Information and Coding Theory **Winter 2021**

Lecture 1: January 12, 2021

Lecturer: 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 (60 % of the grade) and a final (40 %). The homeworks will be posted on the course homepage and announced in class, and will be due about one week after they are posted.
- The pre-requisites for the course are familiarity with discrete and continuous probability and random variables, and algorithmic notions. 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.
- The lectures will be held via Zoom, and discussions and office hours will be held using the TTIC Gather space. Please see the canvas page of the course for all access information.
- Video recordings of the lectures will also be posted to the canvas page for the course. Please check the course webpage regularly for updates.

2 A quick reminder about random variables and convexity

2.1 Random variables

Let Ω be a finite set. Let $\mu : \Omega \to [0,1]$ be a function such that

$$\sum_{\omega \in \Omega} \mu(\omega) \;=\; 1.$$

We often refer to Ω as a sample space and the function μ as a probability distribution on this space. When Ω is not finite, μ may need to be replaced by an object called a probability measure (we will discuss this later). Ω and μ are together said to define a probability space (for infinite Ω, pobability spaces need an additional component called a σ-algebra).

A real-valued random variable over Ω is any function $X : \Omega \to \mathbb{R}$. We define

$$\mathbb{E}[X] \;=\; \sum_{\omega \in \Omega} \mu(\omega) \cdot X(\omega)\,.$$

We will also think of a random variable X as given by its distribution. If $\mathcal{X}$ is the (finite) set of values taken by X, we can think of the probability distribution on $\mathcal{X}$ given by

$$p(x) \;=\; \mathbb{P}[X = x] \;=\; \sum_{\omega : X(\omega) = x} \mu(\omega)\,,$$

for all values $x \in \mathcal{X}$.

A word on notation

Note that a random variable X is defined simply as a function on Ω, and the distribution of X is induced by the distribution (or measure) μ on Ω. We use the notation $P(X)$ to denote the distribution P for the random variable X. Note that changing the underlying probability space can result in a different distribution (say) $Q(X)$ for the same function X.

In information theory notation, it is common to only talk of distributions P, Q, if they are for the same X which is clear from context. Similarly, it is common to define quantities (such as entropy) which depend on the *distribution*, simply in terms of random variables X, Y when the underlying probability space is fixed. When we need to talk of multiple random variables, and also of multiple distributions for the same variable X, we will use the more explicit notation $P(X)$.

We will use uppercase letters X, Y, Z for random variables, lowercase letters x, y, z to denote values for these random variables, and caligraphic letters $\mathcal{X}, \mathcal{Y}, cZ$ to denote the sets of *possible* values for random variables (also known as the support of a random variable). We will also use uppercase letters P, Q to denote the names of distributions, and lowercase letters p, q to denote probabilities. Thus, a random variable X with distribution $P(X)$ and support $\mathcal{X}$ may satisfy that for a specific value $x \in \mathcal{X}$, we have $p(x) := \mathbb{P}[X = x] = 1/2$.

2.2 Convexity and Jensen's inequality

A set $S \subset \mathbb{R}^n$ is said to be convex subset of $\mathbb{R}^n$ if the line segment joining any two points in S lies entirely in S i.e., for all $x, y \in S$ and for all $\alpha \in [0, 1]$, $\alpha \cdot x + (1 - \alpha) \cdot y \in S$. For a convex set $S \subseteq \mathbb{R}^n$, a function $f : S \to \mathbb{R}$ is said to be a convex function on S, if for all $x, y \in S$ and for all $\alpha \in [0,1]$, we have

$$f(\alpha \cdot x + (1-\alpha) \cdot y) \;\leq\; \alpha \cdot f(x) + (1-\alpha) \cdot f(y)\,.$$

Equivalently, we say that the function f is convex if the set $S_f = \{(x,z) \mid z \geq f(x)\}$ is a convex subset of $\mathbb{R}^{n+1}$. f is said to be strictly convex when the inequality above is strict for all x, y, α. A function which satisfies the opposite inequality i.e., for all $x, y \in S$ and $\alpha \in [0,1]$

$$f(\alpha \cdot x + (1-\alpha) \cdot y) \;\geq\; \alpha \cdot f(x) + (1-\alpha) \cdot f(y)\,,$$

is said to be a concave function (and strictly concave if the inequalities are strict). Note that if f is a convex function then $-f$ is a concave function (and vice-versa). For a single variable function $f : \mathbb{R} \to \mathbb{R}$ which is twice differentiable, we can also use the easier criterion that f is convex on $S \subseteq \mathbb{R}$ if and only if $f''(x) \geq 0$ for all $x \in S$. We will frequently use the following inequality about convex functions.

Lemma 2.1 (Jensen's inequality). *Let $S \subseteq \mathbb{R}^n$ be a convex set and let X be a random variable taking values only inside S. Then, for a convex function $f : S \to \mathbb{R}$, we have that*

$$\mathbb{E}[f(X)] \;\geq\; f(\mathbb{E}[X])\,.$$

Equivalently, for a concave function $f : S \to \mathbb{R}$, we have

$$\mathbb{E}[f(X)] \;\leq\; f(\mathbb{E}[X])\,.$$

Note that the definition of convexity is the same as the statement of Jensen's inequality for a random variable taking only two values: x with probability α and y with probability $1-\alpha$. You can try the following exercises to familiarize yourself with this inequality.

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

Exercise 2.3. *Check that the $f(x) = x^2$ is a convex function on $\mathbb{R}$. Also show that the functions $\log(x)$ and $x\log(x)$ are respectively, concave and convex functions on $(0,\infty)$.*

Exercise 2.4. *Prove the Cauchy-Schwarz inequality using Jensen's inequality.*

3 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 $\mathcal{X} = \{a_1, \ldots, a_n\}$ such that it takes value $x \in \mathcal{X}$ with probability $p(x)$. Then, we define the *entropy* of the random variable X as

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

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

Proposition 3.1. *Let X be a random variable supported on a finite set $\mathcal{X}$ as above. Then*

$$0 \;\le\; H(X) \;\le\; \log(|\mathcal{X}|).$$

Proof: Since $p(x) \le 1$ we have $\log(1/p(x)) \ge 0$ for all $x \in \mathcal{X}$ and hence $H(X) \ge 0$. 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

$$\begin{aligned} \sum_{x \in \mathcal{X}} p(x) \cdot \log\left(\frac{1}{p(x)}\right) &= \mathbb{E}[\log(Y)] \\ &\le \log(\mathbb{E}[Y]) \\ &= \log\left(\sum_{x \in \mathcal{X}} p(x) \cdot \frac{1}{p(x)}\right) \;=\; \log(|U|). \end{aligned}$$

■

4 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 4.1. *A code for a set $\mathcal{X}$ over an alphabet Σ is a map $C : \mathcal{X} \to \Sigma^*$ which maps each element of $\mathcal{X}$ to a finite string over the alphabet Σ. We say that a code is* prefix-free *if for any $x, y \in \mathcal{X}$ 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 4.2 (Kraft's inequality). *Let $|\mathcal{X}| = n$. There exists a prefix-free code for $\mathcal{X}$ over $\{0,1\}$ with codeword lengths $\ell_1, \ldots, \ell_n$ if and only if*

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

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

Proof: Let us prove the "if" part first. Given $\ell_1, \ldots, \ell_n$ satisfying $\sum_i 2^{-\ell_i} \leq 1$, we will construct a prefix-free code C with these codeword lengths. Without loss of generality, we can assume that $\ell_1 \leq \ell_2 \leq \cdots \leq \ell_n = \ell^*$.

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 $\ell_1, \ldots, \ell_n$ 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 $1/2^{\ell_1}$ 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 choose s_2 as the second codeword in our code C. We can similarly proceed to choose other codewords. At each step, we have some leaves left in the tree since $\sum_i 2^{-\ell_i} \leq 1$.

Note that 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.

For the "only if" part, we can simply reverse the above proof. Let C be a given prefix-free code with codeword lengths $\ell_1, \ldots, \ell_n$ and let $\ell^* = \max\{\ell_1, \ldots, \ell_n\}$. Considering again the complete binary tree of depth ℓ^*, we can now locate the codewords (say) $C(x_1), \ldots, C(x_n)$ as nodes in the tree. We say that a codeword $C(x)$ *dominates* a leaf L if L occurs in the subtree rooted at $C(x)$. Note that the out of the total 2^{ℓ^*} fraction of leaves dominated by a codeword of length ℓ_i is $2^{-\ell_i}$. Also, note that if $C(x)$ and $C(y)$ dominate the same leaf L, then either $C(x)$ appears in the subtree rooted at $C(y)$ or vice-versa. Since the code is prefix-free, this cannot happen and the sets of leaves dominated by codewords must be disjoint. Thus, we have $\sum_i 2^{-\ell_i} \leq 1$.

This part of the proof also has a probabilitic interpretation. Consider an experiment where we generate ℓ^* random bits. For $x \in \mathcal{X}$, 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

$$1 \;\geq\; \sum_{x \in \mathcal{X}} \mathbb{P}\left[E_x\right] \;=\; \sum_{x \in \mathcal{X}} \frac{1}{2^{|C(x)|}} \;=\; \sum_{i=1}^{n} \frac{1}{2^{\ell_i}}\,.$$

■