Data Compression

Introduction

Description of a data sequence from a random source can be compressed by assigning shorter descriptions to the more frequent symbols and longer descriptions to the less frequent ones. This is the basic idea of data compression.

Source Code

- A source code C is a mapping from $\mathcal{X}$ to $\mathcal{D}^*$, where $\mathcal{X}$ is the source alphabet and $\mathcal{D}$ is the code alphabet. $C(x)$ is called the codeword of x.
- The expected length $L(C)$ is defined by

$$L(C) = \sum_{x \in \mathcal{X}} p(x) l(x),$$

  where $l(x)$ is the length of the codeword for x.
- Examples of source codes: 5.1.1, 5.1.2, 5.1.3 (Morse code)

Classes of Source Codes

- Non-singular codes: the codewords of two different symbols are different.
- Uniquely decodable codes: A code is uniquely decodable if its extension is non-singular. The extension C^* of a source code C is defined by

$$C^*(x_1 \dots x_n) = C(x_1) \dots C(x_n)$$

- Prefix or instantaneous codes: no codeword is a prefix of any other codewords.
- Examples (Table 5.1) and relation (Fig 5.1)

Kraft Inequality

The important things about a code are the decodability and the codeword length. The following theorem connects the two.

Theorem (Kraft inequality)

For a prefix code over an alphabet of size D (e.g. $\mathcal{D} = \{0, 1, \ldots, D-1\}$) for m possible outputs, the codeword lengths $l_i, i = 1, \ldots, m$, must satisfy the Kraft inequality

$$\sum_i D^{-l_i} \leq 1.$$

Conversely, given lengths that satisfy the Kraft inequality, one can construct a prefix code.

Kraft's Inequality – Proof

A source code can be related to a D-ary tree. A codeword of length l correspond to a node at level l (the root is at level 0). If it is prefix, no codeword is a descendant of any other codewords in the tree.

Start with the code tree. One can grow from each leaf complete descendants up to the depth of the tree, the length of the longest codeword. By counting the disjoint leafs added at this level and compare to the maximum possible number, the inequality is proven.

To prove the converse, start with a full tree of the maximum length M. Repeatly assign the leftmost node at level l to a codeword of length l and remove its descendants, from $l = 1$ to $l = M$. At any stage, the existence of a node at level l is guaranteed by the Kraft inequality. ■

Extended Kraft Inequality

The Kraft inequality can be extended to cases where the cardinality of symbol set is countably infinite.

Proof

Instead of a D-ary tree, relate the codewords to intervals in $[0, 1]$ with the D-ary representation. That is, relate the codeword $y_1 y_2 \ldots y_{l_i}$ to the interval

$$(0.y_1 y_2 \ldots y_{l_i}, 0.y_1 y_2 \ldots y_{l_i} + \frac{1}{D^{l_i}}).$$

Since it is a prefix code, the related intervals are disjoint. Therefore, the extended Kraft inequality is satisfied. ■

Optimal Codes

- Minimize the expected length $L(C)$ over the set of prefix codes. This is the same as minimizing $L(C)$, subject to the constraint of the Kraft inequality.
- Form the Lagrangian

$$J(l_1, \ldots, l_m, \lambda) = \sum p_i l_i + \lambda \left(\sum D^{-l_i} - 1 \right).$$

  By differentiating with respect to ${l_i}'s$ and λ, one can obtain

$$l_i^* = -\log_D p_i.$$

- The optimal expected length is the entropy,

$$L^* = \sum p_i l_i^* = \sum p_i (-\log_D p_i) = H_D(X)$$

Optimal Expected Length

The expected length of any D-ary prefix code is greater than or equal to the base-D entropy of X.

Proof

Define $c = \sum_i D^{-l_i}$ and $r_i = \frac{D^{-l_i}}{c}$ (so r is a probability), one has

$$\begin{aligned} L - H_D(X) &= \sum_i p_i l_i - \sum_i p_i \log_D \frac{1}{p_i} \\ &= -\sum_i p_i \log_D D^{-l_i} + \sum_i p_i \log_D p_i = \sum_i p_i \log_D \frac{p_i}{r_i} \frac{1}{c} \\ &= D(p||r) + \log_D \frac{1}{c} \geq 0. \ \blacksquare \end{aligned}$$

Note that the equality only holds if $c = 1$ and $p = r$.

D-adic probability

- A probability function is *D-adic* if each probability can be written as

$$p_i = D^{-n} \text{ where } n \text{ is an integer.}$$

- The expected codelength of a D-adic distribution is optimal when

$$l(x_i) = -\log_D p(x_i).$$

- This optimal value is equal to the entropy of X.
- How about distributions that are not D-adic?

Bounds on the Optimal Codelength

Given distribution p and a D-ary alphabet, the minimum expected length L^* over the prefix codes satisfies

$$H_D(X) \le L^* < H_D(X) + 1.$$

Proof

Choose the codeword length for x_i

$$l_i = \lceil \log_D \frac{1}{p_i} \rceil,$$

which satisfies the Kraft inequality, since

$$c = \sum D^{-l_i} \le \sum D^{-\log \frac{1}{p_i}} = \sum p_i = 1,$$

so a prefix code can be constructed. For this prefix code,

$$\log_D \frac{1}{p_i} \leq l_i < \log_D \frac{1}{p_i} + 1.$$

Multiplying by p_i and sum over i, we have

$$H_D(X) \leq L \leq H_D(X) + 1.$$

The optimal expected length can only be better than L but cannot be better than $H_D(X)$, thus the bounds. ■

The idea of proof is that the function value of a point is an upper bound for the minimum value and a lower bound for the maximum value over a set containing the point.

Expected Codeword Length Per Symbol

- Definition

$$L_n = \frac{1}{n}\sum p(x_1,\ldots,x_n)l(x_1,\ldots,x_n) = \frac{1}{n}El(X_1,\ldots,X_n)$$

- From

$$H(X_1,\ldots,X_n) \le El(X_1,\ldots,X_n) \le H(X_1,\ldots,X_n)+1$$

the minimum expected codeword length per symbol satisfies

$$\frac{H(X_1,\ldots,X_n)}{n} \le L_n^* \le \frac{H(X_1,\ldots,X_n)}{n} + \frac{1}{n}.$$

If $X_1, X_2, \ldots, X_n$ is stationary, then $L_n^* \to H(\mathcal{X})$, the entropy rate.

Using Incorrect Distribution for Shannon Code

If the true distribution of X is p and the codeword length is assigned according to q,

$$l(x) = \lceil \log \frac{1}{q(x)} \rceil,$$

then

$$H(p) + D(p||q) \le El(X) < H(p) + D(p||q) + 1.$$

Thus the penalty is $D(p||q)$.

For the upper bound,

$$\begin{aligned} El(X) &= \sum p(x) \lceil \log \frac{1}{q(x)} \rceil < \sum p(x) \left(\log \frac{1}{q(x)} + 1 \right) \\ &= \sum p(x) \log \frac{1}{p(x)} \frac{p(x)}{q(x)} + 1 = H(p) + D(p||q) + 1. \end{aligned}$$

Similarly for the lower bound. ■

Kraft Inequality for Uniquely Decodable Codes

We show that the for the set of uniquely decodable source codes, the Kraft inequality holds. That is, for any uniquely decodable codes,

$$\sum D^{-l_i} \leq 1.$$

Consider the k-th extension of code C where $x_1 \dots x_k$ is mapped to $c(x_1) \dots c(x_k)$. Note that $l(x_1 \dots x_k) = \sum_n l(x_n)$.

$$\left(\sum_{x \in \mathcal{X}} D^{-l(x)} \right)^k = \sum_{x^k \in \mathcal{X}^k} D^{-l(x^k)} = \sum_{m=1}^{kl_{max}} a(m) D^{-m}$$

$$\leq \sum_{m=1}^{kl_{max}} D^m D^{-m} = kl_{max},$$

where $a(m) \leq D^m$ is the number of codewords in C^k with length m. So

$$\sum_{x \in \mathcal{X}} D^{-l(x)} \leq (kl_{max})^{\frac{1}{k}} \to 1.$$

Huffman Codes

Huffman code is a prefix code with the minimum expected length for a given distribution. Before we give the details for the Huffman code, we notice the following results.

There exists an optimal prefix code that satisfies the following properties.

1. If $p_j > p_k$, then $l_j \leq l_k$. (swapping)
2. The two longest codewords have the same length. (trimming)
3. The two longest codewords differ only in the last bit and corresponds to the two least likely symbols. (re-arranging)

Suppose that $p_1 \geq p_2 \geq \cdots \geq p_m$, then there exists a code with $l_1 \leq l_2 \leq \cdots \leq l_{m-1} = l_m$, and $C(x_{m-1})$ and $C(x_m)$ differ only in the last bit.

Huffman Codes

Given an optimal code C_m satisfying the above properties with m symbols, we define a merged code C_{m-1} with $m-1$ symbols in which we merge the two least likely symbols into a new symbol. For this new symbol we assign the common prefix as its codeword and assign the probability sum as its probability. Other symbols stay put. The expected length of C_{m-1} and C_m is differed by $p_m + p_{m-1}$, which is constant. We can now look for the optimal code for the new set of $m-1$ symbols. Proceeding this way, we will finally reach a two-symbol problem, which has the trivial solution of assigning 0 to one symbol and assigning 1 to the other. At this point, we can trace back for the original code. A few examples will help to see how this works, e.g., pp. 93-94.

Shannon-Fano-Elias Coding

- Use the *cumulative distribution function* to assign codewords.
- Defined a modified cdf

$$\bar{F}(x) = \sum_{a<x} p(a) + \frac{1}{2}p(x),$$

  so its value at x is the middle point of the cdf jump.

- The codeword for x is the truncated binary representation of $\bar{F}(x)$ to $l(x) = \lceil \log \frac{1}{p(x)} \rceil + 1$ bits, denoted by $\lfloor \bar{F}(x) \rfloor_{l(x)}$. Then $2^{-l(x)} \leq \frac{p(x)}{2}$ and such assignment assures that the intervals

$$[\lfloor \bar{F}(x) \rfloor_{l(x)}, \lfloor \bar{F}(x) \rfloor_{l(x)} + 2^{-l(x)}]$$

  for all x's are disjoint. So these codes are prefix-free. Furthermore, the expected codeword length is less then $H(X) + 2$. Example 5.9.1.

Arithmetic Coding 1/2

- The idea of the Shannon-Fano-Elias coding can be extended to blocks of source symbols.
- Here we encode a fixed length n of binary symbols. We associate the length-n sequences with the leaves of a depth-n binary tree.
- The cdf $F(x^n)$ is the sum of the probabilities of all the subtrees to the left of x, plus $p(x^n)$.
- See Figure 5.6 and Example 5.10.1.

Arithmetic Coding 2/2

- To encode the next source symbol, we need only calculating $p(x^i x_{i+1})$ to update $F(x^i x_{i+1})$.
- This is easy if $p(x^i x_{i+1})$ can be computed easily for all x^i and x_{i+1}.
  - I.i.d symbols
  - Markov model
- To decode, we use Figure 5.6 as a decision tree. At the top of the tree, we check if the codeword exceeds $p(0)$. If it does, then the 0-subtree is to the left of the leaf of the codeword, so the first symbol is 1. Continue this process until we find the leaf matching the codeword.