Applications of Entropy

Math 4175

§3.5. Application of Entropy

In this section, we will apply the entropy results to a cryptosystem.

If Oscar intercepts a cipher text,

- How much information does he obtain for the plaintext?
- How much information does he obtain for the key?

For this purpose we need to compute $H(P|C)$ and $H(K|C)$ for a given cryptosystem.

Recall that for any two random variable X and Y,

$$\begin{aligned} H(X|Y) &= -\sum_y \sum_x p[y]p[x|y] \log_2 p[x|y] \\ &= \sum_y p[y]\Big(-\sum_x p[x|y] \log_2 p[x|y]\Big) \\ &= \sum_y p[y] H(X|y) \end{aligned}$$

§3.5. Computation of $H(P|C)$

We have,

$$H(P|C) = -\sum_{y\in\mathcal{C}} p[y] \sum_{x\in\mathcal{P}} p[x|y] \log_2 p[x|y] = \sum_{y\in\mathcal{C}} p[y] H(P|y)$$

Recall our example* (slides 4-7 of sec3.3.pdf) where $\mathcal{P} = \{a, b\}$ and $\mathcal{C} = \{1, 2, 3, 4\}$. For this example, we get

$$H(P|C) = \sum_{j=1}^{4} p[j] H(P|j)$$

$$\text{where} \quad H(P|j) = -\Big(p[a|j] \log_2 p[a|j] + p[b|j] \log_2 p[b|j]\Big)$$

In particular,

$$\begin{aligned} H(P|3) &= -\Big(p[a|3] \log_2 p[a|3] + p[b|3] \log_2 p[b|3]\Big) \\ &= -\frac{1}{\ln 2}\Big(1/4 \ln(1/4) + 3/4 \ln(3/4)\Big) \approx 0.81 \end{aligned}$$

§3.5. Computation of $H(P|C)$

Using the computations done previously we have

$$H(P|1) = 0, \quad H(P|2) \approx 0.59, \quad H(P|3) \approx 0.81, \quad H(P|4) = 0$$

Thus,

$$H(P|C) = \sum_{j=1}^{4} p[j]H(P|j) \approx \frac{1}{8}(0) + \frac{7}{16}(0.59) + \frac{1}{4}(0.81) + \frac{3}{16}(0) \approx 0.46$$

Recall that $H(P) \approx 0.81$ (slide 12 of sec3.4) and so uncertainty has been reduced by knowing ciphertext. Hence ciphertext reveals some information about the plaintext in our example.

Exercise: Compute $H(P, C)$ directly and verify property 2 and 3 in slide sec3.4 for our example.

§3.5. Computation of $H(K|C)$

Theorem 1: Let $(\mathcal{P}, \mathcal{C}, \mathcal{K}, \mathcal{E}, \mathcal{D})$ be a cryptosystem. Then

$$H(K|C) = H(K) + H(P) - H(C).$$

The conditional entropy $H(K|C)$ is called the key equivocation; it is a measure of the amount of uncertainty of the key remaining when the ciphertext is known.

Applying this result to our example, we have

$$H(K|C) = 1.5 + 0.81 - 1.85 \approx 0.46$$

Recall $H(K) = 1.5$ and $H(C) = 1.85$ (slide 12 of sec3.4). In this particular cryptosystem $H(K|C) = H(P|C)$.

In general the entropies $H(K|C)$ and $H(P|C)$ do not need to be the same; however, in any cryptosystem, we have the following theorem.

§3.5. Computation of $H(K|C)$

Theorem 2: For any cryptosystem, we have $H(K|C) \geq H(P|C)$.

Intuitively, this results says that given a cipher text, the opponent's uncertainty about the key is at least as great as his uncertainty about the plaintext.

Remark: Recall that a cryptosystem has perfect secrecy if $p[x|y] = p[x]$ for all $x \in \mathcal{P}$ and $y \in \mathcal{C}$. In that case, we have $H(P|C) = H(P)$. In other words, the uncertainty for the plaintext does not change with given cipher text.

§3.5. Spurious Keys

- Suppose that Oscar has a string of cipher text and we assume that Oscar has infinite computational resources.
- Oscar's goal is to determine the key by ciphertext-only attack.
- We also assume that Oscar knows that the plaintext is a "natural" language, such as English.
- In general, Oscar will be able to rule out certain keys.
- But many "possible" keys may remain, only one of which is the correct key.
- The remaining possible, but incorrect keys are called spurious keys.

§3.5. Spurious Keys

For example, let us suppose that Oscar obtains the cipher text string ULYHU which has been obtained by encryption using a Shift Cipher.

What key was used?

There are two "meaningful" plaintext strings.

- If $k = 3$, the plaintext is river
- If $k = 20$, the plaintext is arena

One of these keys is the correct one and another one is spurious.

Goal: Obtain a bound on the expected number of spurious keys.

§3.5. Entropy and Redundancy

For this purpose, first we have to define what we mean by the entropy per letter of a natural language L, which we denote H_L.

The quantity H_L should be a measure of the average information per letter in a "meaningful" string of plaintext.

- Note that a random string of alphabetic characters (not necessarily a meaningful word in English) with a uniform probability distribution, would have the entropy (per letter) equal to $\log_2(26) \approx 4.7$.
- As a "first order approximation" to H_L we could take $H(P)$. In the case where L is the English language, we get $H(P) \approx 4.19$ by using the probability distribution:
  $p(a) = 0.082, p(b) = 0.015, \cdots, p(z) = 0.01$.

§3.5. Entropy and Redundancy

- Successive letters in a language are not independent, and correlations among successive letters reduce the entropy. For example, in English, the letter "Q" is always followed by the letter "U". For a "second approximation" we would compute the entropy of the probability distribution of all digrams and then divide by 2.

- Let P^2 denote the random variable of digrams. In the case of the English language, a tabulation of a large number of digrams and their frequencies would produce an estimate for $H(P^2)$. $H(P^2)/2 \approx 3.56$ is an estimate obtained in this way.

- In general, let P^n denote the random variable of all n-grams of plaintext. Notice that the sample space is $\mathcal{P}^n$.

§3.5. Entropy and Redundancy

Let L be a natural language and P^n be as above. The entropy of L is defined to be the quantity

$$H_L = \lim_{n\to\infty} \frac{H(P^n)}{n}$$

and the redundancy of L is defined to be

$$R_L = 1 - \frac{H_L}{\log_2 |\mathcal{P}|}$$

- H_L measures the entropy per letter of the language L.
- A random language would have entropy $\log_2 |\mathcal{P}|$.
- The quantity R_L measures the fraction of "excess characters" which we think of as redundancy.

§3.5. Entropy and Redundancy

Remarks:

- Experiments have yielded the empirical result that $1.0 \leq H_{English} \leq 1.5$. That is, the average information in English is something like one to one-and-a-half bits per letter!

- Using $H_{English} = 1.25$ we find

$$R_{English} = 1 - \frac{1.25}{4.7} \approx .75$$

So English language is about 75% redundant and this does not mean that one can randomly remove three out of every four letters and still be able to read. This means that we should be able to compress an English text file of around 10 MB down to 2.5 MB by finding a 'Huffman encoding' of n-grams, for a large enough value of n.

§3.5. Entropy and Redundancy

Let $(\mathcal{P}, \mathcal{C}, \mathcal{K}, \mathcal{E}, \mathcal{D})$ be a cryptosystem. Given probability distributions on $\mathcal{K}$ and on $\mathcal{P}^n$, we define the induced probability distributions on $\mathcal{C}^n$, the set of n-grams of cipher text (we have already done this in the case $n = 1$). We have defined P^n to be a random variable representing an n-gram of plaintext. Similarly, define C^n to be a random variable representing an n-gram cipher text.

Given $\mathbf{y} \in \mathcal{C}^n$, define

$$K(\mathbf{y}) = \{k \in \mathcal{K} : \exists\, \mathbf{x} \in \mathcal{P}^n \text{ such that } p[\mathbf{x}] > 0 \text{ and } e_k(\mathbf{x}) = \mathbf{y}\}$$

$K(\mathbf{y})$ is the set of those keys which produce a "meaningful" decryption of $\mathbf{y}$. Then clearly $|K(\mathbf{y})| - 1$ is the number of spurious keys for given $\mathbf{y}$.

§3.5. Entropy and Redundancy

Let $\overline{s}_n$ be the average number of spurious keys (over all possible cipher text strings of length n). Then we can compute its value as below:

$$\begin{aligned}\overline{s}_n &= \sum_{\mathbf{y}\in\mathcal{C}^n} p[\mathbf{y}](|K(\mathbf{y})| - 1) \\ &= \sum_{\mathbf{y}\in\mathcal{C}^n} p[\mathbf{y}]|K(\mathbf{y})| - \sum_{\mathbf{y}\in C^n} p[\mathbf{y}] \\ &= \sum_{\mathbf{y}\in\mathcal{C}^n} p[\mathbf{y}]|K(\mathbf{y})| - 1.\end{aligned}$$

§3.5. Entropy and Redundancy

Theorem 3: Suppose $(\mathcal{P}, \mathcal{C}, \mathcal{K}, \mathcal{E}, \mathcal{D})$ is a cryptosystem where $|\mathcal{C}| = |\mathcal{P}|$ and keys are chosen equiprobably. Let R_L be the redundancy of the underlying language. Then given a string of cipher text of length n, where n is sufficiently large, the expected number of spurious keys, $\bar{s}_n$ satisfies

$$\bar{s}_n \geq \frac{|\mathcal{K}|}{|\mathcal{P}|^{nR_L}} - 1.$$

We prove it in two steps as follows:

- **Step 1:** If $|\mathcal{P}| = |\mathcal{C}|$, then $H(K) - nR_L \log_2 |\mathcal{P}| \leq H(K|C^n)$.
- **Step 2:** $H(K|C^n) \leq \log_2(\bar{s}_n + 1)$.

§3.5. Entropy and Redundancy

Proof: If both step 1 and 2 hold, then we prove the theorem as follows:

$$\begin{aligned}
\log_2(\overline{s}_n + 1) &\geq H(K) - nR_L \log_2 |\mathcal{P}| \text{ (by combining step 1 and 2)} \\
&= \log_2 |\mathcal{K}| - nR_L \log_2 |\mathcal{P}| \text{ (keys are chosen equiprobably)} \\
&= \log_2 \left(\frac{|\mathcal{K}|}{|\mathcal{P}|^{nR_L}} \right)
\end{aligned}$$

So the proof will be completed if we prove both Step 1 and 2.

§3.5. Entropy and Redundancy

Proof of Step 1: By theorem 1, we have

$$H(K|C^n) = H(K) + H(P^n) - H(C^n)$$

Now for large n, we have

$$H(P^n) \approx nH_L = n(1 - R_L)\log_2 |\mathcal{P}|$$

On the other hand, by property 1 of last section,

$$H(C^n) \leq \log_2 |\mathcal{C}|^n = n \log_2 |\mathcal{C}|$$

$$\text{Hence } H(K|C^n) \geq H(K) + n(1 - R_L)\log_2 |\mathcal{P}| - n \log_2 |\mathcal{C}|$$

Now substituting $|\mathcal{P}| = |C|$ proves Step 1.

§3.5. Entropy and Redundancy

Proof of Step 2:

$$\begin{aligned} H(K|C^n) &= \sum_{\mathbf{y}\in\mathcal{C}^n} p[\mathbf{y}]H(K|\mathbf{y}) \\ &\leq \sum_{\mathbf{y}\in\mathcal{C}^n} p[\mathbf{y}]\log_2|K(\mathbf{y})| \\ &\leq \log_2\Big(\sum_{\mathbf{y}\in\mathcal{C}^n} p[\mathbf{y}]|K(\mathbf{y})|\Big) \text{ (by Jensen's inequality)} \\ &= \log_2(\overline{s}_n+1) \end{aligned}$$

This completes the proof of the theorem.

Remark: Notice that the right quantity of the above theorem approaches to zero exponentially quickly as n increases. Also, the estimate in the theorem may not be accurate for small values of n.

§3.5. Entropy and Redundancy

Definition: The unicity distance of a cryptosystem is defined to be the value of n, denoted by n_0 at which the expected number of spurious keys becomes zero; i.e., the average amount of ciphertext required for an opponent to able to uniquely compute the key, given enough computing time.

If we set $\overline{s}_n = 0$ in the preceding theorem, and solve for n, we get an estimate for the unicity distance, namely

$$n_0 \geq \frac{\ln |\mathcal{K}|}{R_L \ln |\mathcal{P}|}.$$

§3.5. Entropy and Redundancy

Example: Consider the Substitution Cipher.

- In this cryptosystem, $|\mathcal{P}| = 26$ and $|\mathcal{K}| = 26!$.
- If we take $R_L = 0.75$, then we get an estimate for the unicity distance:

$$n_0 \geq \frac{\ln 26!}{0.75 \ln 26} \approx 61.26/(0.75 \times 3.25) \approx 25.13$$

- This suggest that given a cipher text string of length at least 26, (usually) unique decryption is possible.