GRESHAM COLLEGE

23 OCTOBER 2018

IT FROM BIT: THE SCIENCE OF INFORMATION

PROFESSOR RICHARD HARVEY FBCS

The word *information* is bandied about so frequently that it is hard to know what people mean by it anymore[i]. Often it is used as a rather grander alternative for stuff or data, but sometimes in government and intelligence circles as a euphemism for stuff to do with computers as in the term *information warfare.* These lectures are sponsored by the Worshipful Company of Information Technologists whose mission is to use IT skills to make a difference. Specifically, they would wish to be "Improving the understanding of IT and its capabilities to the wider public." That is obviously a very commendable goal, the issue, I suspect, is that the Worshipful Company of Information technologists have a completely different level of understanding of what information is compared to the wider public. When they talk about information, they mean something quite different from the rest of us. This lecture aims to sort that out – what exactly is information and what technology is needed to process it? This lecture will also give pointers to the other lectures so, if I leave some questions unanswered then good, like an episode of Paul Temple, or a Dickens chapter, I want you to tune into the next instalment.

The uncontested father of Information Theory is Claude Ellwood Shannon. In my slides there is a previously unpublished photograph of him in 1982 striding purposefully across the UEA campus on his way to receive an honorary degree. Besides being an engineer, Shannon was fascinated with games, tricks and entertainments including juggling machines, maze-solving mouse and, my favourite, something now called the "ultimate machine" – a machine that when turned-on turns itself off again. Replicas of this machine can be yours for anywhere between £15 and $30.

However, let me avoid the classic historian's mistake of talking about a person rather than his ideas. Shannon realised that a lot of digital systems, remember we are talking about 1948, might be represented as source of digital data, a transmitter, a channel that potentially corrupted the signal with noise, a receiver and a destination. Shannon was using a conceptualisation that we now call the "Black box" approach[1][ii] but in order to generate any theoretical ideas he realised that he had to be more precise about what information was. His paper "A mathematical theory of communication"[iii] was published 70 years ago and is still relevant, fresh and completely staggering in its breadth since it laid the foundations for all digital communication. Shannon defined information, and we will talk about that, he discussed communication in noise, codes that when corrupted by noise can correct themselves (error-correcting codes) and much much more. Shannon may indeed be the most famous person you have never heard of.

All modern computers are based on the binary system. They conduct all their arithmetic and data processing using only zeros and ones. Thus, if we took a chunk of text, say the first 500 characters of a Wikipedia entry on "Gresham's Law" we face the problem of how to transmit that digitally using only zeros and ones. Of course, there are many *codes* that might be used for this purpose. Historically one of the most common was a code known as ASCII[2] (pronounced ass key). In this code the capital letters run from the binary for 65 to 90, the lowercase

[1] There is some doubt as to the origin of the phrase "Black box." I think it highly likely that it refers to Black's boxes. Harrold Black was a telephone engineer who was working on amplifiers for the telephone system. They were tricky to design since they often went unstable. In combination with Harry Nyquist, Black realised that using negative feedback would stabilise the amplifiers hence the superior Black's boxes. This term certainly pre-dates the discovery of the "black box" approach so, as with so many technical things, I imagine Black's boxes got conflated with the black box approach and the name stuck.

[2] American Standard Code for Information Interchange

letters from 97 to 122, space is 32 and punctuation and other special characters fill the other positions. ASCII uses eight binary digits (or bits) so it can represent $2^8 = 256$ characters. There are other codes, and nowadays we are less Amerocentric and admit that there might be languages, other than American, so we use longer codes to represent characters that are common in other languages. But let's stick with ASCII for the time being. In which case, the first 500 characters of our Wikipedia entry would each be coded with 8-bits so the total message size would be 4000 bits[3] with 8 bits per character.

However, in English, not all letters are equally likely. 'e' is the most common letter appearing around 12.5% of the time[4] followed by "t" and the vowels "o" and "i". Indeed, the rank order of characters by frequency was immortalised in the order of the characters of the linotype keyboard: ETOIN SHRDLU[5]. So, surely, we should use shorter codes for the more common characters? Yes. That works very well. There is some care needed, because no code can be a prefix of another code -- we now have variable-length codes and if one code is the same as the start of another, we will not be able to decode the code at the receiver. There are many such codes and the classic one of undergraduate and/or A-level curricula is the Huffman code. David A Huffman was another brilliant man you have never heard of, but maybe that is for another lecture. To compute the average size is quite easy – we multiple each code length by its probability and add up. In this case the average code length turns out to be 4.2 bits per character[6] so our 4000-bit message has shrunk to 2100 bits. That's a compression ratio of 52.2%. Not bad. But could we do better? Are there better codes than Huffman codes? Is there a lower limit?

Well that question was examined by Shannon who defined information to be $I = \log_2 (1/p)$. The derivation arises from thinking about how one might represent say N different possibilities using binary digits. Well if $N = 2^I$ where I is the number of bits, then $\log_2 N = I \log_2 2 = I$. But if $N = 1/p$ then $I = \log_2(1/p)$. So, I can be thought of as the number of bits required to represent p. When p is one, something is completely predictable, and the information is zero. When p is zero something is completely unpredictable and there is infinite information. Hence surprise implies information implies bits. So the science of information is the science of surprise.

From that definition of information comes entropy. Entropy is the average information in the code. For this example, it turns out to be 4.176 bits – slightly lower than the Huffman size. In undergraduate courses, people often say that Huffman is an optimal code, by which is meant that it hits the Shannon limit. It is not. Huffman is only optimal when the symbol probabilities are exact powers of ½. In practice the Huffman code is usually close to optimal though. However modern codecs[7] often use arithmetic codes in which the message is transmitted as a binary floating-point number since arithmetic codecs can represent symbols with any probability.

Just as we can have a conditional probability which is the probability that something happens given something else, we can have conditional information. Actually, almost all information is conditional on something. When we computed the information of English characters we were assuming, or taken as given, that the probabilities of those characters were typical for English text and not from the novel *Gadsby* for example. I can illustrate that conditionality with a few visual metaphors in which the viewer's interpretation changes completely once they have available to them some "side information". It's important to remember that *given* aspect of information. Without a common understanding of what is given no communication can take place.

When it comes to practical compression, the Shannon limit does not tell us how to design a practical compressor, but it does tell us, in principle, if a particular form of compressor is capable of hitting the Shannon limit. That's useful but Shannon's principal contribution, as far this lecture goes, was the realisation that information was related to probability and that the capacity of channels was related to how many improbable events we were required to transmit down them.

---

[3] Some authors would represent 4000 as 4Kbits. However, 4kbits is 4×1024 bits (1024 being 2^{10}) so there is potential for confusion.
[4] *Gadsby* by Ernest Vincent Wright is an entertaining counter-example
[5] Shrdlu was also the name of a famous early natural language processing system.
[6] Experts might be puzzled why we do not have 1.5 bits per character which is folklore for text compression. The answer is that here we are considering only the occurrence of single characters (unigrams) whereas real coders consider pairs, triples, quads of characters (n-grams) and coding these gives better compression. See the later lecture on Text for more on this.
[7] A coder and decoder combination is usually called a codec.

In practice *lossless* compression is commonly used for text but less so for other modes. The reason is that the amount of information in video, for example, is simply staggering. This lecture is being livestreamed. We are using a 1080p camera, with three channels which means there are 1080 × 1920 × 3 8-bit digits per frame (49.8 Mbits per frame) and there are at least 25 frames per second. That's 1.2Gbits per second. Youtube livestreaming at a decent quality level probably needs around 5 Mbits per second. So, video needs to be compressed to $1/250^{th}$ of its original size. That is only achievable using *lossy* compression. So, for video, audio and images we discard information that we hope the receiver will not notice. What to throw away is a fascinating question and it has to be coupled with a deep understanding of what task we are trying to complete with the data. I want to look at that in more depth in the next three lectures which cover speech, vision and deep learning. Needless to say, a huge amount of data is discarded and while some of the discard can be accounted for source-coding (that is devising codes that hit the Shannon limit) much of it is data that we believe the human brain can be fooled into reconstructing.

You might notice that, even for lossless compression we seem to use different compressors for different signals. There's a reason for that. There is no universal compressor. Some people would invoke a marvellously named theorem known as the "No Free Lunch Theorem" to show that, but perhaps the easiest observation is to imagine we have a compressor into which we feed in a text file. We then take the output and feed that back into the input. Does it get any smaller? Well if it did then it was terrible compressor. So, it might get smaller, but eventually there will a point where the output is longer than, or of equal length to, the input. At this point, we have found a string or sequence for which that compressor does not work. Since that argument holds for all compressors, therefore, there is no universal compressor.

So far, I have talking about insights that all derived from Shannon. Shannon realised that one should think of messages as probabilities and that information was a formalisation of probability that allowed us to measure the size of information in bits. If I gave you a string of digits, then…

14159265358979323846264338327950288419716939937510582097494459230781640628620899862803482534211706 7

You would follow the Shannon recipe and measure the probability of the digits. You might, for such a short string, whinge that it was too short and insist that I tell you what the probability of the digits was. They are equiprobable. You would then apply the Shannon recipe and, hey presto, compute the information as $-\log_2(1/10) = 3.3$ bits per digit. Hence 100 digits would require at least 330 bits to transmit, and probably more since this is the Shannon Information which is the lower limit.

But these are the first 100 digits of π. Are we really saying that to transmit π requires an infinite amount of time and bandwidth? That seems mad – especially as I have just transmitted it to you via the simple expedient of saying "pi" or using the symbol π. One escape from this unsatisfactory situation is to use conditionality – we all have shared knowledge of what π is, and so, somehow, we just need to add some more symbols to our dataset. Well it's a bit ugly. How many symbols? What is common knowledge for me, is not for you. And why π but not e?

The solution comes from another famous man you may not have heard of, Andrey Kolmogorov. Actually, I should point out that Shannon is rather unusual in that his invention of information theory seems to be quite widely accepted. For most theoretical inventions there is often a bit of battle as to who named it. And for quite a bit of the twentieth century Russia and the rest of the world were not on speaking terms, so it was quite common to discover that something that we thought was invented by Professor X was actually invented by Professor Y from the former Soviet Union[8]. Anyway, in this case Kolmogorov got the laurels pipping Ray Solomonoff and Greg Chaitin who also have their supporters.

Kolmogorov information is a rather beautiful idea. The information in an object is the size of the shortest program that creates that object. The shortest computer program I could find to reproduce pi is 160 characters. So, I can be confident that pi contains less than 160 characters of information.

[8] People can get rather hot under the collar about this. The Tom Lehrer song 'Lobachevsky" which, for humorous effect accuses the pre-Soviet mathematician Nikolai Lobachevsky of plagiarism, is not always greeted with smiles.

One important caveat is that the Kolmogorov idea does not tell you how to compute that program. Indeed, there is some work that shows you may not be ever able to compute the shortest program – even if you are a Google and have thousands and thousands of code-monkeys[9] at your disposal. The conditionality of Kolmogorov complexity is important clearly one has to be quite clear about what the program is allowed to know. Despite these rather restrictive thoughts, Paul Vitanyi realised that Kolmogorov information might be used to measure the information distance between things. He realised that quantities such as K(x|y) and K(y|x) could be interpreted to be mean, how large is the computer that computes x given y. So if x was identical to y then it is a short program…`print y`. If x is some function of y then the program is also simple `print f(y)`. Which leads us to the idea of an information distance between objects. Now one problem is that multiple distances can be tricky to plot a flat piece of paper. Consider two objects, obviously I can distribute them on a line. Add third object and I can measure its distances to the other two and it will form the third vertex in a triangle. A fourth object might not fit on the page though! So, to represent this we have to have a mapping algorithm which maps high dimensional space to two dimensions. I'm going to use a primitive one called "Sammon mapping". But how to compute the incomputable Kolmogorov information. We are going to use zip. It sounds outrageous I know. And there are quite a few computer scientists who do find it outrageous. But, if I create that map without worry too much about the outrage for the time being, I can separate Conan-Doyle from Shakespeare. Zip understands Shakespeare!

Now what is all the fuss about. The main fuss was created by a paper by two Italian scientists who were able to reconstruct the tree of world languages by just using zip. Several well-known text experts got quite aerated about this paper which, they pointed out was a little worse than they could have done themselves using basic algorithms developed in the 1960s. They also made some disparaging noises about non-experts dabbling in fields they did not understand. The first point is fair enough. The second point is not. However even the first objection misses my point which is that zip is a general-purpose compression algorithm, it was not designed to differentiate between Romanian and Italian. Subsequent to this we developed a system that could classify images by considering images to be strings of characters that represented objects. Again, using a general-purpose compressor, we built a machine learning system. More on classification and machine in subsequent lectures.

Kolmogorov information gives us what appears to be a rather innocuous statement which is that an object can be represented by a computer program that can create it. However, this statement leads to some rather startling conclusions not least of which is "universe is a computer program" hypothesis. You may recall a rather more mundane version of this in the books by Douglas Adams – Earth was a computer program designed to answer the question of "life the universe and everything". It's an old idea, the computer pioneer Konrad Zuse postulated that we were part of a giant digital computer. Such ideas are not very popular – no one really wants to believe that they are part of some vast digital computer simulation.

However, there is a shift in the way we are doing science. There is increasing interest in looking for the algorithms behind phenomena and artefacts. Although it is invidious, I will single-out the work of Professor Enrico Coen FRS at the John Innes Centre in Norwich[iv]. He and his collaborators are interested in how living things grow and develop (Developmental Biology). They have a developed a neat technique that allows them to mark Snapdragon leaves with spots at a particular time of development. By tracking those spots, they can mathematically model how leaves change shape and hence, they hope, link back shape to plant genetics. I'm sure you have had someone remark that you have your Father's ears or your Mother's eyes. But it's a little imprecise. Do they mean your genetic code contains the program for building your Father's ears? From scratch? Or does it contain an input to a machine that could create your Father's ears? That sounds more likely. In which case every human contains an ear-making algorithm. But, as Professor Mark Nixon has noted[v], Ears are a biometric, so everyone's ear is different (even a father and son). So how much is random, how much is algorithm and how much is input? Answering questions like these will lead to fundamental new insights into the way that life is formed and propagated.

[9] A "code-monkey" is a slang term for a particularly skilled coder and problem solver but whose skill set maybe narrower than is conventionally desirable.

Not only is information the "New Language of Science"[vi] it raises fundamental questions about the universe and the laws that govern it. John Wheeler the famous Physicist had a turn for an aphorism and formulated five RBQs or Really Big Questions: Why the quantum? How come existence? A "participatory universe"? What makes "meaning"? and It from bit? That last one is interesting. What Wheeler was saying was that every object in the universe can be described by digital string. Hence, every object has a program associated with it. So, which came first? It is the chicken-and-egg question to end all chicken-and-egg questions.

[i] The Wikipedia article on "The Philosophy of Information" gives several definitions:
https://en.wikipedia.org/wiki/Philosophy_of_information
[ii] Harold Black and the Negative Feedback Amplifier, Ronald Kline, IEEE Control Systems Magazine, pp 82—85, 4, (3), 1993.
[iii] C E Shannon, "A Mathematical Theory of Communication", Vol XXVII, July 1948, No 3, Bell System Technical Jounal.
[iv] https://www.jic.ac.uk/directory/enrico-coen/
[v] https://www.newscientist.com/article/dn7672-ear-biometrics-may-beat-face-recognition/
[vi] "Information: the new language of science" Hans Christian Beyer, Harvard University Press, 2003,