Guesswork is not a substitute for Entropy

***Dr. David Malone[1], Dr. Wayne Sullivan[2],**
[1]Hamilton Institute, NUI Maynooth, Ireland,
Tel: (01) 708 6100
E-mail: david.malone@nuim.ie
[2] Department of Mathematics, UCD, Dublin, Ireland.

Abstract: Shannon entropy is often considered as a measure of uncertainty. It is commonly believed that entropy is a good measure of how many guesses it will take to correctly guess a single value generated by a source. This belief is not well founded. We summarise some work in this area, explore how this belief may have arisen via the asymptotic equipartition property and outline a hands-on calculation for guesswork asymptotics.

1. Introduction

Shannon entropy, $h(p) := -\sum p_i \lg p_i$, is often considered as a measure of the number of bits of uncertainty associated with a source which produces symbol i with probability p_i, where $\lg = \log_2$. This use, which began with Shannon's work on Information Theory, has become widespread in cryptology where it is often used outside its original context. For example, suppose the symbol i is a key for some cypher and is chosen with distribution p_i. Key guessing attacks are discussed in [10]:

We can measure how bad a key distribution is by calculating its entropy. This number E is the number of real bits of information of the key: a cryptanalyst will typically happen across the key within 2^E guesses. E is defined as the sum of $-\sum p_K \log_2 p_K$, where p_K is the probability of key K.

Similar inferences are made in Section 17.14 of [9] while discussing Biases and Correlations of random sequence generators. The quality of the random data harvested by the Yarrow pseudo-random number generator is also referred to as entropy [4]. The Entropy Gathering Daemon [11], a substitute for the Unix `/dev/random` device, speaks for itself in this respect.

There are many possible criteria for measuring 'guessability'. The one we consider here is the expected number of guesses required to get the correct answer. Various strategies can be used for guessing. Commonly know are *brute force attacks*, where all symbols are guessed in no particular order, and *dictionary attacks,* where the symbols deemed more probable are guessed first. Well known software packages, such as Crack [7], use a dictionary attack.

The guessing strategy we consider is the optimal dictionary attack, where symbols are guessed in decreasing order of probability. If the symbols produced by the source are relabeled so that p_1 is the most likely and the sequence p_i is non-increasing then the expected number of guesses is $G(p) = \sum i p_i$.

In [8] this is referred to as the guesswork. On average it takes (n + 1)/2 guesses to correctly guess from n equally likely possibilities. Thus, for comparison with entropy we define $H(p) := (2^{h(p)} + 1)/2$.

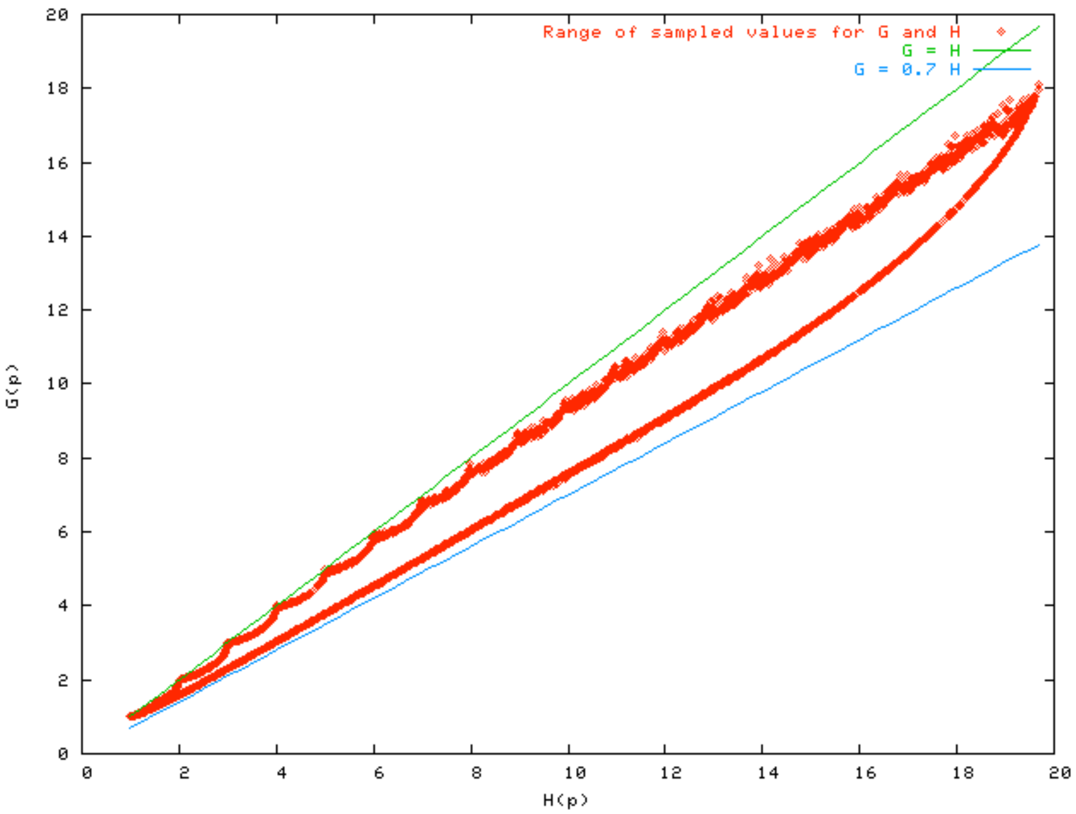

Figure 1: Samples of G(p) and H(p) for alphabets of ≤ 20 symbols.

The popular notion, *entropy ≈ number of bits of uncertainty,* suggests that we look for some sort of equivalence between G(p) and H(p). Casual numerical experiments suggest that $0.7H(p) \leq G(p) \leq H(p)$, see Figure 1. Here, a million random distributions were generated for sets of 2 to 20 symbols and the largest and smallest values of G seen for a particular H value were recorded.

2. Bounds on G and H

In [6] it is shown that a lower bound for G(p)/H(p) is 2/e. This can be derived by showing that a geometric sequence for p_i produces an extrema of H(p) while keeping G(p) fixed. The value 2/e is obtained for an infinite geometric sequence as the ratio goes to 1.

The upper bound, $G(p) \leq H(p)$, suggested by the numerical experiment in Figure 1 is shown to be incorrect in [6]. By taking a sequence where $p_1 = 1 - b/n$ and $p_2, \ldots, p_n = b/(n^2\text{-}n)$ and letting $n \to \infty$, we get sequences with $G(p) = 1 + b/2$ but H(p) tending to 1.

So H(p) is within a few bits of being a lower bound on the expected number of guesses, but may be an arbitrarily large underestimate. This is fortunate for those designing cryptosystems where entropy is used as a measure of guessability. In [3] Rényi entropy is used to give two-sided bounds on the expected number of guesses.

Note that it is possible to construct guessing problems that are related to Shannon entropy. Instead of guessing one symbol at a time, consider the problem where we may guess a set of symbols and we are told if the correct symbol is in our set. This problem is clearly easier than the simple guesswork problem. However this problem is also the same as putting the symbols into a binary tree of minimum average depth, which is the coding problem and so requires about h(p) guesses (rather than H(p) guesses).

3. Other Measures of Guessability

The example in Section 2, which dispels the possibility of an upper bound, raises an interesting issue. It produces distributions where the average number of guesses is arbitrarily large, but it places almost all the weight on the first symbol, so the mode of the number of guesses will be 1. This suggests that the average number of guesses may not be a good measure of guessability for cryptography. One better possibility would be to consider the moments of the guesswork, rather than just the mean.

Another alternative to G(p) as a measure of guessability is *the number of guesses required so that probability of having guessed correctly is at least α*. In [8] this is referred to as the α-work factor and denoted $wf_\alpha(p)$. The authors examine $wf_{1/2}$ and decide that again entropy does not provide a good estimate. However, they offer 1 $||p-u||$ as a more hopeful estimator, where u is the uniform distribution and $||p-q|| := \sup_i |p_i-q_i|$, is the variation distance.

4. Guesswork and Asymptotic Equipartition

How did this perceived link between entropy and guesswork arise? One suggestion in [8] is that it is a misapplication of the Asymptotic Equipartition Property (AEP).

The AEP applies to a collection of n i.i.d sources of symbols and the words they produce. Roughly speaking, the AEP says that if you take n large enough then there is a typical set of $2^{nh(p)}$ words which all have approximately the same probability $2^{-nh(p)}$, while the remaining words have only a small probability associated with them (see [2] for a precise statement).

A good estimate of the guesswork of these $2^{nh(p)}$ equiprobable typical words would be $(2^{nh(p)}+1)/2$, and setting n = 1 we get the folklore that G(p) ≈ H(p). The first problem with this argument is that the AEP deals with large n, what arises for the case n = 1 may be very different. Another difficulty is that some terms of low probability may contribute significantly because of the i in the expectation $\sum ip_i$ grows exponentially as n does.

5. Guesswork on Long Words

Given that the AEP requires large n, it makes sense to ask if there is an equivalence between guesswork and entropy in some “large n” sense.

Consider the situation where we guess an entire word of length n. Each character of the word has been chosen independently with distribution $p_1, p_2, \ldots$. We denote the guesswork for this problem as $G(p^n)$. A straightforward application of the AEP for large n is still not valid: as the probability of the atypical words becomes small, the weight associated to them in the sum for $G(p^n)$ grows exponentially.

We can also consider this in terms of *the principal of the largest term* and *typical sets*. When calculating expectations for n i.i.d. sources, we look at sums of the form:

$$\sum_{n_1 \ldots n_r} \binom{n}{n_1 \ldots n_r} p_1^{n_1} \ldots p_r^{n_r} f(p).$$

If the function f(p) is relatively small, then the most important term in this sum is the one which maximise the product of the multinomial coefficient and the probabilities. This term will have $n_k/n \approx p_k$. These points correspond to the typical set of the AEP.

When calculating guesswork, f(p) = rank(p) and the sum we consider is closer to:

$$\sum_{n_1 \ldots n_r} \binom{n}{n_1 \ldots n_r}^2 p_1^{n_1} \ldots p_r^{n_r}.$$

Here the largest terms will be those with $n_k/n \approx c\sqrt{p_k}$, where c is a normalising constant. Thus the dominant terms for the guesswork problem are different from those for the coding problem.

To give an explicit example, suppose our word is a binary string and that 0 is chosen with probability p and 1 is chosen with probability q = 1 - p. For simplicity, suppose $0 \leq p \leq 0.5$ so that $p^k q^{n-k}$ is in non-increasing order. Then

$$G(p^n) = \sum_{k=0}^{n} f(k,n) p^k q^{n-k} \binom{n}{k}$$

where

$$f(k,n) = \sum_{j=0}^{k-1} \binom{n}{j} + \frac{1}{2}\binom{n}{k}.$$

By balancing the binomial terms against the geometric terms, we can identify the largest term in the expression for $G(p^n)$ and show that it dominates as n becomes large. Formally, we can evaluate the average number of extra bits of guesswork we get per bit in the word:

$$\lim_{n\to\infty} \frac{1}{n} \lg G_n(p^n).$$

and show that we get $\lg((\sqrt{p} + \sqrt{q})^2)$ extra bits per character. This is clearly not the same as the estimate from Shannon entropy -p lg p -q lg q.

This result can be generalised. In [1], Arikan employs clever inequalities to produce estimates of the guesswork, showing that this result generalises to $\lg((\sqrt{p_1} + \sqrt{p_2} + \ldots)^2)$. Interestingly this quantity has already been studied and is known as the Rényi entropy. This result has also been generalised in [5] to give the moments of the guesswork when the words are generated using a Markov chain.

6. Conclusion

The entropy provides a lower bound but no upper bound on the expected amount of work required to guess a single output from a source. This is fortunate for cryptographers that have designed systems assuming that entropy is the same as guesswork. However, we also note that the expected amount of work may not be a good measure of the guessability of source. This is a sober reminder that one must be careful to consider what is required of random number generators used in computing.

It is interesting to note that these estimates do not seem to have been considered until relatively recently [6, 1, 5] and that they use abstractions such as Rényi entropy.

7. References

[1] E. Arikan. An inequality on guessing and its application to sequential decoding. IEEE Transactions on Information Theory, 42:99–105, Janurary 1996.

[2] Thomas M. Cover and Joy A. Thomas. Elements of Information Theory. Wiley, New York, 1991.

[3] S. S. Dragomir and S. Boztas. Two sided bounds on guessing moments. Research Report, Department of Mathematics, Royal Melbourne Institute of Technology, (8), May 1997.

[4] J. Kelsey, B. Schneier, and N. Ferguson. Yarrow-160: Notes on the design and analysis of the yarrow cryptographic pseudorandom number generator. In Sixth Annual Workshop on Selected Areas in Cryptography. Springer Verlag, August 1999.

[5] D. Malone and W. Sullivan. Guesswork and entropy. IEEE Transactions on Information Theory, 50:525-526, 2004.

[6] James L. Massey. Guessing and entropy. In Proc. IEEE Int. Symp. on Info Th., page 204, 1994.

[7] Alec Muffett. Crack: Password cracker. http://www.users.dircon.co.uk/~crypto/.

[8] John O. Pliam. The disparity between work and entropy in cryptology. http://philby.ucsd.edu/cryptolib/1998/98-24.html, February 1999.

[9] Bruce Schneier. Applied Cryptography, volume 61. Wiley, New York, second edition, 1995.

[10] Various. sci.crypt cryptography faq.sci.crypt.

[11] Brian Warner. EGD: The entropy gathering daemon. http://egd.sourceforge.net/.