CSC 310: Information Theory **Winter 2026**

Tutorial 2: Review of Symbol Codes and Huffman Codes

Instructor: Swastik Kopparty *TA: Ziyang Jin*

Date: 14 Jan 2026

2.1 Review of Symbol Codes

Recall the definition of symbol codes that we learned in class.

Definition 2.1 (Symbol Codes) *A symbol code C for a discrete random variable X is a mapping from $\mathcal{X}$, the range of X, to $\{0,1\}^*$. For each $x \in \mathcal{X}$, let $C(x)$ denote the codeword corresponding to x, and let $l(x)$ denote the length of the codeword.*

Definition 2.2 (Expected Length) *The expected length, denoted by $L(C)$, of a symbol code C for a discrete random variable X with range $\mathcal{X}$ and probability mass function p is given by*

$$L(C) = \mathbb{E}[|C(X)|] = \sum_{x\in\mathcal{X}} p(x)l(x).$$

For a string $X_1 \dots X_n$ where each X_i for $i \in [n]$ is an independent copy of X, we compress it by writing

$$C(X_1)C(X_2)\dots C(X_n) \in \{0,1\}^*.$$

The expected length of the compressed string $C(X_1)C(X_2)\dots C(X_n)$ will be $n \cdot L(C) = n \cdot \mathbb{E}[|C(X)|]$. Note that symbol code is one of many approaches to compressing the random source $X_1 \dots X_n$, and it may not be optimal. We claim that $n \cdot \mathbb{E}[|C(X)|] \geq n \cdot H(X)$ and it may be the case that $n \cdot \mathbb{E}[|C(X)|] = n \cdot (H(X) + \epsilon)$ for some small $\epsilon > 0$, which means the compression can possibly have ϵn excess bits than truly required.

For some sources, symbol codes can never reach $n \cdot H(X)$. Consider the following two examples.

Example 1. Consider a discrete random variable X with alphabet $\mathcal{X} = \{a, b, c, d, e\}$, where

$$X = \begin{cases} a & \text{w.prob. } \frac{1}{4}, \\ b & \text{w.prob. } \frac{1}{4}, \\ c & \text{w.prob. } \frac{1}{4}, \\ d & \text{w.prob. } \frac{1}{8}, \\ e & \text{w.prob. } \frac{1}{8}. \end{cases}$$

Consider symbol code $C : \mathcal{X} \to \{0,1\}^*$ defined as follows:

$$C(a) = 00,\ C(b) = 01,\ C(c) = 10,\ C(d) = 110,\ C(e) = 111.$$

The entropy of X is

$$H(X) = 3 \times \frac{1}{4}\log 4 + 2 \times \frac{1}{8}\log 8 = \frac{3}{2} + \frac{3}{4} = \frac{9}{4}.$$

The expected length of the symbol code C is

$$L(C) = \sum_{x\in\mathcal{X}} p(x)l(x) = 3 \times \frac{1}{4} \times 2 + 2 \times \frac{1}{8} \times 3 = \frac{9}{4}.$$

Suppose we have n independent copies of X, denoted by $X_1 \dots X_n$. Using the symbol code C, we can see that $n \cdot L(C) = n \cdot \mathbb{E}[|C(X)|] = n \cdot H(X)$, which means that the compression is optimal.

Example 2. Consider a discrete random variable Y with alphabet $\mathcal{Y} = \{a, b, c\}$, where

$$Y = \begin{cases} a & \text{w.prob. } \frac{1}{3}, \\ b & \text{w.prob. } \frac{1}{3}, \\ c & \text{w.prob. } \frac{1}{3}. \end{cases}$$

Consider symbol code $E : \mathcal{Y} \to \{0,1\}^*$ defined as follows:

$$E(a) = 0,\ E(b) = 10,\ E(c) = 11.$$

The entropy of Y is

$$H(Y) = 3 \times \frac{1}{3}\log 3 = \log 3 \approx 1.585.$$

The expected length of the symbol code E is

$$L(E) = \sum_{y\in\mathcal{Y}} p(y)l(y) = 1 \times \frac{1}{3} \times 1 + 2 \times \frac{1}{3} \times 2 = \frac{5}{3} \approx 1.667 \geq \log 3.$$

Suppose we have n independent copies of Y, denoted by $Y_1 \dots Y_n$. Using the symbol code E, we can see that $n \cdot L(C) = n \cdot \mathbb{E}[|C(Y)|] > n \cdot H(Y)$, which means that the compression is not optimal (i.e. not matching the entropy bound).

2.2 Quiz Time

We will do a 10 minute quiz during the tutorial. If you are previewing this tutorial scribe notes, please stop here before taking the quiz such that you will not be confused.

2.3 Huffman Codes

We present the Huffman coding algorithm, a greedy algorithm that produces an optimal prefix-free code. Huffman codes are also optimal uniquely decodable codes. At a high level, Huffman's algorithm builds the encoding tree bottom-up.

Definition 2.3 (Huffman Coding Algorithm) *Let X be a discrete random variable with range $\mathcal{X} = \{x_1, \dots, x_m\}$ and corresponding probabilities $p(x_1), \dots, p(x_m)$. Repeat the following procedure until only one (super)symbol is left.*

1. *Take the two least probable symbols in the alphabet. Let x_i, x_j denote these two symbols with probability $p(x_i), p(x_j)$ respectively. (These two symbols will be given the longest codewords, which will have equal length, and differ only in the last digit.)*
2. *Combine these two symbols into a supersymbol x' with corresponding probability equal to $p(x_i)+p(x_j)$, and update the alphabet by replacing x_i, x_j with x'.*
3. *Build a binary tree with x_i, x_j as children and x' as parent.*

In the end, we assign the codewords based on the binary tree built.

Let's run Huffman's coding algorithm with some examples:

- Consider discrete random variable X with range $\{a, b, c\}$, where

$$X = \begin{cases} a, & \text{w.prob. } \frac{1}{3}; \\ b, & \text{w.prob. } \frac{1}{3}; \\ c, & \text{w.prob. } \frac{1}{3}. \end{cases}$$

  First take symbols b, c (you can choose a, b or a, c too and get a different but equally optimal encoding) and combine them to a supersymbol bc with probability $\frac{1}{3} + \frac{1}{3} = \frac{2}{3}$. The new alphabet is $\{a, bc\}$ with probability $\frac{1}{3}, \frac{2}{3}$ respectively. Then, combine a and bc to a supersymbol abc with probability $\frac{1}{3} + \frac{2}{3} = 1$. Let C be the Huffman code. By examining the binary tree we built, we have $C(a) = \texttt{0}, C(b) = \texttt{10}, C(c) = \texttt{11}$. Thus, the lengths are $l(a) = 1, l(b) = 2, l(c) = 2$. The expected length is

$$L(C) = \frac{1}{3} \times 1 + \frac{1}{3} \times 2 + \frac{1}{3} \times 2 = \frac{5}{3} \approx 1.667.$$

  Entropy of X is

$$H(X) = \frac{1}{3} \log\left(\frac{1}{\frac{1}{3}}\right) + \frac{1}{3} \log\left(\frac{1}{\frac{1}{3}}\right) + \frac{1}{3} \log\left(\frac{1}{\frac{1}{3}}\right) = \log 3 \approx 1.585.$$

- Consider discrete random variable Y with range $\{a, b, c\}$ where:

$$Y = \begin{cases} a, & \text{w.prob. } \frac{1}{2}; \\ b, & \text{w.prob. } \frac{1}{3}; \\ c, & \text{w.prob. } \frac{1}{6}. \end{cases}$$

  First take symbols b, c (this is the only choice) and combine them to a supersymbol bc with probability $\frac{1}{3} + \frac{1}{6} = \frac{1}{2}$. The new alphabet is $\{a, bc\}$ with probability $\frac{1}{2}, \frac{1}{2}$ respectively. Then, combine a and bc to a supersymbol abc with probability $\frac{1}{2} + \frac{1}{2} = 1$. Let C be the Huffman code, and we have $C(a) = \texttt{0}, C(b) = \texttt{10}, C(c) = \texttt{11}$. Thus, the lengths are $l(a) = 1, l(b) = 2, l(c) = 2$. The expected length is

$$L(C) = \frac{1}{2} \times 1 + \frac{1}{3} \times 2 + \frac{1}{6} \times 2 = \frac{3}{2} = 1.5.$$

  Entropy of Y is

$$H(Y) = \frac{1}{2} \log\left(\frac{1}{\frac{1}{2}}\right) + \frac{1}{3} \log\left(\frac{1}{\frac{1}{3}}\right) + \frac{1}{6} \log\left(\frac{1}{\frac{1}{6}}\right) = \frac{1}{2} + \frac{1}{3} \log 3 + \frac{1}{6} \log 6 \approx 1.459.$$

We do not have time to cover the proof of optimality of Huffman Code during this tutorial. Below is a reference for anyone who is interested.

Theorem 2.4 *Huffman codes are optimal prefix-free codes.*

Proof: We will only sketch the proof. Recall that a code is optimal when $L(C) := \sum_{x\in\mathcal{X}} p(x)l(x)$ is minimized. Without loss of generality, suppose random variable X has alphabet $\mathcal{X} = \{1, 2, ..., m\}$ with probabilities $p(1) \geq p(2) \geq ... \geq p(m)$. There are two major steps to prove this theorem.

Step 1. We show that there exists an optimal code that satisfies the following properties:

1. The lengths are ordered inversely with the probabilities, i.e., $l(1) \leq l(2) \leq ... \leq l(m)$.
2. The two longest codewords have the same length, i.e., $l(m-1) = l(m)$.
3. The two longest codewords differ only in the last bit and correspond to the two least likely symbols, i.e., $C(m-1)$ and $C(m)$ differ only in the last bit.

Note that all optimal code should satisfy 1 and 2. There is an optimal code satisfying 3 that is equally good with other optimal code.

For 1, suppose an optimal code C has $p(j) > p(k)$ but $l(j) > l(k)$. Define a new code C' that switches the codeword $C(j)$ and $C(k)$. Then we have $L(C) - L(C') = p(j)l(j) + p(k)l(k) - p(j)l(k) - p(k)l(j) = (p(j) - p(k))(l(j) - l(k)) > 0$. Contradiction!

For 2, if not, the last bit of the longer codeword can be eliminated, which is a code with shorter expected length. By property 1, the longest codewords must belong to the least probable source symbols.

For 3, not all optimal codes satisfy this property, but by rearranging, we can find an optimal code that does. If there is a maximal-length codeword without a sibling, we can delete the last bit of that codeword and get a shorter expected length. Hence, every maximal-length codeword in any optimal code has a sibling. Now we exchange the longest codewords so that the two lowest-probability symbols become siblings. This does not change the expected length.

Step 2. We prove by induction that the expected length of Huffman code is equal to the expected length of an optimal code. The key argument in the proof comes from the following two operations.

1. Let C^m_{opt} be an optimal code for m symbols, where $p(1) \geq p(2) \geq ... \geq p(m)$ and $l(1) \leq l(2) \leq ... \leq l(m)$. We take a *Huffman reduction* on these m symbols, which takes the two least likely symbols and combine them into a supersymbol, with new probabilities $\{p(1),\ p(2),\ ...,\ p(m-2),\ p(m-1) + p(m)\}$. Denote the new code C^{m-1}. For $i \in [m-2]$, $C^{m-1}(i) = C^m_{opt}$. And $C^{m-1}(m-1)$ is the codeword $C^m_{opt}(m)$ without the last bit.
2. Let C^{m-1}_{opt} be an optimal code for $m-1$ symbols with probabilities $\{p(1),\ p(2),\ ...,\ p(m-2),\ p(m-1)+ p(m)\}$. We create a new code C^m for m symbols with probabilities $\{p(1),\ p(2),\ ...,\ p(m-2),\ p(m-1),\ p(m)\}$. For $i \in [m-2]$, $C^m(i) = C^{m-1}_{opt}(i)$. $C^m(m-1)$ is $C^{m-1}_{opt}(m-1)$ appending a `0` in the end; $C^m(m)$ is $C^{m-1}_{opt}(m-1)$ appending a `1` in the end.

For the first operation, the expected length of C^{m-1} is

$$\begin{aligned}
L(C^{m-1}) &= \sum_{i=1}^{m-2} p(i)l(i) + (p(m-1) + p(m))(l(m) - 1) \\
&= \sum_{i=1}^{m} p(i)l(i) - p(m-1)l(m-1) - p(m)l(m) + p(m-1)l(m) + p(m)l(m) - p(m-1) - p(m) \\
&= L(C^m_{opt}) - p(m-1) - p(m).
\end{aligned}$$

Note that we use the property $l(m-1) = l(m)$ to derive the result above. Thus, we have

$$L(C^{m-1}) = L(C^m_{opt}) - p(m-1) - p(m). \tag{2.1}$$

For the second operation, the expected length of C^m is

$$\begin{aligned}
L(C^m) &= \sum_{i=1}^{m} p(i)l(i) \\
&= \sum_{i=1}^{m-2} p(i)l(i) + p(m-1)l(m-1) + p(m)l(m) \\
&= \sum_{i=1}^{m-2} p(i)l(i) + (p(m-1) + p(m))l(m-1) \\
&= L(C^{m-1}_{opt}) - (p(m-1) + p(m))(l(m-1) - 1) + (p(m-1) + p(m))l(m-1) \\
&= L(C^{m-1}_{opt}) + p(m-1) + p(m).
\end{aligned}$$

Thus, we have

$$L(C^m) = L(C^{m-1}_{opt}) + p(m-1) + p(m). \tag{2.2}$$

Adding (2.1) and (2.2), we have

$$L(C^{m-1}) + L(C^m) = L(C^m_{opt}) + L(C^{m-1}_{opt}).$$

Reordering the terms, we obtain

$$(L(C^m) - L(C^m_{opt})) + (L(C^{m-1}) - L(C^{m-1}_{opt})) = 0.$$

Note that C^m_{opt} and C^{m-1}_{opt} are optimal codes, thus $L(C^m) - L(C^m_{opt}) \geq 0$ and $L(C^{m-1}) - L(C^{m-1}_{opt}) \geq 0$. Therefore, we get $L(C^m) = L(C^m_{opt})$ and $L(C^{m-1}) = L(C^{m-1}_{opt})$. This means if we start with an optimal code for $m-1$ symbols with probabilities $\{p(1), p(2), ..., p(m-2), p(m-1)+p(m)\}$ and extend it using Huffman's algorithm, then the new code is also optimal for m symbols with probabilities $\{p(1), p(2), ..., p(m-2), p(m-1), p(m)\}$.

We start with the base case being two symbols. It is trivial that Huffman code is optimal for two symbols. It follows from induction that the Huffman code is optimal. ■