Class no 9 :- FIXED LENGTH SOURCE CODE (output of encoder is some binary tuple of a fixed length)

Key Significance of entropy & other terms:

Tx Source X → Source code encoder → codewords (rep of the source) → Rx Decoder → Source

decoder is assumed to know the P_X

① Idea no 1:

If we are willing to live with / tolerate some small probability of decoding error, we can compress the source better, ⇒ we can have smaller length for representing the source.

by 'ignoring' source symbols which have very low probability of occurrence.

ignoring means, not encoding (or) means encoding

② Idea no 2: 'Club' multiple source RV instances.

Source → $(X_1, X_2, X_3) \in \{(a,a,a), \ldots, (b,b,b)\} = \{a,b\}^3$ (like $\mathbb{R}^3$)

(Assume that X_1, X_2, X_3 are all independent RVs

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

We know joint distribution from the individual distributions

marginal distributions $(P_{X_1}(x_1), P_{X_2}(x_2), P_{X_3}(x_3))$

$$\boxed{P_{X_1,X_2,X_3}(\underset{(x_1,x_2,x_3)}{\underline{x}}) = p^{n_a(\underline{x})}(1-p)^{n_b(\underline{x})}}$$

$P_X(a) = p$

$P_X(b) = 1-p$.

$n_a(\underline{x})$ = no of times 'a' occurs in $\underline{x}$

$n_b(\underline{x})$ = no of times 'b' occurs in $\underline{x}$.

Suppose we have some 'compression scheme' (a mapping)

from $C_S: \{a,b\} \to \{0,1\}$. $\left(C_S(x) \in \{0,1\} \right)$

Can we use this to get a scheme for $\{a,b\}^3$?

Yes: $C_S' : (x_1, x_2, x_3) \to \left(C_S(x_1), C_S(x_2), C_S(x_3) \right)$.

$: \{a,b\}^3 \longrightarrow \{0,1\}^3$ (Fixed length code)

$C_S'(a, b, a) \longrightarrow (0, 1, 0)$ (suppose $C_S(a) = 0$, $C_S(b) = 1$)

Length of this code: 3 bits. to 3 source symbols

↓

This code C_S' is as good as the original code C_S.

→ In the case of encoding only one source symbols, our possible code lengths were either 0 or 1 (only)

→ Here we have more choices 0, 1, 2, 3 length binary strings (vectors or tuples) can be used.

$\boxed{C_S'} : \{a,b\}^3 \longrightarrow \{0,1\}$ [Length = 1, Normalized length = 1/3]

This is a good code if $(a, a, a) \Rightarrow$ very high prob

$\{7 \text{ vectors}\} \to$ totally has a small prob.

In this case we can map

$C_S'\left((a,a,a)\right) = 0$, & $C_S'(\underline{x}) = 1 \;\forall\; \underline{x} \in \{a,b\}^3 \setminus (a,a,a)$

$(\underline{x}) \longrightarrow \boxed{\text{Encoder}} \longrightarrow 0 \longrightarrow \boxed{\text{Rx decoder}} \longrightarrow (a,a,a)$

knows the code, knows P_X

Idea no 2: Suppose we are allowed to combine multiple source symbols & encode them together into some fixed length binary string, then this gives a more 'efficient' source code (smaller normalized length)

Example:

Source $X \sim P_X$; $P_X(a) = p$
$P_X(b) = 1-p$

Remember: (1) We are allowing for encoding long source strings & we can tolerate some small prob of error.

(2) We have to a fixed length source code (every 'n' length source string is to be encoded into a 'l' length binary vector / string / tuple.
Fixed length means l doesn't change with the source string.)

Assumption: 'n'-length Random Source vector is represented by

$(X_1, \ldots, X_n)$, where

X_i is the RV representing i^{th} output of source. $\in \{a, b\}$

[X_is have same distribution] $\leftarrow$ $P_{X_i} = P_X$ ($P_{X_i}(a) = P_X(a)$, $P_{X_i}(b) = P_X(b)$)

[X_is are independent]

In the language of communications, $X_i : i \in 1 \ldots n$ are said to be "independent and identically distributed" [i.i.d]

Question:

Suppose n is 'very large', how many 'a's and 'b's do we expect to 'see' in the random source sequence $(X_1, \ldots, X_n)$?

No of 'a's $\approx np$

" " "b's $\approx n(1-p)$

No of such sequences with such a distribution of 'a's & 'b's

Set of Typical sequences $\approx \binom{n}{np}$ [Notice that this is only a subset of the 2^n sequences]

Idea of the efficient source code we want to use is that we will encode only these $\binom{n}{np}$ sequences with unique codewords

→ {for all other sequences we will use a single codeword}

Class no 10:

Atypical sequences: not typical ⇒ no of 'a's is much different from np.
& no of b's is very diff from $n(1-p)$

Intuitively easy to say that

w.h.p (with high probability), any n-length sequence obtained as output of the source (running n times) is going to be a typical sequence.

& P(we will get a atypical sequence as output) is very small.

Fixed length (n) - sequences

Source code: (Fixed length source code) → ["Block → Block source coding" (book terminology)]

Assign to each typical sequence a unique codeword of some length. L

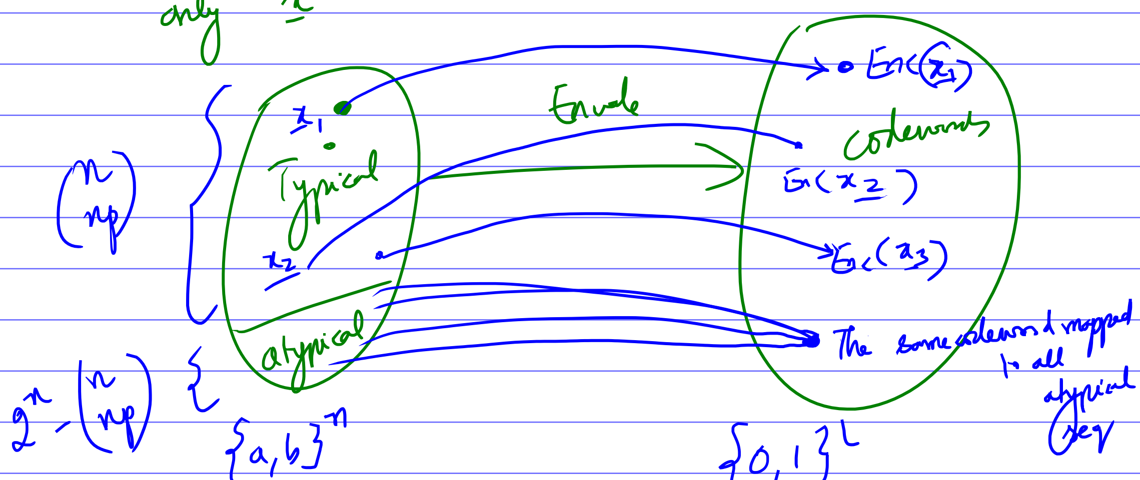
$\boxed{\text{Typical sequences} \rightarrow \text{Codewords is a one-one map.}}$

Assign to each atypical sequence, assign some "same" codeword of length L different from those assigned to typical sequences

$\underline{x} \in$ Source output. (n length seq)

$Enc(\underline{x}) \rightarrow$ codeword associated with $\underline{x}$

If $\underline{x} \in$ Typical seq, then decoder/Rx will be able to identify $\underline{x}$ without errors as $Enc(\underline{x})$ will be uniquely associated with only $\underline{x}$



For defining such a map, what should be min length of L?

L is atleast $\underbrace{\log_2 \binom{n}{np}}_{\text{typical}} + 1$ → for atypical.

Now,

$$\log_2 \binom{n}{np} = \log_2 \left(\frac{n!}{(np)!\,(n-np)!} \right)$$

$$= \log_2(n!) - \log((np)!) - \log((n(1-p))!)$$

$n! = n(n-1)(n-2)\ldots(n-(n-1))$

$\approx n^n - \left[\text{Some poly in } n \text{ of degree} < n\right.$.

(Better approx are there, we are doing rough analysis) $\rightarrow$ Stirlings approximation

rough

$$\approx n\log_2 n - np\log_2 np - n(1-p)\log_2(n(1-p)) + \underline{o(n)}$$

$\hookrightarrow$ much smaller than other terms in the summation.

$$= \underline{n\log n} - np\log p - \underline{np\log n} - \underline{n(1-p)\log n} - n(1-p)\log(1-p) - \underline{\text{small term}}$$

$$= n\left[\underline{\underline{p\log_2\frac{1}{p} + (1-p)\log_2\frac{1}{1-p}}} - \frac{\text{small terms}}{n}\right]$$

remains constant as n grows

$\hookrightarrow$ goes to 0 as $n \to \infty$.

$\approx n\,H(X)$, where X is the source RV.

$\Rightarrow$ It is sufficient to have length of codewords $\approx \underline{nH(X)+1}$

n length source sequences $\longrightarrow$ $nH(X)+1$ length source code.

Per source symbol, what is the length of the codeword?

$= H(X) + \frac{1}{n}$ bits $\left(\rightarrow H(X) \text{ as } n \to \infty\right)$.

$$\boxed{\text{Per source symbol, we "are using" } H(X) \text{ bits. [sufficient] Infact, we "need" } H(X) \text{ bits otherwise we will have large } P(\text{error}).}$$

Fixed length source sequences to variable length codewords

$$X \in \{A, B, C, D\} \longrightarrow \text{varying length codewords}$$

$X \sim P_X$

X RV is distributed according to the distribution P_X

eg

Source code 1:

$A \longrightarrow 0$

$B \longrightarrow 1$

$C \longrightarrow 10$

$D \longrightarrow 11$

We expect to do 'better' than previous fixed-fixed length scenario because we have the freedom here to set varying length codewords.

We will therefore demand zero probability of error

In prev fixed-fixed scenario we cannot do any compression if we wanted $P(\text{error}) = 0$.

However we have a problem :-

Suppose we use source code 1 above

Then if source generates BA $\longrightarrow$ 10

or C $\longrightarrow$ 10

same codeword

Confusion at decoder