

Handout 2: Data compression

Instructor: Shashank Vatedka

Disclaimer: *These notes have not been subjected to the usual scrutiny reserved for formal publications. Please email the course instructor in case of any errors.*

2.1 Data compression

- Purpose of data compression: save space.
- Convert a *source* file having n bytes to a *compressed* file having $k < n$ bytes.
- Model for source file X^n : randomly generated according to source distribution.
- $X^n \sim p_{X^n}$ in general.
- In this course, assume that X^n is iid $\sim p_X$.
- More realistic model: Markov source. But ideas similar for this case as well.

2.1.1 The source compression problem for discrete memoryless sources

A source sequence X^n is said to be discrete memoryless with source distribution p_X if $X_i \in \mathcal{X}$, where \mathcal{X} is a discrete set, and X^n is an iid sequence with components drawn according to p_X .

A (k, n) compressor/compression scheme for an iid source X^n with each component drawn according to p_X consists of a pair of maps (f, g) , an *encoder* $f : \mathcal{X}^n \rightarrow \{0, 1\}^k$ and *decoder* $g : \{0, 1\}^k \rightarrow \mathcal{X}^n$. The rate of the scheme is $R \stackrel{\text{def}}{=} k/n$. The probability of error is

$$P_e \stackrel{\text{def}}{=} \Pr[g(f(X^n)) \neq X^n].$$

This is also called a fixed-length compressor.

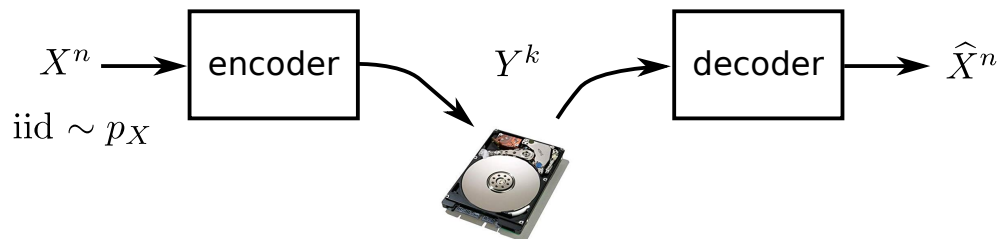


Figure 2.1: Formal setup for the source compression problem.

We want to minimize R for a given P_e , or minimize P_e for a given R .

Note: For the moment, we will assume that \mathcal{X} is discrete. The case when \mathcal{X} is a continuous set (such as \mathbb{R}), will be dealt with in a future course.

Naive solution: Represent X^n in binary without any compression. This gets us to $R = \lceil \log_2 |\mathcal{X}| \rceil$.

2.1.1.1 Optimal solution for minimum P_e

Order sequences in order of decreasing probability, and pick the first 2^k sequences. Call this set \mathcal{S} . Set $f(x^n)$ to be i if x^n is i th in order and $x^n \in \mathcal{S}$, and set it to be the all-zeros vector 0^k otherwise.

The probability of error of the optimal scheme is

$$P_e = \Pr[X^n \in \mathcal{S}].$$

Can we get an expression for P_e for a desired R , or minimum R for a given P_e ? This turns out to be difficult.

Easier: Give a scheme that minimizes R for large n (i.e., $n \rightarrow \infty$), while ensuring that $\lim_{n \rightarrow \infty} P_e = 0$.

The following is called the fundamental theorem of source compression/source coding.

Theorem 2.1 (Shannon, 1948). *There exists a compression scheme that achieves $\lim_{n \rightarrow \infty} P_e = 0$ and*

$$\lim_{n \rightarrow \infty} R \approx H(X) \stackrel{\text{def}}{=} - \sum_{x \in \mathcal{X}} p_X(x) \log_2 p_X(x).$$

Moreover, if a compression scheme for large n has rate less than $H(X)$, then $\lim_{n \rightarrow \infty} P_e > 0$.

The quantity $H(X)$ is called the entropy of the random variable X . Some comments:

- $H(X)$ in fact, is abuse of notation. This is because X is a random variable, while the entropy is a deterministic function of p_X . The notation $H(p_X)$ would have been a better choice. However, we will follow $H(X)$ since this is more commonly used in practice (and by the Cover-Thomas textbook).
- There is another abuse of notation above. The correct definition should be

$$H(X) \stackrel{\text{def}}{=} - \sum_{x \in \mathcal{X}: p_X(x) > 0} p_X(x) \log_2 p_X(x).$$

We will therefore “redefine” $x \log_2(x)$ such that $x \log_2 x = 0$ for $x = 0$. Indeed, $\lim_{x \rightarrow 0} x \log x = 0$.

- $H(X)$ captures the amount of randomness of a source. The data compression problem can be thought of as one of formulating a sequence of yes/no questions to arrive at X^n .

We will prove this for Bernoulli sources. In fact, we will use a suboptimal source code and still achieve the rate guaranteed in Theorem 2.1.

Recall that a binary random variable X is said to be Bernoulli(p) if $\Pr[X = 1] = p$ and $\Pr[X = 0] = 1 - p$. Let us assume that X^n is iid with Bernoulli(p) components, for some $0 < p < 1/2$.

Some questions for you:

- Why does it suffice to consider $0 < p < 1/2$?

- What happens if $p = 0$? What is the optimal source code?
- What happens if $p = 1/2$? What is an optimal source code?

Recall that in the optimal scheme in Sec. 2.1.1.1, \mathcal{S} is a set of 2^k sequences with the largest probabilities. We will instead use the following set: For any $0 < \epsilon < 1$, define

$$\mathcal{T}_\epsilon = \{x^n : np(1 - \epsilon) \leq \text{number of 1's in } x^n \leq np(1 + \epsilon)\}.$$

Prove the following claims:

Claim 2.2.

$$\Pr[X^n \notin \mathcal{T}_\epsilon] \leq 2e^{-n\epsilon^2 p/3}$$

This can actually be improved.

Claim 2.3.

$$\Pr[X^n \notin \mathcal{T}_\epsilon] \leq 2^{-nD(p(1+\epsilon)\|p)(1+o(1))}$$

where we define

$$D(p\|q) \stackrel{\text{def}}{=} p \log_2 \frac{p}{q} + (1-p) \log_2 \frac{1-p}{1-q}$$

to be the Kullback-Liebler (KL) divergence between p and q .

Claim 2.4.

$$|\mathcal{T}_\epsilon| \leq 2^{n(H_2(p)+\epsilon)(1+o(1))}$$

where

$$H_2(p) = -p \log_2 p - (1-p) \log_2 (1-p)$$

is called the binary entropy of p .

Proof. First, argue that the number of sequences of length n having exactly l 1's is equal to $\binom{n}{l}$. Then, justify the following chain of inequalities:

$$\begin{aligned} |\mathcal{T}_\epsilon| &= \sum_{l=np(1-\epsilon)}^{np(1+\epsilon)} \binom{n}{l} \\ &\leq 2np\epsilon \max_{np(1-\epsilon) \leq l \leq np(1+\epsilon)} \binom{n}{l} \end{aligned}$$

Now use Stirling's approximation $k! = \sqrt{2\pi k}(k/e)^k$ in the above, simplify, and maximize the expression to prove the claim. \square

Now prove Theorem 2.1 using the above claims.

You can prove the following quite easily:

Claim 2.5. Entropy is a nonnegative function of p_X .

In the coming lectures, we will study various properties of the entropy, which reinforce the intuition that it measures the randomness of a source.

There are sources which are not "compressible" in some sense.

Claim 2.6. *If p_X is the uniform distribution on a finite alphabet \mathcal{X} , then $H(X) = \log_2 |\mathcal{X}|$. In other words, the best compressor for X^n can not do any better than the naive scheme which represents the input directly in binary form.*

Note: Intuitively, if a compressor is optimal (in the sense of achieving minimum possible rate), then the distribution of the codewords should be uniform. Otherwise, we could compress the compressed sequence to reduce the rate further, leading to a contradiction.

Note: The entropy rate of a source X^n is defined as

$$H_r = \lim_{n \rightarrow \infty} \frac{H(X^n)}{n}.$$

For stationary and ergodic sources, the entropy rate is the best possible compression rate that we can achieve. We will show the following in later lectures:

- For an iid source, $H_r = H(X)$.
- For a first-order Markov source with stationary distribution π and transition probability $p_{X_2|X_1}$, the entropy rate simplifies to

$$H_r = - \sum_{x_1, x_2 \in \mathcal{X}} p_{X_2|X_1}(x_2|x_1) \pi(x_1) \log_2 p_{X_2|X_1}(x_2|x_1).$$