Week 4

1 Significance of entropy and other terms

We will be dealing with fixed length source code.

1. If we are willing to tolerate some small probability of error, we can conpress the source better.

   We can use a smaller length for representing the source by ignoring symbols which have a very low probability of occurence.

2. Club multiple random variables together.

   Suppose the source is transmitting data as $(X_1, X_2, X_3) = \{a, b\}^3$. Assume that X_1, X_2, X_3 are all independent random variables.

   $$P_{X_1,X_2,X_3}(x_1, x_2, x_3) = P_{X_1}(x_1)P_{X_2}(x_2)P_{X_3}(x_3) \quad \forall\, x_1, x_2, x_3 \in \{a, b\}$$

   We know the joint distribution from the individual distribution(called marginal distribution $P_{X_1}(x_1), P_{X_2}(x_2), P_{X_3}(x_3)$).

   $$P_{X_1,X_2,X_3}(x) = p^{n_a(x)}(1-p)^{n_b(x)}$$

   where, x is (x_1, x_2, x_3), $n_a(x)$ is the number of times 'a' occurs in x.

   If we have a 'compression scheme' for one variable from

   $$C_s : \{a, b\} \to \{0, 1\}$$

   We can get a scheme for $\{a, b\}^3$ using the above as:

   $$C'_s : (x_1, x_2, x_3) \to (C_s(x_1), C_s(x_2), C_s(x_3))$$

   $$\{a, b\}^3 \to \{0, 1\}^3$$

   Length of code: 3 bits.

   This code C'_s is as good as the original code C_s.

   In the case of encoding just one source symbol, our possible code lengths were either 0 or 1. Here, we have more choices 0, 1, 2 or 3 length binary strings (vectors or tuples) can be used.

   Let,

   $$C'_s : \{a, b\}^3 \to \{0, 1\}$$

   Length is 1, while normalised length is $\frac{1}{3}$. This would be a good code if, (a, a, a) has a very high probability and the 7 other vectors have small probability.

3. Suppose we are allowed to combine multiple symbols and encode them together into some fixed length binary string, then this gives a more 'efficient' source code i.e. smaller normalised length.

   We shall impose some requirements:

   - We are encoding long source strings and can tolerate some small probability of error.
   - We have to use a fixed length source code. (Every n length source string is to be encoded into a l length binary vector/string/tuple. Fixed length meaning l doesn't change with the source string.)

   Assumption: n length random source vector is represented by $(x_1, x_2, \cdots, x_n)$, where x_i is the random variable representing the i^{th} output of the source $\in \{a, b\}$.

   $$P_{X_i} = P_X \qquad (P_{X_i}(a) = P_X(a),\ P_{X_i}(b) = P_X(b))$$

   $X_i's$ have same distribution and independent. In the language of communications, $X_i : i \in 1 \cdots n$ are said to be independent and identically distributed (IID).

   Question: Suppose n is very large, how many $a's$ & $b's$ do we expect to see in the random source sequence $(x_1, \cdots, x_n)$?

   Number of $a's$: np.

   Number of $b's$: $n(1-p)$.

   Number of sequences with such distribution of $a's$ and $b's$: ${}^nC_{np} = \binom{n}{np}$

   Now, for an efficient source code, we will encode only these $\binom{n}{np}$ sequences with unique codewords. And for all other sequences we use a single codeword.