**EE5585 Data Compression** January 22, 2013

# Lecture 1

*Instructor: Arya Mazumdar* *Scribe: Arya Mazumdar*

## Syllabus

- Preliminaries: Probability, Discrete Mathematics, Information Theory
- Lossless coding: Source Coding Theory(Prefix-free and Universally Decodable Codes), Huffman Coding, Arithmetic Codes, The Lempel-Ziv Algorithm
- Lossy Coding: Rate-distortion Theory, Scalar and Vector Quantization, Transform Coding Techniques, Subband Coding, Predictive Coding, Case Studies (JPEG/MPEG)
- Compressed Sensing, Update/Query Efficient Compression.

## Compressible Data

Some of the data are more likely than others.

Example: The source produces a binary $\{0,1\}$-vector of length 6. The number of 1s in the vector is exactly 2. We know that there are $\binom{6}{2} = 15$ such sequences.

| Data | Index | Compressed Data |
|---|---|---|
| 110000 | 0 | 0000 |
| 101000 | 1 | 0001 |
| 100100 | 2 | 0010 |
| 100010 | 3 | 0011 |
| 100001 | 4 | 0100 |
| 011000 | 5 | 0101 |
| 010100 | 6 | 0110 |
| 010010 | 7 | 0111 |
| 010001 | 8 | 1000 |
| 001100 | 9 | 1001 |
| 001010 | 10 | 1010 |
| 001001 | 11 | 1011 |
| 000110 | 12 | 1100 |
| 000101 | 13 | 1101 |
| 000011 | 14 | 1110 |

**Table 1**: A scheme showing six bit compressed to four bits

Given a source sequence, say 010010, we find out its index from the above table and represent it in binary. The rate of compression here is $\frac{4}{6} = \frac{2}{3}$.

Let us generalize this example to the case where the source can produce any $n$ length binary sequence with exactly $k$ ones. It is clear that there are $\binom{n}{k}$ such sequences. Hence a rate of compression

$$\left\lceil \frac{1}{n} \log_2 \binom{n}{k} \right\rceil,$$

is achievable as well as best possible. We may use Stirling approximation, $n! \approx \sqrt{2\pi n}\left(\frac{n}{e}\right)^n$ to evaluate this compression rate. Indeed, we have,

$$\frac{1}{n}\log_2 \binom{n}{k} \approx \frac{1}{2n}\log_2 \frac{n}{2\pi k(n-k)} - \frac{k}{n}\log_2 \frac{k}{n} - (1-\frac{k}{n})\log_2(1-\frac{k}{n}).$$

Hence,

$$\begin{aligned}\lim_{n\to\infty} \frac{1}{n}\log_2 \binom{n}{k} &= -\frac{k}{n}\log_2 \frac{k}{n} - (1-\frac{k}{n})\log_2(1-\frac{k}{n}) \\ &= h(k/n),\end{aligned}$$

where

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

is called the *binary entropy function*.

Note that data compression was possible because the set of feasible sequences was of a much smaller size than the ambient space.

# Entropy

Suppose a random variable $X$ has range the finite set $\mathcal{X}$. Let, $\mathbb{P}(X = x) \equiv p(x)$ for all $x \in \mathcal{X}$. Define,

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

to be the entropy of the random variable $X$. Note that the entropy is a function of the probability distribution and not the random variable itself.

# Transform Coding

Let us look at another model of data. Let $\mathcal{Y} = \{0, 1, \ldots, 255\}$ and let the source is $\mathcal{X} \subset \mathcal{Y}^7$. However, for any $x = (x_1, \ldots, x_7) \in \mathcal{X}$, $|x_i - x_{i+1}| < 32$, for $1 \le i \le 6$. For example, one possible source sequence might be

$$(192, 218, 200, 170, 160, 160, 140).$$

We could have enumerated the number of such sequences. However, we choose to multiply the source sequence with the following matrix.

$$\begin{pmatrix} 1 & -1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & -1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & -1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & -1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & -1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix}$$

Upon multiplication with a source sequence, say $(192, 218, 200, 170, 160, 160, 140)^T$, we will always obtain an array whose entries are between $-32$ and 32. Because the matrix is invertible, it is easy to get back the original sequence. This is a change of basis for the source sequence. As the amplitude of the source in this new basis has a significantly smaller range (-32 to 32) than the original one (0 to 255), data compression is immediate.

This change of basis for dimensionality reduction or data compression is called transform coding. In the most studied cases this transformation is unitary, for example multiplying with a DFT (discrete Fourier transform) matrix.

# Source Coding

Suppose the finite set $\mathcal{X}$ is the source and for any $x \in \mathcal{X}$, $p(x)$ denotes the probability of $x$. We assume that the encoder will assign binary $\{0,1\}$ valued vectors (codewords) for each symbol. Let for the code $\mathcal{C}$, $C(x)$ denote the codeword corresponding to $x \in \mathcal{X}$. Also let $\ell(x)$ denote the length of $C(x)$.

Our measure of compression will be the expected value of $\ell(x)$, i.e., the average number of bits per source symbol. We denote it by

$$L(\mathcal{C}) \equiv \mathbb{E}\ell(x) = \sum_{x \in \mathcal{X}} p(x)\ell(x).$$

Example: Let $\mathcal{X} = \{1,2,3\}$ and $p(1) = \frac{5}{6}, p(2) = \frac{1}{12} = p(3)$. Let for a code $\mathcal{C}$, $C(1) = 0, C(2) = 10, C(2) = 11$. Hence,

$$L(\mathcal{C}) = \frac{5}{6} + \frac{1}{6} + \frac{1}{6} = \frac{7}{6}.$$

For unambiguous decoding it is apparent that we need no two source symbol to be mapped to the same codeword. That is, for all $x, x' \in \mathcal{X}$ such that $x \neq x'$, $C(x) \neq C(x')$. Such codes are called *nonsingular code*.

Example: Suppose $\mathcal{X} = \{1,2,3,4\}$ and $C(1) = 0, C(2) = 1, C(3) = 01, C(4) = 101$. This code is nonsingular. But note that, source sequences $(1,4)$ and $(3,3)$ produce the same bit stream 0101.

A code is called *uniquely decodable* if for any $\{0,1\}$-vector there is at most one sequence of source symbols that can produce it up on encoding.

Example: Suppose $\mathcal{X} = \{1,2,3,4\}$ and $C(1) = 00, C(2) = 10, C(3) = 11, C(4) = 110$. This code is uniquely decodable. Take for example the encoded sequence 10110100011110 that decodes uniquely to 242134.

**Theorem 1** *For any uniquely decodable code $\mathcal{C}$ for the source $\mathcal{X}$ with source symbols following the distribution of random variable $X$,*

$$L(\mathcal{C}) \geq H(X).$$

We need the following famous result to prove this.

**Lemma 2 (Kraft inequality)** *For any uniquely decodable code for source $\mathcal{X}$,*

$$\sum_{x \in \mathcal{X}} 2^{-\ell(x)} \leq 1.$$

**Proof** Consider, for any $k > 1$, the following term.

$$\begin{aligned} \Big[\sum_{x \in \mathcal{X}} 2^{-\ell(x)}\Big]^k &= \textstyle\sum_{x_1 \in \mathcal{X}} \sum_{x_2 \in \mathcal{X}} \cdots \sum_{x_k \in \mathcal{X}} 2^{-\ell(x_1)-\ell(x_2)\ldots-\ell(x_k)} \\ &= \textstyle\sum_{x_1^k \in \mathcal{X}^k} 2^{-\ell(x_1^k)}, \end{aligned}$$

where $x_1^k$ is a super symbol consisting of $k$ source symbols. Suppose $a(m)$ is the number of such symbols such that $\ell(x_1^k) = m$. As the code is uniquely decodable $a(m) \leq 2^m$. Hence,

$$\begin{aligned} \Big[\sum_{x \in \mathcal{X}} 2^{-\ell(x)}\Big]^k &= \textstyle\sum_m a(m) 2^{-m} \\ &\leq \textstyle\sum_m 1 \\ &= k \max_{x \in \mathcal{X}} \ell(x). \end{aligned}$$

If the Kraft inequality is not satisfied, the left hand side will grow exponentially with $k$ whereas the right hand side grows only linearly. For sufficiently large $k$ we hence will have a contradiction. Therefore,

$$\sum_{x \in \mathcal{X}} 2^{-\ell(x)} \leq 1.$$

■