CS 493: Algorithms for Massive Data Sets **Information Theory**
2/5/2002 **Scribe: Elena Zaslavsky**

1 Introduction

1.1 Basic Definitions

Information Theory studies the statistical characteristics of data and communication systems. It was originally motivated by the problem of efficiently transmitting information over a noisy communication channel. In this context, given are:

- Set of messages S
- *Source*: generates messages from S and transmits the sequence to a *receiver* (any corruption in transmission is ignored)
- *Encoder*: converts the messages to a sequence of bits
- *Decoder*: retrieves the original message sequence from the encoded one

Schemes for conversion of the data into encoded form, referred to as *compression* schemes, fall into two categories:

Lossless: In this case, the decoder is able to recover the original sequence completely. Lossless compression is used in context of text data, for example.

Lossy: Here, the decoder recovers only an approximation to the original message sequence. Lossy compression is used in context of image (or sound) data, where the human eye (ear) is not sensitive to certain systematic changes.

1.2 Compression Framework

The basic question of Information Theory deals with how many bits are necessary to encode a sequence of messages. Note first that it is not always possible to compress a data sequence, as example below illustrates.

Example: Consider all 100-bit data sequences. There are 2^{100} of them. If every such sequence was expressible in a smaller number of bits, the maximum number of encoded sequences would be: $2^{99} + 2^{98} + \cdots + 2^{1} = 2^{100} - 2 < 2^{100}$.

Thus, there is no universal compression scheme.

Therefore any compression algorithm must assume some bias in the input messages, and use it to develop an efficeint compression code. Often this bias is based on the particular structure that the messages possess. Utilizing this structure, schemes can be developed to represent the messages more compactly. The problem is typically viewed in the framework of a particular *model / coder* pair. Each model exploits different message features to capture information about the message probability distribution. The coder then utilizes this bias in producing the encoded representation. Information Theory studies how an individual model relates to the lengths of code words produced by its coder.

1.3 Compression Quality

The quality of a compression scheme is usually measured by:

- Time used to compress / reconstruct messages
- Size of encoded message
- Generality of the technique. It is often possible to optimize a scheme for a specific domain, but such a scheme is unlikely to be extensible to other domains.

In the case of lossy compression, the quality of the reconstructed approximation is also an added issue. Typically, there exists a tradeoff when trying to improve any one of the above factors.

2 Information Theory

2.1 Entropy

Let s be a message in S and let $p(s)$ be the probability that s was generated by the source.

Definition 2.1 *Self-information of s,* $i(s) = \log_2 1/p(s)$

Definition 2.2 *Entropy* $H(S) = \sum_{s \in S} p(s) i(s)$

Intuitively, the self-information tells us how much information the particular messages carries, and therefore roughly how many bits we should use in representing it. It also says that the higher the probability of a message, the less information it carries. Entropy is thus the average self-information of a message set, and represents the average information content.

Why is the above definition of entropy the right measure of information content for a set of messages? There are a few properties that any information content measure should satisfy, and the entropy function does the job.

1. Let $n = 2^i$, and consider a set S of equiprobable messages. Since i bits are needed to encode every message, information content of this set should be i. $H(S)$ satisfies this requirement.
$$H(S) = \log_2 n = i$$

2. Suppose we have two independent sources A, B. Then it is natural to expect that the information content of a concatenated message should be that of the sum of individual messages. Indeed, $H(AB) = H(A) + H(B)$ due to the properties of the logarithm function.

2.1.1 Entropy of the English Language

Let us try to estimate the entropy of the English language to get an idea of how many bits we need per character in encoding a typical piece of English text. We will consider English text to be a sequence of characters from the set S consisting of the 96 ASCII characters. We will measure the entropy in units of bits/character by using a series of estimates. As we see below compression at the character level does not achieve a good bound.

if the characters are equiprobable	$H(S) = \lceil \log_2 96 \rceil \approx 7$
if probabilities are character count based	$H(S) \approx 4.5$
if assume a code for each character (e.g. Huffman — optimal)	$H(S) \approx 4.7$
if group characters into chunks of 8	$H(S) \approx 2.4$
if group characters into larger and larger chunks	$H(S) \longrightarrow 1.3$

2.2 Source Models

The message source can be modeled in various probabilistic frameworks.

1. *I.I.D. source.* The messages are generated by independent identically distributed random variables. This is the scenario described above, and the entropy of such a source is only dependent on the message set S, $H(S) = \sum_{s \in S} p(s) i(s)$

2. *Markov Chain source.* This source model should be applied when messages probabilities are dependent on previous message occurrences. So, for a k-th order Markov chain, $p(x_n|x_1, \ldots, x_{n-1}) = p(x_n|x_{n-k}, \ldots, x_{n-1})$

   Let's denote the event of a message occurence in this model by e, and the sequence of previous messages (or context) by c. Let C be the set of all possible contexts. Then $p(e|c)$ is the conditional probability of event e based on context c. The unconditional probability of event e is defined as $p(e) = \sum_{c \in C} p(c) p(e|c)$

   Definition 2.3 *Conditional Self-information of* e, $i(e|c) = \log_2 1/p(e|c)$

   Definition 2.4 *Conditional Entropy* $H(S|C) = \sum_{c \in C} p(c) \sum_{s \in S} p(s|c) \log_2 1/p(s|c)$

   If the probability distribution of S is independent of C, $H(S|C) = H(S)$. In general, $H(S|C) \leq H(S)$. That is, knowing the context can only reduce entropy. Consider an example:

Example: Suppose a source generates a black / white image. Denote the generation of a black pixel by b, and of a white pixel by w. Let the source be a first-order Markov chain and the probabilities be biased as follows.

$p(w|w) = .99 \quad p(b|b) = .7$
$p(b|w) = .01 \quad p(w|b) = .3$

The source has to be in either b or w state, i.e. $p(w) + p(b) = 1$. From the transition probabilities, we get $p(w) = p(w)p(w|w) + p(b)p(w|b)$. Solving the equations, we get $p(w) = 30/31$, $p(b) = 1/31$. Then $H(S|C) \approx 1.07$ and $H(S) \approx 2.06$, where the entropy would be significantly overestimated if the conditions were ignored.

3. *Arbitrary source.* Let A^n denote all strings of length n from alphabet A.

   Definition 2.5 *n-th order entropy* $H_n = 1/n \sum_{x \in A^n} p(x) \log_2 1/p(x)$

   Definition 2.6 *Entropy* $H = \lim_{n\to\infty} H_n$

3 Coding Schemes

Compression algorithms can be distinguished by the kinds of codes they produce. Some (like Huffman coding) produce unique codewords for every message. Others (like Arithmetic coding) blend code words of various message sequences. Algorithms can also be differentiated by fixed / variable length codes they produce. Clearly, fixed length codes are easier to decode, but they are inefficient, whereas variable length codes provide efficiency at the expense of decoding ease.

Definition 3.1 *A code C is a mapping from each message to a codeword, i.e. a bit sequence. Denote $C = \{(s_i, w_i)_{i=1}^n\}$, where s_i is a message, and w_i is the corresponding codeword.*

For any code, we want to be able to uniquely reconstruct the original message sequence. Note that not all codes possess this property. For instance, the code $\{(a, 1), (b, 01), (c, 101), (d, 011)\}$ is ambiguous.

3.1 Prefix Codes

Definition 3.2 *A uniquely decodable code is one in which every encoded string has a unique decoding.*

An important subclass of uniquely decodable codes is the class of *prefix codes.* A prefix code is one in which no codeword is a prefix of another. This scheme has an advantage over other uniquely decodable codes in that it allows the message to be decoded in real time, i.e. the message sequence is reconstructed piece by piece as every occurring codeword is decoded. There is no need to decode the entire message sequence before getting partial results.

If we restrict prefix codes to the binary alphabet, they can be viewed as binary trees. In fact, prefix codes are in $1-1$ correspondence with *prefix code trees.* In a prefix code tree every codeword corresponds to a leaf, and the codeword is obtained by walking the path from the root to the corresponding leaf, appending a 0 each time a left branch is taken and a 1 each time a right branch is taken. These trees are especially useful in decoding: as soon as a leaf is reached, a message is obtained.

3.1.1 Relationship with Entropy

Given a message set S, the corresponding set of probabilities $\{p(s)\}_{s\in S}$, and code C, we introduce a 'quality measure' for a code, the average code length.

Definition 3.3 *Average code length $l_a(C) = \sum_{s\in S} p(s)l(w_s)$, where $l(w_s)$ is the length of the codeword corresponding to message s.*

Lemma 3.4 (Kraft-McMillan Inequality) *(feasible collections of codeword lengths)*

1. *For any uniquely decodable code C,*

$$\sum_{(s,w_s)\in C} 2^{-l(w_s)} \leq 1$$

2. *For any collection of lengths L, satisfying*

$$\sum_{l\in L} 2^{-l} \leq 1$$

there exists a prefix code C of the same size, such that $l(w_i) = l_i$

Lemma 3.5 *For any message set S with a probability distribution and an associated uniquely decodable code C, $l_a(C) \geq H(S)$.*

Proof.

$$\begin{aligned}
H(S) - l_a(C) &= \sum_{s\in S} p(s)\log_2 \frac{1}{p(s)} - \sum_{s\in S} p(s)l(w_s) \\
&= \sum_{s\in S} p(s)\left(\log_2 \frac{1}{p(s)} - l(w_s)\right) \\
&= \sum_{s\in S} p(s)\left(\log_2 \frac{1}{p(s)} - \log_2 2^{l(w_s)}\right) \\
&= \sum_{s\in S} p(s)\left(\log_2 \frac{2^{-l(w_s)}}{p(s)}\right) \\
&\leq \log_2 \sum_{s\in S} 2^{-l(w_s)} \\
&\leq 0
\end{aligned}$$

The second to last line is based on Jensen's inequality, which states: for a concave function $f(x)$ and $\{p_i \geq 0\}$, $\sum_i p_i f(x_i) \leq f(\sum_i p_i x_i)$. The last line uses the Kraft-McMillan inequality.
■

Lemma 3.6 *For any message set S with a probability distribution, there exists a prefix code C, such that*

$$la(C) \leq H(S) + 1$$