Keeping the Noise Down

I have long held the opinion that the amount of noise that anyone can bear undisturbed stands in inverse proportion to his mental capacity and therefore be regarded as a pretty fair measure of it.

Arthur Schopenhauer (1788 – 1860)

The next stage - Channel behaviour

We have considered one aspect of Information Theory: what “happens” when a message is “prepared” for transmission and, also, the processes of transforming symbols into binary.

The mechanisms involved when a message is **received** mirror the actions involved in sending.

The next stage - Channel behaviour

We now look at what affects the "middle" of this activity:
messages are sent through a "*channel*" (wire, ether etc).

Q1. How can the **sender** be "*sure*" the message has arrived "*uncorrupted*"?

Q2. How can the **receiver** be confident that the message received is what was **intended** to be **sent**?

()

The effect of "noisy" channels

Some examples of garbled messages

1. "*Send three-and-fourpence we're going to a dance*"

Field-telephone message to British army HQ (WW1).

2. Coach-hire company:

"*How many people do you need buses for?* "

Reply:

"*aboot six tae seven*"

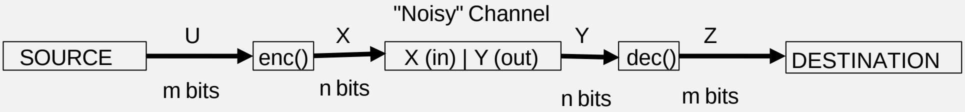
The effect of "noisy" channels

The "real" messages

"*Send reinforcements we're going to advance*"

"*about six to seven*" (6–7 **not** 67)

An abstract model of noisy channel behaviour



After encoding U with m-bits a binary sequence ($enc(U) = X$) with n-bits is passed to the "*channel*".

If "*noise*" is present what will emerge is a binary sequence (Y) which **may differ** from X.

Y is decoded to Z ($dec(Y) = Z$) and Z (m bits) is sent.

We wish to "*minimise*" possibility of errors by encoding (in n bits) messages that use m bits.

Ideally, n should be "*small*" relative to m: we don't want to "*pad out*" the message with "*too much*" "*redundancy*".

Noise as the "probability of errors"

The abstraction describes the action of the channel as a **function** mapping *encodings* (*X*) of *input* words (*U*) to *transmitted* words (*Y*) that *decode* to *final texts* (*Z*).
Given the nature of "noise" we cannot **predict with certainty** what will appear as *Y* when *X* is given as input.
(we cannot tell **exactly how long** it will take for the person braying into their mobile phone to scream "*I'm on the TRAIN*")

Binary Channels

We focus on a particular class of “noisy” channel where the stream of source symbols U are $\{0, 1\}$.

A sequence of symbols, U, is translated as $X = enc(U)$ for transmission through a channel which outputs Y.

This is used to give $Z = dec(Y)$ as the message received.

Binary Channels

Suppose we have values p ($0 \le p \le 1$) and q ($0 \le q < 0.5$): with:

U. $P[U = 0] = p$; $P[U = 1] = 1 - p$.

The input is 0 with probability p; 1 with probability $1-p$.

Y. Given an **input** bit X the channel leaves this **unchanged** with probability $1 - q$ and "*flips*" (negates) this bit with probability q.

Problems when sending m bits

With the scenario of the previous slide if we send U without making any changes ($enc(U) = U$) basic probability show us that $\sim qm$ bits will be **corrupted**.

If q is "large" (very close to 0.5) the message received (Z) is **unlikely to resemble the message sent** (U).

Thus, P_e, the probability of error, that $U \neq dec(enc(U))$ will be "*quite high*".

Avoiding the problem: building redundancy

Now consider an alternative:

For U, $X = enc(U)$ "***repeats***" (the symbols in) U, 3 times:

if $U = 0$, $enc(U) = 000$,

if $U = 011$, $enc(U) = 000111111$.

Z is $dec(Y)$: its ith bit is the *majority* of $y_{3i}y_{3i+1}y_{3i+2}$,

if $X = enc(U) = 000111111$ and $Y = 010101110$

$Z = dec(Y) = 011$.

Error probabilities and "repetition coding"

In the approach described we reduce the chance of text being corrupted by *repeating its content* and applying a "*majority*" vote scheme to the result.

Before (making no change to U) with "one-bit-at-a-time" we have $P_e = q$.

When we repeat each bit 3 times,

$$P_e = P[\text{At least 2 of the 3 bits are corrupted}] = 3(1-q)q^2 + q^3 < q$$

Error probabilities and "repetition coding"

We cut error probability **BUT** at the cost of *reducing the bit rate*

BEFORE: Bit rate with *no change* to *U*: $n/n = 1$; $P_e = q$;

AFTER: Bit rate with *3-repeats*: $n/3n$; $P_e < q$

Transmission depends on the encode-decode convention.

Using *e* for *enc*() and *d* for *dec*() : (e, d) is called a **scheme**.

The "*transmission speed*" is called the **scheme rate**, denoted R, and is the *number of bits* sent each time.

Messages with m bits are "*padded*" to n bits: the **rate** is m/n

()

Information Theory – Summary

The material is a very introductory discussion looking at:

a) "encoding" symbols in binary (*Source Coding*)

b) Noise and reducing its effect (*Channel Coding*).

Schemes in (a) try to **reduce** the number of bits sent.

Schemes in (b) try to minimize the number of bits **added**.

Coding Schemes are an important area of CS.

Some of these (Huffman Codes, Liv-Zempel Codes) may be met later in the programme.