from a source X can be compressed into roughly $N \cdot H(X)$ bits.

  In other words, our claim is that the number of bits necessary to compress a source is linear with respect to the entropy of the source.

- This claim implies an intimate connection between data compression and the measure of information content of the source.

  Before giving support for our third claim, let us understand better what we mean by "data compression".

Data compression

- A **source** of information is a stochastic process

$$X_1, X_2, X_3, \ldots$$

  in which the outcome of each ensemble X_i is a **symbol** produced by the source.

- Examples of sources of information include:

  1. the speech produced by a human (each word is a symbol),
  2. the sequence of pixels in a black and white image (each symbol is either "black" or "white"),
  3. the sequence of states of the weather in a region in a sequence of days (each symbol is "good", "cloudy", "rainy").

Data compression

- **Average information content per symbol of a source:**

  If we can show that we can compress data from a particular source into a file of L bits per source symbol and recover the data reliably, then we will say that the average information content of that source is at most L bits per symbol.

Data compression

- The **raw bit content** of an ensemble X is

  $$H_0(X) = \log_2 |\mathcal{A}_X|.$$

  H_0 represents the minimum number of bits necessary to give a unique **codeword** (i.e., a "name") to every element in the ensemble X.

- H_0 is a measure of information of X:

  it is a bound on the number of bits necessary to encode elements of X as codewords.

- This measure of information only considers the encoding of ensemble X, but it does not consider how the encoding for the ensemble can be compressed.

  To do compression, we need to take into consideration the probability of each outcome of the ensemble.

Data compression

- $\boxed{\text{Example 2}}$ (MacKay 4.5) Could there be a compressor that maps an outcome x to a binary code $c(x)$, and a decompressor that maps c back to x, such that every possible outcome is compressed into a binary code of length shorter than $H_0(X)$ bits?

  Solution. No! Just use the pigeonhole principle to verify that. (This exercise is part of your homework assignment for this lecture.)

  ◁

Data compression

- There are only two ways a compressor can compress files:

  1. A **lossy compressor** maps all files to shorter codewords, but that means that sometimes two or more files will necessarily be mapped to the same codeword.

     The decompressor will be, in this case, unsure of how to decompress ambiguous codewords, leading to a **failure**.

     Calling δ the probability that the source file is one of the confusable files, a lossy compressor has probability δ of failure.

     If δ is small, the compressor is acceptable, but with some loss of information (i.e., not all codewords are guaranteed to be decompressed correctly).

Data compression

- There are only two ways a compressor can compress files:

  2. A **lossless compressor** maps most files to shorter codewords, but it will necessarily map some files to longer codewords.

     By picking wisely which files to map to shorter codewords (i.e., the most probable files) and which files to map to longer codewords (i.e., the least probable files), the compressor can usually achieve satisfactory compression rates, and without any loss of information.

- In the remaining of this lecture we will cover a simple lossy compressor, and in future lectures we will cover lossless compressors.

Lossy data compression

- All compressors must take into consideration the probabilities of the different outcomes a source may produce.

- $\boxed{\text{Example 3}}$ Let

$$\mathcal{A}_X = \{\mathtt{a}, \mathtt{b}, \mathtt{c}, \mathtt{d}, \mathtt{e}\, \mathtt{f}, \mathtt{g}, \mathtt{h}\}$$
$$\mathcal{P}_X = \left\{\frac{1}{4}, \frac{1}{4}, \frac{1}{4}, \frac{3}{16}, \frac{1}{64}, \frac{1}{64}, \frac{1}{64}, \frac{1}{64}\right\}$$

The raw bit content of this ensemble is

$$H_0 = \log_2 |\mathcal{A}_X| = \log_2 8 = 3 \text{ bits},$$

so to represent any symbol of the ensemble we need, in principle, codewords of 3 bits each.

But notice that a small set contains almost all the probability:

$$p(x \in \{\mathtt{a}, \mathtt{b}, \mathtt{c}, \mathtt{d}\}) = \frac{15}{16}.$$

Lossy data compression

- Example 3 (Continued)

If we accept a risk $\delta = 1/16$ of not having a codeword for a symbol x, we can use an encoding using only 2 bits per symbol instead of 3:

$\delta = 0$	
x	$c(x)$
a	000
b	001
c	010
d	011
e	100
f	101
g	110
h	111

$\delta = 1/16$	
x	$c(x)$
a	00
b	01
c	10
d	11
e	—
f	—
g	—
h	—

Lossy data compression

- The above example can be generalized to principle below.

- **Principle of lossy data compression.**

  Let S_δ denote the **smallest** δ-**sufficient** set (that is a subset of $\mathcal{A}_X$) satisfying

  $$p(x \notin S_\delta) \leq \delta \qquad \text{or, equivalently,} \qquad p(x \in S_\delta) \geq 1 - \delta.$$

  The maximum compression tolerating a probability of error at most δ uses codewords of size $\log_2 |S_\delta|$ bits.

- The quantity

  $$H_\delta(X) = \log_2 |S_\delta|$$

  is called the **essential bit content** of X.

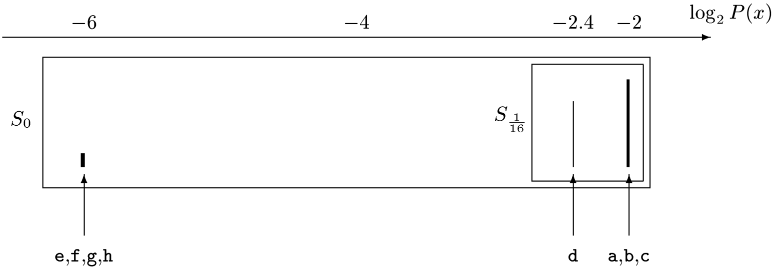
- If we are willing to accept a probability δ of error, we can compress the source from $H_0(X)$ bits per symbol to $H_\delta(X)$ bits per symbol.

Lossy data compression

- $\boxed{\text{Example 4}}$ For the lossy compressor where

$$\mathcal{A}_X = \{\texttt{a}, \texttt{b}, \texttt{c}, \texttt{d}, \texttt{e}\, \texttt{f}, \texttt{g}, \texttt{h}\}, \qquad \text{and}$$
$$\mathcal{P}_X = \left\{\frac{1}{4}, \frac{1}{4}, \frac{1}{4}, \frac{3}{16}, \frac{1}{64}, \frac{1}{64}, \frac{1}{64}, \frac{1}{64}\right\},$$

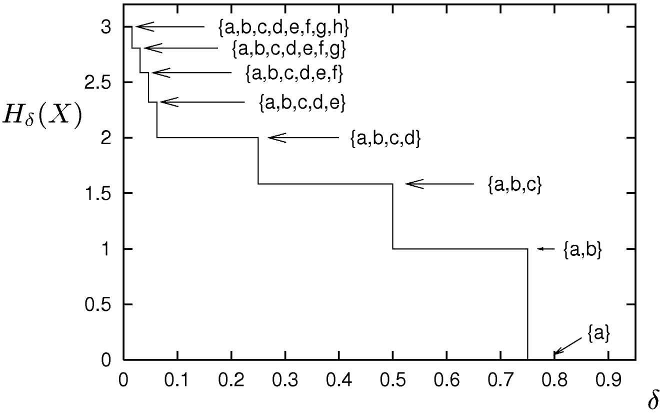
we have:



Lossy data compression

- Example 4 (Continued)

For this lossy compressor, we have:



Data compression of groups of symbols

- We can ask ourselves if we can do better compression if, instead of encoding each symbol of the source individually, we encode groups of symbols as blocks.

- Let's start by reasoning about the entropy of a group of symbols as a block.

- Let $x = (x_1, x_2, \ldots, x_N)$ be a string of N independent identically distributed (i.i.d.) random variables from a single ensemble X.

  Let X^N denote the ensemble $(X_1, X_2, \ldots, X_N)$.

  Because entropy is additive for independent variables, we have

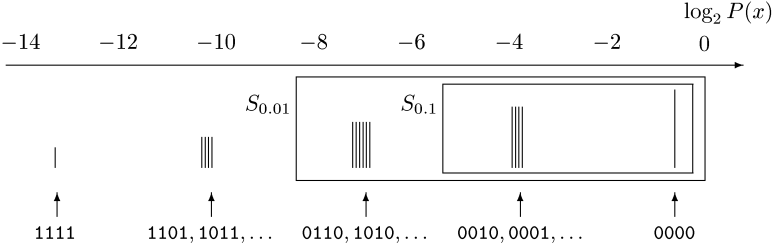
$$H(X^N) = N \cdot H(X).$$

Data compression of groups of symbols

- $\boxed{\text{Example 5}}$ Consider a string of N flips of a bent coins, $x = (x_1, x_2, \ldots, x_N)$, where $x_n \in \{0, 1\}$, with probabilities $p_0 = 0.9$ and $p_1 = 0.1$.

  If $r(x)$ is the number of 1s in x, then

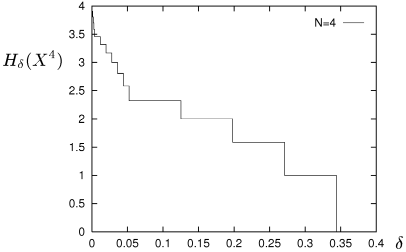
$$p(x) = p_0^{N-r(x)} p_1^{r(x)}.$$

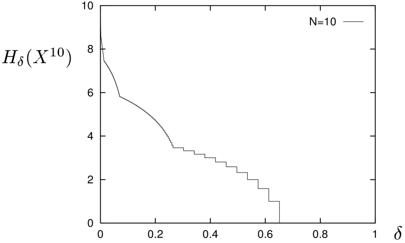


Data compression of groups of symbols

- Example 5 (Continued)

If we want to encode blocks of size N, we can make a graph of how the necessary number $H_\delta(X^N)$ of bits to encode the blocks varies as a function of the error δ we are willing to tolerate.





Data compression of groups of symbols

- $\boxed{\text{Example 5}}$ (Continued)

  To encode blocks of size N we need $H_\delta(X^N)$ bits per block of size N symbols.

  That means that the number of bits per symbol is

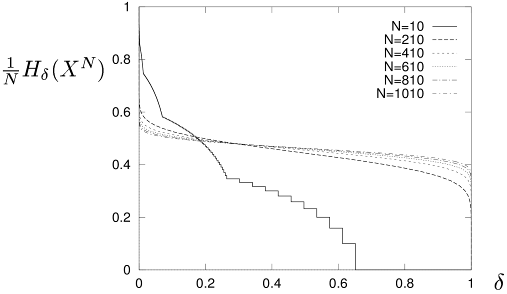
  $$\frac{1}{N} H_\delta(X^N).$$

  What happens as N grows?

Data compression of groups of symbols

- Example 5 (Continued)

As N grows we have the following graph:



It seems that as N grows, $\frac{1}{N}H_\delta(X^N)$ flattens out (i.e., becomes constant), independently of the value of the error δ tolerated.

What is the fixed value that $\frac{1}{N}H_\delta(X^N)$ tends to as N grows?

Shannon's source coding theorem tells us what it is...

The Source Coding Theorem

- **Shannon's source coding theorem.** Let X be an ensemble. Given $\epsilon > 0$ and $0 < \delta < 1$, there exists a positive integer N_0 such that for $N > N_0$,

$$H(X) - \epsilon \;<\; \frac{1}{N} H_\delta(X^N) \;<\; H(X) + \epsilon.$$

- In English, Shannon's source coding theorem states that if

  1. you are encoding N symbols of a source X, and
  2. you are willing to accept a probability δ of error in the decompression,

  then

  3. the maximum achievable compression will use approximately $H(X)$ bits per symbol if the number of N of symbols being encoded is large enough.

The Source Coding Theorem - Why it works

- To see why Shannon's coding theorem is true we will use the concept of a typical string generated by the source.
- A string of size N is **typical** if the frequency of each symbol in the string is the same as the probability of the symbol being produced by the source:

$$P(x)_{typ} \approx p_1^{p_1 N} p_2^{p_2 N} p_3^{p_3 N} \ldots p_I^{p_I N}.$$

The Source Coding Theorem - Why it works

- The information content of a typical string is

$$\begin{aligned} h(x)_{typ} &= \log_2 \frac{1}{P(x)_{typ}} \\ &\approx \log_2 \frac{1}{p_1^{p_1 N} p_2^{p_2 N} p_3^{p_3 N} \ldots p_I^{p_I N}} \\ &= N \sum_i p_i \log_2 \frac{1}{p_i} \\ &= NH(X). \end{aligned}$$

- Note that we just showed that

$$h(x)_{typ} = \log_2 \frac{1}{P(x)_{typ}} \approx NH(X),$$

  which implies that any typical string has probability

$$P(x)_{typ} \approx 2^{-NH(X)}.$$

The Source Coding Theorem - Why it works

- Using the observations above, we define the **typical set** to be that of typical strings, up to a tolerance $\beta \geq 0$:

$$T_{N\beta} = \{x \mid 2^{-NH(X)-\beta} \leq p(x) \leq 2^{-NH(X)+\beta}\}.$$

- The typical set satisfies two important properties:

  1. The typical set $T_{N\beta}$ contains almost all probability: $p(x \in T_{N\beta}) \approx 1$.

     - That is a direct consequence of the law of large numbers: as N grows, it becomes less and less likely that the frequency of any given symbol in a string will differ from its probability of being generated by the source.

       Hence, if N grows, more and more strings will happen to be typical.

  2. The typical set $T_{N\beta}$ contains roughly $2^{NH(X)}$ elements: $|T_{N\beta}| \approx 2^{NH(X)}$.

     - That is a consequence from the fact that the typical set has probability almost 1, and the probability of a typical element is $2^{-NH(X)}$, so the set must have about $1/2^{-NH(X)} = 2^{NH(X)}$ elements.

The Source Coding Theorem - Why it works

- The properties of the typical set lead us to Shannon's coding theorem as follows.
- We know that

  1. S_δ contains all probability of the sequences in X^N, up to an error δ, and
  2. the typical set $T_{N\beta}$ contains almost all probability of the sequences in X^N.

  Hence can conclude that the two sets must have a great intersection:

$$|S_\delta| \approx |T_{N\beta}| \approx 2^{NH(X)}.$$

- That means that the essential information content of X^N is

$$H_\delta(X^N) = \log_2 |S_\delta| \approx \log_2 |T_{N\beta}| \approx \log_2 2^{NH(X)} = NH(X) \text{ bits},$$

  which is exactly what Shannon's coding theorem states.

The Source Coding Theorem - Asymptotic Equipartition Property

- As an addendum, if you like to be formal, our justification of Shannon's coding theorem can be formalized by the following principle, which is a direct consequence of the **law of large numbers**.

- **Asymptotic Equipartition Principle (AEP).** For an ensemble of N independent identically distributed (i.i.d.) random variables $X_N = (X_1, X_2, \ldots, X_N)$, with N sufficiently large, the outcome $x = (x_1, x_2, \ldots, x_N)$ is almost certain to belong to a subset of $\mathcal{A}_X^N$ having only $2^{NH(X)}$ members, each having probability "close" to $2^{-NH(X)}$.

Take-home message

- At the beginning of this lecture we set the goal to convince you of three claims:

    1. The Shannon information content $h(x = a_i) = \log_2 \frac{1}{p(x=a_i)}$ is a sensible measure of the information content of the outcome $x = a_i$.
    2. The entropy $H(X) = \sum_{x \in \mathcal{A}_X} p(x) \log_2 \frac{1}{p(x)}$ is a sensible measure of the expected information content of an ensemble X.
    3. The Source Coding Theorem: N outcomes from a source X can be compressed into roughly $N \cdot H(X)$ bits.

- Are you convinced?

  After this lecture, can you provide good arguments in favor of them?