10-704: Information Processing and Learning **Spring 2012**

Lecture 7: Prefix codes, Kraft-McMillian Inequality

Lecturer: Aarti Singh *Scribes: Aaditya Ramdas*

Disclaimer: *These notes have not been subjected to the usual scrutiny reserved for formal publications. They may be distributed outside this class only with the permission of the Instructor.*

7.1 Recap

7.1.1 Achievability of data compression limit via Typicality

In the previous lecture, we showed that Shannon constructed a code, which was a one-to-one mapping, that took a stream of data $X^n = (X_1, ..., X_n)$ generated iid from a distribution $P(X)$ over a finite alphabet $\mathcal{A} = (a_1, ..., a_A)$ of size A, and compressed it using $\approx nH(X)$ bits in total or $\approx H(X)$ bits per symbol, on average (for sufficiently large n). The code was based considering a special subset of all sequences of size n called the *typical set* $\mathcal{A}_\epsilon^n$, which has very few sequences compared to the size, A^n, of all possible sequences of length n, but contained a large proportion $1-\epsilon$ of the probability mass.

The idea was that since the typical set is very small, we can encode any such typical sequence using much fewer $\log(|\mathcal{A}_\epsilon^n|) \approx nH(X) << n \log A$ bits, and the elements outside the typical set appear so rarely (with probability ϵ), that a lazy encoding using $n \log A$ bits would hardly affect the total number of bits used. Note that encoding sequences using $\lceil \log A \rceil$ bits per symbol is trivial, but this might be much larger than $H(X)$, so Shannon's proofs were special.

We also showed some important properties of the typical set, like

- $P(X^n \in \mathcal{A}_\epsilon^n) \to 1$ (the typical set contains nearly all the mass)
- $|\mathcal{A}_\epsilon^n| \approx 2^{nH(X)}$ (the size of the typical set is small if X has low entropy)
- $P(X^n : X^n \in \mathcal{A}_\epsilon^n) \approx 2^{-nH(X)}$ (all sequences in the typical set are nearly equiprobable).

Note that the worst case is when P is a uniform distribution over $\mathcal{A}$, and in this case the typical set is nearly the whole set, the size of the typical set is largest and equals $\approx 2^{n \log A} = A^n$, and every sequence is equiprobable even before compression, and hence cannot be compressed (no information to be gained).

Since we cannot hope to compress a uniform source (to fewer than the naive $\log A$ bits/symbol), lets consider compressors that map all but a negligible probability mass of sequences to a uniform distribution, i.e.

$$\begin{array}{llll} \text{with probability } 1-\delta, & Z \sim \text{ uniformly random } m_\delta \text{ bits where } m_\delta \ll n & \text{and} & Y = Z \\ \text{with probability } \delta, & Z \sim \text{ uniformly random } n \text{ bits} & \text{and} & Y = \lambda \text{ (a constant)} \end{array}$$

Then a trivial estimator of Z is $\widehat{Z} = Y$, and the corresponding probability of error $P_e = \delta$. Using Fano's lemma, we see that no other estimator can have smaller probability of error.

$$P_e \geq (H(X|Y) - 1)/n = \delta - 1/n$$

since $H(X|Y) = \delta H(X|Y = \lambda) + (1-\delta) H(X|Y = X) = \delta n + (1-\delta) \cdot 0 = \delta n$.

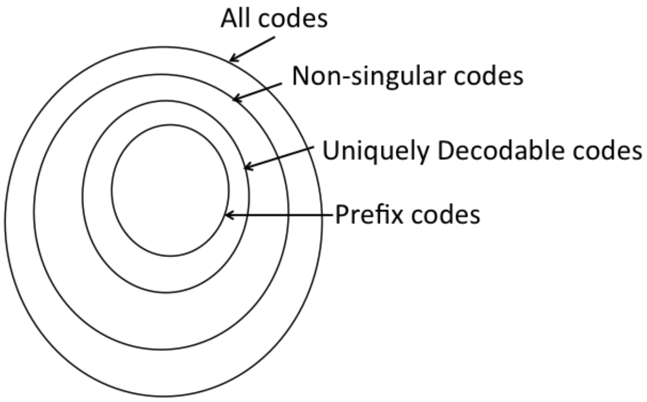
The expected length of this code is minimized if m_δ is smallest for a given δ. This in turn is equivalent to minimizing the size of the set containing the sequences with uniformly random m_δ bits. In last class, we showed that all high probability sets have size asymptotically of order 2^{nH}, hence the smallest m_δ can be (and hence the smallest the expected length of the code can be) is nH. This provides a rough justification that we cannot hope to compress a sequence of length n from a source to fewer than nH bits on average. Later we will see a rigorous proof of this for the class of uniquely decodable codes.

Since the coding scheme mentioned above requires maintaining lookup tables of size equal to the size of the typical set, which is exponential in n, we need to look for more practical codes that are easy to encode and decode.

We then talked about the basics of *symbol coding*, where we consider properties of the code $C(X)$ of single symbol X which is drawn using P from the alphabet $\mathcal{A}$. We then define the extension of the code using concatenation as $C(X^n) = C(X_1)...C(X_n)$ We looked at several appealing properties that such symbol codes could possibly have, such as

- Non-singularity : $\forall X_1, X_2 \in \mathcal{A}, X_1 \neq X_2 \Rightarrow C(X_1) \neq C(X_2)$
- Unique-decodability : $\forall X_1^n \in \mathcal{A}^n, \forall X_2^m \in \mathcal{A}^m, X_1^n \neq X_2^m \Rightarrow C(X_1^n) \neq C(X_2^m)$
- Self-punctuating : $\forall X_1, X_2 \in \mathcal{A}, X_1 \neq X_2 \Rightarrow C(X_1) \neq Prefix(C(X_2))$

Note that unique decodability implies non-singularity and self-punctuating implies unique decodability. Self-punctuating codes are also called **instantaneous or prefix codes**. For unique decodability, we may need to see the entire sequence to decode it uniquely, but for instantaneous ones, you can decode a symbol as soon as you've seen its encoding.



7.2 Kraft-McMillan's Inequality and Optimal Symbol Codes

The kind of questions we want to answer in this lecture relate to how to construct a symbol code having some of the above good properties with minimum expected length of the code. For the rest of the lecture, we shall assume that the encoding is not in binary, but in a general D-ary code (uses D symbols $[D] = \{0, 1, ..., D-1\}$ instead of 2 symbols $\{0, 1\}$).

As we saw in an earlier lecture, Shannon constructed a block code achieving per-symbol codelength of nearly the entropy. If we only want non-singular symbol codes (which is quite weak), we can even beat the entropy limit. For example, if $P(a_1, a_2, a_3, a_4) = (0.5, 0.25, 0.125, 0.125)$ and a non-singular binary symbol code $C(a_1) = 0, C(a_2) = 1, C(a_3) = 00, C(a_4) = 01$ then we have entropy $H(P) = 0.5 \log 2 + 0.25 \log 4 + 2{*}0.125 \log 8 = 1.75$, but expected per-symbol codelength $E[l(C)] = 0.5{*}1{+}0.25{*}1{+}0.125{*}2{+}0.125{*}2 = 1.25$. However, clearly, on receiving a stream 00001, there are many ways of decoding it (like $a_3a_3a_2$ or $a_1a_3a_4$) making the code quite impractical. So we would want additional properties like prefix-ness, while being able to infer something about minimum expected codelength.

Which codelengths are permissible if we restrict the codewords to be prefix codes? The Kraft inequality provides the answer:

Theorem 7.1 (Kraft Inequality) *Assume that a particular symbol coding scheme C uses encoding codelengths $(l_1, ..., l_A)$ where $a_i \in \mathcal{A}$ is encoded by l_i D-ary symbols (i.e. $C(a_i)$ has length l_i). If C is a prefix code, then*

$$\sum_{i=1}^{A} D^{-l_i} \leq 1.$$

Conversely, given a set of integer codelengths that satisfy the above inequality, we can construct a prefix code C' with these codelengths.

Proof: First note that any prefix code can be represented by a D-ary tree T_D (where each non-leaf node has D children) where the A symbols of $\mathcal{A}$ appear only at leaves of T_D (the encoding is given by reading off the path from the root to the leaf). The reason that symbols can only occur at leaves is as follows.

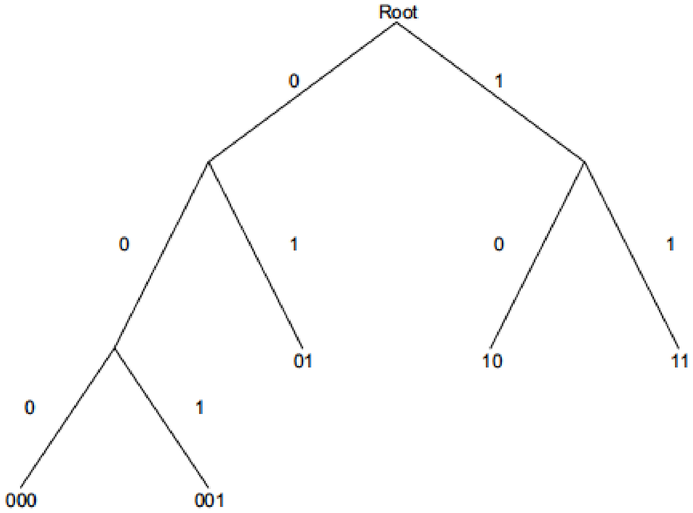


Figure 7.1: A binary prefix tree corresponding to a prefix code for 4 symbols.

Consider a symbol a, which occurs at a node n at depth k in the tree. Its D-ary encoding of length k is given by $C(a) = \mathbf{d} = d_1d_2...d_k$ ($d_i \in [D]$ depends on the path from the root to n). Codes that are prefixes of $\mathbf{d}$ are those corresponding to nodes on the path from the root to n. So, none of those nodes can correspond to symbols because their codes would be prefixes of the code for a. Similarly, none of the descendants of n can ever correspond to any symbol, because $\mathbf{d}$ would be a prefix of that symbol. Hence, we see that only leaves (at possibly different depths) can possibly correspond to symbols in this D-ary tree.

Let l_{max} be the length of the longest encoded symbol, and so the number of possible leaves at depth l_{max} is $D^{l_{max}}$. A codeword leaf at depth i ensures that its $D^{l_{max}-l_i}$ descendant leaves are ruled out. But, all the symbols together cannot possibly rule out more than $D^{l_{max}}$ leaves. So $\sum_{i=1}^{A} D^{l_{max}-l_i} \leq D^{l_{max}}$. Done!

For more intuition, think of dropping one gram of gold dust at the root, at the top of the tree. Let this gold dust distribute downwards evenly (if x grams reaches an internal node, x/D grams will go to each of its children) and this continues till all the gold is at the leaves (no gold can accumulate in the interior nodes, it always drops down evenly). It is clear that if a leaf is at depth k, it would accumulate D^{-k} grams of gold.

What's the total amount of gold at the leaf-symbols? $\sum_{i=1}^{A} D^{-l_i}$ (because if a symbol's encoding is of length l_i then it will be found at depth l_i in the tree). What's the total amount of gold that we started with? One gram. Kraft's inequality just asserts the obvious fact that the total amount of gold at the leaf-symbols cannot exceed the amount that we started with (the inequality is because not all leaves need to correspond to symbols, and the summation was not over all leaves).

The converse can also be thought of in the same way. If the inequality is obeyed by a set of lengths, just imagine putting those symbols at the appropriate depths (equal to their lengths) in a D-ary tree. The reason you can always do this such that symbols only appear at leaves, hence giving us a prefix code, can be imagined using the gold dust argument again (since obeying the inequality is like obeying the law of conservation of gold dust over prefix code D-ary trees). ■

Can we gain anything, i.e. allow a larger collection of codelengths and hence hope to get smaller expected codelength, by giving up prefix property and only requiring the codes to be uniquely decodable? The following theorem due to McMillan says the answer is no.

Theorem 7.2 *(McMillan) Assume that a particular symbol coding scheme C uses encoding codelengths $(l_1, ..., l_A)$ where $a_i \in \mathcal{A}$ is encoded by l_i D-ary symbols (i.e. $C(a_i)$ has length l_i). If C is a uniquely decodable code, then*

$$\sum_{i=1}^{A} D^{-l_i} \leq 1.$$

Conversely, given a set of integer codelengths that satisfy the above inequality, we can construct a uniquely decodable code C' with these codelengths.

Proof: Let C_k be the k-th extension (as mentioned earlier in the lecture) of the uniquely decodable code C, i.e. the extension of C from symbols to sequences of length k by concatenating the codes for individual symbols (and hence $l(C_k(X^k)) = \sum_i l(C(X_i))$.

Since $\forall l, C_l$ is uniquely decodable, if n_l is the number of sequences whose codewords have length l and D^l is the total possible number of codewords of length l, $n_l \leq D^l$, because otherwise by the pigeonhole principle, at least two sequences would map to the same codeword and it would not be uniquely decodable. This is the inequality that does not hold for non-singular codes, since each of the D^l possible encoded codewords of length l could correspond to the encoding of many sequences, allowing n_l to be much larger than D^l.

Let $l_{max} = \max_{X \in \mathcal{A}} |C(X)|$ be the length of the longest code for a symbol. Here $\sum_X$ represents summing over the A different values that X can take from $\mathcal{A}$, $\sum_{X^k}$ represents summing over sequences of length k, and $\sum_l$ represents summing over the different lengths that sequence codes can have (where the longest sequence code is of length kl_{max} for sequences of length k).

Consider the quantity $\left(\sum_X D^{-l(C(X))}\right)^k$
$= (\sum_{X_1} D^{-l(C(X_1))})(\sum_{X_2} D^{-l(C(X_2))})...(\sum_{X_k} D^{-l(C(X_k))})$
$= \sum_{X_1} \sum_{X_2} ... \sum_{X_k} D^{-l(C(X_1))} D^{-l(C(X_2))} ... D^{-l(C(X_k))}$
$= \sum_{X^k} D^{-l(C_k(X^k))}$
$= \sum_l n_l D^{-l}$
$\leq \sum_l D^l D^{-l}$
$= kl_{max}$

This yields $\sum_X D^{-l(C(X))} \leq (kl_{max})^{1/k}$.

The key step is to note is that instead of summing over sequences X^k, we can group sequences by their encoded codelength (n_l such sequences exist for each l) and sum over different possible encoded lengths of sequences instead. Now, since the left hand side doesn't depend on k and this above inequality is true for all k (including arbitrarily large k), by noting that for any constant c, $\lim_{k\to\infty}(ck)^{1/k} = 1$, we must have $\sum_X D^{-l(C(X))} \leq 1$, or $\sum_i D^{-l_i} \leq 1$, establishing the inequality.

The converse is trivially true, since given such codelengths l_i satisfying the inequality, we can construct a prefix code using the previous theorem, and all prefix codes are uniquely decodable. ■

We can now characterize the minimum expected length of any uniquely-decodable code.

Theorem 7.3 *Any uniquely decodable symbol coding scheme (and hence any prefix code) using D-ary codes has an expected encoded length of a random symbol greater than or equal to the entropy $H_D(P)$ where P is the distribution by which symbols are randomly drawn from $\mathcal{A}$.*

Proof: We consider the convex minimization problem $\min_{l_i} \sum_i p_i l_i$ subject to the convex unique decodability (or prefix code) constraint $\sum_i D^{-l_i} \leq 1$. Differentiating the Lagrangian $\sum_i p_i l_i + \lambda \sum_i D^{-l_i}$ with respect to l_i and noting that at the global minimum (λ^*, l_i^*) it must be zero, we get :

$p_i - \lambda^* D^{-l_i^*} \ln D = 0$ which implies that $D^{-l_i^*} = \frac{p_i}{\lambda^* \ln D}$

Using complementary slackness, noting that $\lambda^* > 0$ for the above condition to make sense, we have :

$\sum_i \frac{p_i}{\lambda^* \ln D} = 1$ which implies $\lambda^* = 1/\ln D$ and hence $D^{-l_i^*} = p_i$, or the optimum length $l_i^* = \log_D(1/p_i)$ (the Shannon information content).

This gives the expected minimum codelength for uniquely decodable codes as $\sum_i p_i l_i^* = \sum_i p_i \log_D(1/p_i) = H_D(P)$. ■