Information and Coding Theory **Autumn 2014**

Lecture 1: September 30, 2014

Lecturer: Madhur Tulsiani Scribe: Madhur Tulsiani

1 Administrivia

This course will cover some basic concepts in information and coding theory, and their applications to statistics, machine learning and theoretical computer science.

- The course will have 4-5 homeworks. They will be posted on the course homepage and announced in class. The homeworks will need to be submitted one week after they are posted.
- The only pre-requisite for the course is familiarity with discrete probability and random variables. Some knowledge of finite fields will help with the coding theory part though we will briefly review the relevant concepts from algebra.
- We will not follow any single textbook, though the book *Elements of Information Theory* by T. M. Cover and J. A. Thomas is a good reference for most of the material we will cover. The "Resources" section on the course page also contains links to some other similar courses.
- Check the course webpage regularly for updates.

2 Entropy

The concepts from information theory are applicable in many areas as it gives a precise mathematical way of stating and answering the following question: How much information is revealed by the outcome of a random event? Let us begin with a few simple examples. Let X be a random variable which takes the value a with probability $1/2$ and b with probability $1/2$. We can then describe the value of X using one bit (say 0 for a and 1 for b). Suppose it takes one of the values $\{a_1, \ldots, a_n\}$, each with probability, then we can describe the outcome using $\lceil \log_2(n) \rceil$ bits. The n possible outcomes for this random variable each occur with probability $1/n$, and require $\approx \log_2(n)$ bits to describe.

The concept of entropy is basically an extrapolation of this idea when the different outcomes do not occur with equal probability. We think of the "information content" of an event that occurs with probability p as being $\log_2(1/p)$. If a random variable X is distributed over a universe $U = \{a_1, \ldots, a_n\}$ such that it takes value $x \in U$ with probability $p(x)$. Then, we define the *entropy* of the random variable X as

$$H(X) \quad = \quad \sum_{x \in U} p(x) \cdot \log\left(\frac{1}{p(x)}\right) .$$

The following basic property of entropy is extremely useful in applications to counting problems.

Proposition 2.1 *Let X be a random variable taking values in a finite set U as above. Then*

$$0 \ \leq \ H(X) \ \leq \ \log(|U|)\,.$$

Proof: Since $p(x) \leq 1$ we have $log(1/p(x)) \geq 0$ for all $x \in U$ and hence $H(X) \geq 0$. For the upper bound, consider a random variable Y which takes value $1/p(x)$ with probability $p(x)$. Since $\log(\cdot)$ is a concave function, we use Jensen's inequality to say that

$$\sum_{x \in U} p(x) \cdot \log\left(\frac{1}{p(x)}\right) \ = \ \mathbb{E}\left[\log(Y)\right] \ \leq \ \log\left(\mathbb{E}\left[Y\right]\right) \ = \ \log\left(\sum_{x \in U} p(x) \cdot \frac{1}{p(x)}\right) \ = \ \log(|U|)\,.$$

■

Jensen's inequality: The above proof used Jensen's inequality which is perhaps the most important inequality we will use in this course. Let f be a real-valued convex function and let Y be a random variable taking values in it's domain. Then,

$$\mathbb{E}\left[f(Y)\right] \ \geq \ f(\mathbb{E}\left[Y\right])\,.$$

A function g is concave if and only if $-g$ is convex. Thus, for a concave function g we get

$$\mathbb{E}\left[g(Y)\right] \ \leq \ g(\mathbb{E}\left[Y\right])\,.$$

Exercise 2.2 *Prove Jensen's inequality when the random variable Y has a finite support.*

3 Source Coding

We will now attempt to make precise the intuition that a random variable X takes $H(X)$ bits to describe on average. We shall need the notion of prefix-free codes as defined below.

Definition 3.1 *A code for a set U over an alphabet Σ is a map $C : U \to \Sigma^*$ which maps each element of U to a finite string over the alphabet Σ. We say that a code is* prefix-free *if for any $x, y \in U$ such that $x \neq y$, $C(x)$ is not a prefix of $C(y)$ i.e., $C(y) \neq C(x) \circ \sigma$ for any $\sigma \in \Sigma^*$.*

For now, we will just use $\Sigma = \{0, 1\}$. For the rest of lecture, we will use prefix-free code to mean prefix-free code over $\{0, 1\}$. The image $C(x)$ for an image x is also referred to as the *codeword* for x.

Note that a prefix-free code has the convenient property that if we are receiving a stream of coded symbols, we can decode them online. As soon as we see $C(x)$ for some $x \in U$, we know what we have received so far cannot be a prefix for $C(y)$, for any $y \neq x$. The following inequality gives a characterization of the lengths of codewords in a prefix-free code. This will help prove both upper and lower bounds on the expected length of a codeword in a prefix-free code, in terms of entropy.

Proposition 3.2 (Kraft's inequality) *Let $|U| = n$. There exists a prefix-free code for U over $\{0, 1\}$ with codeword lengths $l_1, \ldots, l_n$ if and only if*

$$\sum_{i=1}^{n} \frac{1}{2^{l_i}} \ \leq \ 1\,.$$

For codes over a larger alphabet Σ, we replace 2^{l_i} above by $|\Sigma|^{l_i}$.

Proof: We first prove the "only if" part. Let C be a prefix-free code with codeword lengths $l_1, \ldots, l_n$ and let $\ell = \max\{l_1, \ldots, l_n\}$. Consider an experiment where we generate ℓ random bits. For $x \in U$, let E_x denote the event that the *first* $|C(x)|$ bits we generate are equal to $C(x)$. Note that since C is a prefix-free code, E_x and E_y are mutually exclusive for $x \neq y$. Moreover, the probability that E_x happens is exactly $1/2^{|C(x)|}$. This gives

$$1 \;\geq\; \sum_{x \in U} \mathbb{P}[E_x] \;=\; \sum_{x \in U} \frac{1}{2^{|C(x)|}} \;=\; \sum_{i=1}^{n} \frac{1}{2^{l_i}}\,.$$

For the "if" part, given $l_1, \ldots, l_n$ satisfying $\sum_i 2^{-l_i} \leq 1$, we will construct a prefix-free code with these codeword lengths. Without loss of generality, we can assume that $l_1 \leq l_2 \leq \cdots \leq l_n = \ell$.

It will be useful here to think of all binary strings of length at most ℓ as a complete binary tree. The root corresponds to the empty string and each node at depth d corresponds to a string of length d. For a node corresponding to a string s, its left and right children correspond respectively to the strings $s0$ and $s1$. The tree has 2^ℓ leaves corresponding to all strings in $\{0,1\}^\ell$.

We will now construct our code by choosing nodes at depth $l_1, \ldots, l_n$ in this tree. When we select a node, we will delete the entire tree below it. This will maintain the prefix-free property of the code. We first chose an arbitrary node s_1 at depth l_1 as a codeword of length l_1 and delete the subtree below it. This deletes $1/2^{l_1}$ fraction of the leaves. Since there are still more leaves left in the tree, there exists a node (say s_2) at depth l_2. Also, s_1 cannot be a prefix of s_2 since s_2 does not lie in the subtree below s_1. We can similarly proceed to choose other codewords. At each step, we have some leaves left in the tree since $\sum_i 2^{-l_i} \leq 1$. ∎

As was noted in the lecture, we need to carry out this argument in increasing order of lengths. Otherwise, if we choose longer codewords first, we may have to choose a shorter codeword later which does not lie on the path from the root to any of the longer codewords and this may not always possible e.g., there exists a code with lengths $1, 2, 2$ but if we choose the strings 01 and 10 first then there is no way to choose a codeword of length 1 which is not a prefix.

The Shannon code: Given a random variable X taking values in U, we now construct a (prefix-free) code for conveying the value of X, using at most $H(X)$ bits on average (over the distribution of X). For an element $x \in U$ which occurs with probability $p(x)$, we will use a codeword of length $\lceil \log(1/p(x)) \rceil$. By proposition 3, there exists a prefix-free code with these codeword lengths, since

$$\sum_{x \in U} \frac{1}{2^{|C(x)|}} \;=\; \sum_{x \in U} \frac{1}{2^{\lceil \log(1/p(x)) \rceil}} \;\leq\; \sum_{x \in U} \frac{1}{2^{\log(1/p(x))}} \;=\; \sum_{x \in U} p(x) \;=\; 1\,.$$

Also, the expected number of bits used is

$$\sum_{x \in U} p(x) \cdot \lceil \log(1/p(x)) \rceil \;\leq\; \sum_{x \in U} p(x) \cdot (\log(1/p(x)) + 1) \;=\; H(X) + 1\,.$$

This code is known as the Shannon code.

Of course the upper bound above would not be very impressive if it was possible to do much better using some other code. However, we will now show that *any* prefix-free code must use at least $H(X)$ on average.

Claim 3.3 *Let X be a random variable taking values in U and let C be a prefix-free code for U. The the expected number of bits used by C to communicate the value of X is at least $H(X)$.*

Proof: The expected number of bits used is $\sum_{x \in U} p(x) \cdot |C(x)|$. We consider the quantity

$$\begin{aligned} H(X) - \sum_{x \in U} p(x) \cdot |C(x)| &= \sum_{x \in U} p(x) \cdot \left(\log \left(\frac{1}{p(x)} \right) - |C(x)| \right) \\ &= \sum_{x \in U} p(x) \cdot \log \left(\frac{1}{p(x) \cdot 2^{|C(x)|}} \right) . \end{aligned}$$

We consider a random variable Y with takes the value $\frac{1}{p(x) \cdot 2^{|C(x)|}}$ with probability $p(x)$. The above expression then becomes $\mathbb{E}[\log(Y)]$. Using Jensen's inequality gives

$$\mathbb{E}[\log(Y)] \leq \log(\mathbb{E}[Y]) = \log \left(\sum_{x \in U} p(x) \cdot \frac{1}{p(x) \cdot 2^{|C(x)|}} \right) = \log \left(\sum_{x \in U} \frac{1}{2^{|C(x)|}} \right)$$

which is non-positive since $\sum_{x \in U} \frac{1}{2^{|C(x)|}} \leq 1$ by Proposition 3. ■