EE276: Homework #3 Solutions

Due on Friday Jan 30, 6pm - Gradescope entry code: E6VP4X

1. **Arithmetic Coding.**

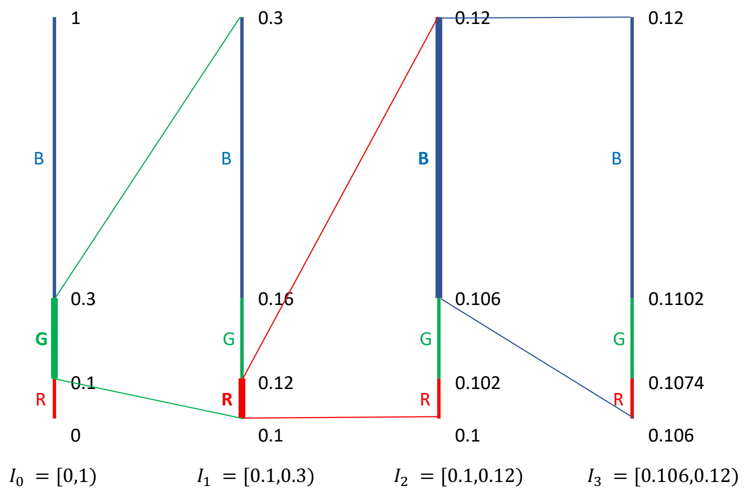

Figure 1: Illustration of arithmetic coding.

Note: Throughout this problem, we will work with digits rather than bits for simplicity. So the logarithms will be base 10 and the compressor will output digits $\{0, 1, \dots, 9\}$.

This problem introduces a simplified version of arithmetic coding, which is itself based on Shannon-Fano-Elias coding. Arithmetic coding takes as input a sequence $x^n \in \mathcal{X}^n$ and a distribution q over $\mathcal{X}$. The encoder maintains an interval which is transformed at each step as follows:

- Start with $I_0 = [0, 1)$.
- For $i = 1, \dots, n$:
  - Divide I_{i-1} into $|\mathcal{X}|$ half-open subintervals $\{I_{i-1}^{(x)},\ x \in \mathcal{X}\}$ with length of $I_{i-1}^{(x)}$ proportional to $q(x)$, i.e., $\left|I_{i-1}^{(x)}\right| = q(x)\,|I_{i-1}|$ for $x \in \mathcal{X}$.
  - Set $I_i = I_{i-1}^{(x_i)}$

Figure 1 shows an example of this for $\mathcal{X} = \{R, G, B\}$, $(q(R), q(G), q(B)) = (0.1, 0.2, 0.7)$ and $x^3 = GRB$. At the end of this process, the encoder selects a number in the interval I_n and outputs the digits after the decimal point for that number. In the example shown, the encoder can output 11, which corresponds to $0.11 \in [0.106, 0.12)$. While 1103 (corresponding to 0.1103) is also a valid output, the encoder tries to output the shortest possible valid sequence. The YouTube video `https://youtu.be/FdMoL3PzmSA` might be helpful for understanding this process even better.

(a) Briefly explain how the decoding might work in a sequential manner. You can assume that the decoder knows the alphabet, the distribution q and the length of the source sequence n.

(b) What is the length of interval I_n in terms of q and x^n?

(c) For the following intervals I_n obtained by following the above-described process for some x^n, find the length of the shortest output sequence (in digits):

i. $[0.095, 0.105)$

ii. $[0.11, 0.12)$

iii. $[0.1011, 0.102)$

In general, if the interval length is l_n, then the shortest output sequence has at most $\left\lceil \log \frac{1}{l_n} \right\rceil$ digits.

(d) Show that the length $l(x^n)$ for the arithmetic encoding output satisfies

$$l(x^n) \leq \log \frac{1}{q(x_1) \dots q(x_n)} + 1$$

(e) Suppose that $X^n \overset{\text{i.i.d.}}{\sim} X$ which has PMF P, and we use arithmetic coding with $q = P$. Then show that

$$\frac{1}{n} E[l(X^n)] \leq H(X) + \frac{1}{n}$$

Compare this to Huffman coding over blocks of length n with respect to compression rate and computational complexity.

(f) Suppose both the encoder and the decoder have a prediction algorithm (say a neural network) that provides probabilities $q_i(x|x^{i-1})$ for all i's and all $x \in \mathcal{X}$. How would you modify the scheme such that you achieve

$$l(x^n) \leq \log \frac{1}{q_1(x_1)q_2(x_2|x_1) \dots q_n(x_n|x^{n-1})} + 1$$

Thus, if you have a prediction model for your data, you can apply arithmetic coding on it - good prediction translating to high probability, in turn translating to short compressed representations.

Solution: Arithmetic Coding.

(a) We first locate the interval $I_0^{(x)}$ containing the output number and decode x_1 as the corresponding x. Then we set $I_1 = I_0^{(x_1)}$ and locate the interval $I_1^{(x)}$ containing the output number to decode x_2. Repeating this n times, we get back x^n.

(b) Length of I_n is $q(x_1) \times q(x_2) \times \cdots \times q(x_n)$.

(c) i. Length 1, output sequence 1 (corresponding to 0.1)

ii. Length 2, output sequence 11 (corresponding to 0.11)

iii. Length 4, output sequence 1011 (corresponding to 0.1011) or 1012, etc.

(d) From part (b), the length of I_n is $q(x_1) \times q(x_2) \times \cdots \times q(x_n)$. Using part (c), we get

$$l(x^n) \leq \left\lceil \log \frac{1}{q(x_1) \ldots q(x_n)} \right\rceil$$

The result follows using the fact that $\lceil x \rceil \leq x + 1$.

(e) Using result from part (d),

$$\begin{aligned} l(X^n) &\leq \log \frac{1}{P(X_1) \ldots P(X_n)} + 1 \\ &= \sum_{i=1}^{n} \log \frac{1}{P(X_i)} \end{aligned}$$

Taking expectation and dividing by n,

$$\begin{aligned} \frac{1}{n} E[l(X^n)] &\leq \frac{1}{n} \sum_{i=1}^{n} E\left[\log \frac{1}{P(X_i)}\right] + \frac{1}{n} \\ &= \frac{1}{n} \sum_{i=1}^{n} H(P) + \frac{1}{n} \\ &= H(P) + \frac{1}{n} \end{aligned}$$

As n becomes large, arithmetic coding approaches entropy, which is the optimal compression rate. Huffman codes also approach the same limit, but are also optimal for any given n (although the gap becomes pretty small as n increases) (note that we can show the same $1/n$ upper bound for Huffman codes). Arithmetic coding has linear complexity in n, but Huffman codes have exponential complexity in n (block length) for storing the codebook.

(f) At step i, instead of dividing I_{i-1} into subintervals with lengths proportional to $q(x)$, we divide into subintervals with lengths proportional to $q_i(x|x^{i-1})$. Then the length of I_n is $q_1(x_1) \times q_2(x_2|x_1) \times \cdots \times q_n(x_n|x^{n-1})$ and the length satisfies the desired bound (using same proof as part d).

2. **Entropy Rate.**
Consider the Markov process from class taking values in $\{H, T\}$ with the joint probability distribution given as

$$P(X_1 = x_1, \ldots, X_n = x_n) = P(X_1 = x_1) \prod_{i=2}^{n} P(X_i = x_i | X_{i-1} = x_{i-1})$$

where $P(X_1 = H) = \frac{1}{2}$, $P(X_i = H|X_{i-1} = H) = \frac{3}{4}$ and $P(X_i = T|X_{i-1} = T) = \frac{3}{4}$ for all $i > 1$.

(a) Directly compute $P(X_2 = H)$ and extend that result to show that the process is stationary (we are only looking for the main idea, no need to write a long proof).

(b) Compute $H(X_n|X_{n-1},\dots,X_1)$ as a function of n and find the limit as $n \to \infty$.

(c) Compute $\frac{1}{n}H(X_1,\dots,X_n)$ as a function of n and find the limit as $n \to \infty$. How does this relate to the result in part (b)?

Solution: Entropy Rate.

(a)

$$\begin{aligned}
P(X_2 = H) &= P(X_1 = T, X_2 = H) + P(X_1 = H, X_2 = H)\\
&= P(X_1 = T)P(X_2 = H|X_1 = T) + P(X_1 = H)P(X_2 = H|X_1 = H)\\
&= \frac{1}{2} \times \frac{1}{4} + \frac{1}{2} \times \frac{3}{4}\\
&= \frac{1}{2}
\end{aligned}$$

Thus, $P(X_2 = H) = \frac{1}{2}$. Now by repeating this exercise, it can be shown that $P(X_n = H) = \frac{1}{2}$ for all n. To show that the process is stationary one needs to show that for any n, k and $(x_1,\dots,x_k) \in \{H,T\}^k$, the following holds:

$$P(X_1 = x_1,\dots,X_k = x_k) = P(X_{n+1} = x_1,\dots,X_{n+k} = x_k)$$

By the definition of the Markov process, we have

$$P(X_1 = x_1,\dots,X_k = x_k) = P(X_1 = x_1)\prod_{i=2}^{k} P(X_i = x_i|X_{i-1} = x_{i-1})$$

Similarly by considering $P(X_1,\dots,X_{n+k})$ and marginalizing over $X_1,\dots,X_n$, we obtain

$$P(X_{n+1} = x_1,\dots,X_{n+k} = x_k) = P(X_{n+1} = x_1)\prod_{i=n+2}^{n+k} P(X_i = x_i|X_{i-1} = x_{i-1})$$

The stationarity follows by observing that two products match termwise.

(b) Note that by the Markov property, X_n is independent of $X_1,\dots,X_{n-2}$ given X_{n-1}. Thus,

$$H(X_n|X_{n-1},\dots,X_1) = H(X_n|X_{n-1})$$

By stationarity, this is same as $H(X_2|X_1)$. The joint distribution of (X_1,X_2) is

$$\begin{aligned}
P(X_1 = H, X_2 = H) &= \frac{1}{2} \times \frac{3}{4}\\
P(X_1 = T, X_2 = H) &= \frac{1}{2} \times \frac{1}{4}\\
P(X_1 = H, X_2 = T) &= \frac{1}{2} \times \frac{1}{4}\\
P(X_1 = T, X_2 = T) &= \frac{1}{2} \times \frac{3}{4}
\end{aligned}$$

and the entropy $H(X_2|X_1)$ is just $h_2(\frac{3}{4})$ by direct computation. Thus $H(X_n|X_{n-1},\dots,X_1) = h_2(3/4)$. Since this does not depend on n, the limit is also $h_2(\frac{3}{4})$.

(c) Using chain rule for entropy

$$\frac{1}{n}H(X_1,\dots,X_n) = \frac{1}{n}H(X_1) + \frac{1}{n}\sum_{i=2}^{n} H(X_i|X_{i-1},\dots,X_1)$$

Using the result from part (b), we get

$$\frac{1}{n}H(X_1,\dots,X_n) = \frac{1}{n} + \frac{(n-1)h_2(3/4)}{n}$$

The limit is $h_2(3/4)$ which matches the limit from part (b). Both parts compute the entropy rate of this process, and the proof of equality in the general case is given in the book C&T Theorem 4.2.1.

3. **Individual Sequences and a Universal Compressor.**
Note: Ignore integer constraints on codeword lengths throughout this problem.
Notation: $h_2(p) = -p\log_2 p - (1-p)\log_2(1-p)$ (= binary entropy function).

Let x^n be a given arbitrary binary sequence, with n_0 0's and n_1 1's ($n_1 = n - n_0$). You are also provided a compressor C_q which takes in any arbitrary distribution q on $\{0,1\}$ as a parameter, and encodes x^n using:

$$\bar{L}_q(x^n) = \frac{1}{n}\log\frac{1}{q(x^n)}$$

bits per symbol where $q(x^n) := \prod_{i=1}^{n} q(x_i)$.

(a) Given the sequence x^n, what binary distribution $q(x)$ will you choose as a parameter to the compressor C_q, so that $\bar{L}_q(x^n)$ is minimized. Your answer (values of $q(0)$ and $q(1)$) will be expressible in terms of n, n_0 and n_1.

(b) When compressing any given individual sequence x^n, we also need to store the parameter distribution $q(x)$ (required for decoding). Show that you can represent the optimal parameter distribution $q(x)$ from part (a) using $\log(n+1)$ bits. You can assume that the decoder knows the length of the source sequence n.

(c) Show that the effective compression rate for compressing x^n (in bits per source symbol) with the distribution q from part (a) is $h_2(n_1/n) + \log(n+1)/n$.

(d) Now suppose that we apply the scheme above to X^n sampled from an i.i.d. $Ber(p)$ distribution. Show that the expected compression rate approaches $h_2(p)$ as $n \to \infty$, i.e., the scheme is a *universal* compressor for i.i.d. sources.

Solution: Individual Sequences and a Universal Compressor.

(a) For $q(0) = 1-q, q(1) = q$, we have

$$\bar{L}_q = \frac{1}{n}\log\frac{1}{(1-q)^{n_0}q^{n_1}} = -\frac{n_0}{n}\log(1-q) - \frac{n_1}{n}\log(q).$$

We see that $\bar{L}_q$ is convex in q, and taking derivative w.r.t q gives $q^* = \frac{n_1}{n}$.

(b) By the previous part, it suffices to store $n_1 \in \{0, 1, \cdots, n\}$ for full knowledge of $q(x)$. Hence, $\log(n+1)$ bits are enough (ignoring integer constraints).

(c) Simply substitute $q = q^*$ in the $\bar{L}_q$ expression in part (a) solution above, and add the contribution from part (b) (normalized by n).

$$\bar{L}_q + \frac{\log(n+1)}{n} = h_2\left(\frac{n_1}{n}\right) + \frac{\log(n+1)}{n}.$$

(d) Let N_1 denote the number of 1's in X^n. Then we get that the expected compression rate (call it R_n) is

$$R_n = \mathbb{E}\left[h_2\left(\frac{N_1}{n}\right)\right] + \frac{\log(n+1)}{n}$$

For the first term, we can use Jensen's inequality and the concavity of entropy to get

$$R_n \le h_2\left(\mathbb{E}\left[\frac{N_1}{n}\right]\right) + \frac{\log(n+1)}{n}$$

Now, $\mathbb{E}\left[\frac{N_1}{n}\right] = p$ which gives us

$$R_n \le h_2(p) + \frac{\log(n+1)}{n}$$

Taking the limit,

$$\lim_{n\to\infty} R_n \le h_2(p)$$

But since the entropy $h_2(p)$ is also a lower bound on any compression scheme, we must have

$$\lim_{n\to\infty} R_n = h_2(p)$$

Thus the scheme is universal for i.i.d. sources.

4. **Elias Coding.**
We will construct *universal* codes for integers that compress any integer valued (hence, infinite alphabet) random variable almost to its entropy.

(a) Consider the following universal compressor for natural numbers: For $x \in \mathbb{N} = \{1, 2, \ldots\}$, let $k(x)$ denote the length of its binary representation. Define the codeword $c(x)$ corresponding to x to be $k(x)$ zeros followed by the binary representation of x. Compute $c(9)$. Show that c is a prefix code and describe a decoding strategy for a stream of codewords.

(b) Now we define another code using the previous one. Define the codeword $c'(x)$ corresponding to x to be $c(k(x))$ followed by the binary representation of x. Compute $c'(9)$. Show that c' is a prefix code and describe a decoding strategy for a stream of codewords.

(c) Let X be a random variable on natural numbers with a decreasing probability mass function. Show that $\mathbb{E}[\log X] \le H(X)$.

(d) Show that the average code length of c satisfies

$$\mathbb{E}[l(c(x))] \le 2H(X) + 2.$$

(e) Show that the average code length of c' satisfies

$$\mathbb{E}[l(c'(x))] \le H(X) + 2\log(H(X) + 1) + 3.$$

Solution: The two coding schemes discussed in this question are known as Elias γ-codes and δ-codes.

(a) We can write $k(x) = \lfloor \log x \rfloor + 1$. The binary representation of $x = 9$ is 1001. Therefore, $k(9) = 4$ and $c(9) = 00001001$. To show c is a prefix code, we need to show that $c(x_1)$ is not a prefix of $c(x_2)$ for any $x_1, x_2 \in \mathbb{N}$. Consider two cases.

If $k(x_1) = k(x_2)$, then $c(x_1)$ and $c(x_2)$ have the same length, but $c(x_1) \neq c(x_2)$ because x_1 and x_2 have different binary representations. Thus $c(x_1)$ and $c(x_2)$ cannot be prefixes of each other. If $k(x_1) \neq k(x_2)$, suppose $x_1 < x_2$, then $c(x_1)$ is no longer than $c(x_2)$. $c(x_1)$ begins with $k(x_1)$ zeros followed by a 1, while the corresponding bits in $c(x_2)$ are all zero. Thus $c(x_1)$ cannot be a prefix of $c(x_2)$.

Since c is a prefix code, we can decode a stream of bits codeword by codeword. To decode the first codeword, the decoder counts the number of 0's before the first 1, which gives $k(x)$. Then the decoder converts the $k(x)$ bits starting from the first 1 into an integer, which gives x. The decoder then repeats the same steps on the next codeword.

(b) $c(k(9)) = c(4) = 000100$. Therefore, $c'(9) = 001001001$. Again, to show that c' is a prefix code, we consider two cases.

If $k(x_1) = k(x_2)$, then $c'(x_1)$ and $c'(x_2)$ have the same length, but $c'(x_1) \neq c'(x_2)$ because x_1 and x_2 have different binary representations. Thus $c'(x_1)$ and $c'(x_2)$ cannot be prefixes of each other. If $k(x_1) \neq k(x_2)$, suppose $x_1 < x_2$, then $c'(x_1)$ is no longer than $c'(x_2)$ and the first $l(c(k(x_1)))$ bits of $c'(x_1)$ cannot be a prefix of the first $l(c(k(x_2)))$ bits of $c'(x_2)$ due to Part (a) (as these are codewords of c). Thus $c'(x_1)$ cannot be a prefix of $c'(x_2)$.

We still decode codeword by codeword. For each codeword, the decoder first decodes $k(x)$ using the method in Part (a), then the decoder converts the $k(x)$ bits following $c(k(x))$ to integer x.

(c) If the pmf of X is decreasing, we have that $p(x) \le 1/x \quad \forall x \in \mathbb{N}$. Suppose this is not the case, i.e., for some $x \in \mathbb{N}$, $p(x) > 1/x$. Then, $\sum_{i=1}^{x} p(i) > xp(x) > 1$, which is a contradiction. Therefore,

$$H(X) = \sum_{x \in \mathbb{N}} p(x) \log \frac{1}{p(x)} \ge \sum_{x \in \mathbb{N}} p(x) \log x = \mathbb{E}[\log X].$$

(d) Note that $k(x) = \lfloor \log x \rfloor + 1 \leq \log x + 1$. Hence, $\mathbb{E}[k(X)] \leq \mathbb{E}[\log X] + 1 \leq H(X) + 1$. For any $x \in \mathbb{N}$, $l(c(x)) = 2k(x)$. Therefore, the average code length satisfies

$$\mathbb{E}[l(c(X))] = 2\mathbb{E}[k(X)] \leq 2(H(X) + 1).$$

(e) For any $x \in \mathbb{N}$, $l(c'(x)) = l(c(k(x))) + k(x) \leq 2(\log(k(x)) + 1) + k(x)$. Therefore, the average code length satisfies

$$\begin{aligned}\mathbb{E}[l(c'(x))] &\leq 2(\mathbb{E}[\log k(X)] + 1) + \mathbb{E}[k(X)] \\ &\leq 2(\log(\mathbb{E}[k(X)]) + 1) + \mathbb{E}[k(X)] \\ &\leq 2(\log(H(X) + 1) + 1) + H(X) + 1 \\ &= H(X) + 2\log(H(X) + 1) + 3\end{aligned}$$

5. **Extending to Shannon Codes**
For a general source, let

$$n_u^* = \lceil \log \frac{1}{p(u)} \rceil \qquad \forall u \in \mathcal{U}.$$

Then,

$$\sum_{u \in \mathcal{U}} 2^{-n_u^*} \leq 1.$$

We want to consider a new source $p^*(u) = 2^{-n_u^*}$. $p^*(u)$ does not sum to 1 over $\mathcal{U}$, but we claim that we can add a **finite** number of new symbols to extend the source to $\mathcal{U}^* \supseteq \mathcal{U}$ such that $p^*(u)$ is dyadic over $\mathcal{U}^*$. Prove this claim.

Hint: How can you reduce this problem to showing that certain rational numbers have a finite binary representation?

Solution: Extending to Shannon Codes.
Let's understand the probability gap we need to fill with the new symbols:

$$\Delta = 1 - \sum_{u \in \mathcal{U}} 2^{-n_u^*}.$$

We know that Δ must be rational because the sum of rational numbers must result in a rational number:

$$\frac{p_1}{q_1} + \frac{p_2}{q_2} = \frac{p_1 q_2 + p_2 q_1}{q_1 q_2}.$$

In fact, we are confident about an even stronger statement: the denominator of $\Delta = \frac{p}{q}$ is a power of 2:

$$q = \max_{u \in \mathcal{U}}(2^{n_u^*}).$$

Thus, Δ has a finite binary representation because Δ's binary representation reduces to the binary representation of an integer p (the numerator).

How exactly would the binary representation of Δ look? The number of bits following the decimal point would equal

$$\log q = \max_{u \in \mathcal{U}}(n_u^*)$$

and the least significant bits would consist of the binary representation of p.

Then, it is straightforward that we can express

$$\Delta = \sum_{i=1}^{\log q} 2^{-i} \alpha_i$$

where α_i equals the value of the i'th decimal place of the finite binary representation of Δ. Each non-0 term in the summation corresponds to the probability assigned to a dummy symbol in the extension of $\mathcal{U}$.

Finally, we have

$$\sum_{u \in \mathcal{U}^*} p^*(u) = \sum_{u \in \mathcal{U}} 2^{-n_u^*} + \sum_{i=1}^{\log q} 2^{-i} \alpha_i = 1$$

where each term in the summation corresponds to the probability of a unique symbol in $\mathcal{U}^*$; this formulation satisfies the definition of a dyadic source.

6. **Decoding LZ77**
We encoded a binary sequence using LZ77; we now want to decode the resulting bitstream. We first decode it into the triplets and obtain:

(0, 0, 1)	(0, 0, 0)	(1, 5, 1)	(8, 2, 1)
(a)	(b)	(c)	(d)

Recall that the first entry of the triplet indicates how far back in the sequence you must go to start decoding the phrase; the second entry of the triplet indicates how many elements from that point should be "copied" into your newest phrase entry; and the final entry of the tuple indicates the new element (unseen in the past sequence) that should be added.

Specify how these triplets will now be decoded to reconstruct the original source sequence.

Solution: Decoding LZ77.

(a) 1
Haven't seen anything before.

(b) 0
Haven't seen a 0 before.

(c) 000001
Have seen a "0" 1 spot ago. Continue to see the matching characters for 5 spots. (There is overlap with what has just been decoded.) Then see a 1.

(d) 101
Have seen a "10" 8 spots ago. See a total of 2 matching characters. Then see a 1.