Lecture Notes
Measuring information using entropies

S. Wehner

October 22, 2012

1 A problem: key exchange

Traditionally, one of the central problems in cryptography is secure message transmission, which we already encountered at the start of last week's lecture. Let us briefly revisit this task, and use it as an example to explore several basic concepts in more detail. Secure message transmission involves two honest parties, Alice and Bob, who want to protect themselves from a malicious eavesdropper Eve. Alice has a message M which she wants to convey to Bob. However, Alice and Bob want to make sure that Eve cannot read M.

Needless to say, Alice and Bob's task would be easy if Alice could simply whisper the message into Bob's ear. Unfortunately for them, however, they are usually located very far apart, exposing their communication channel to Eve. A typical modern example of secure message transmission is when you (Alice) try to convey e.g. credit card information to an online merchant (Bob). However, today as already several thousand years ago, secure message transmission has played an important role in military applications. If you are curious, lookup the history of the Enigma on wikipedia for example!

How can Alice send her message to Bob without Eve being able to read it? The purpose of a so-called *encryption algorithm* is to hide Alice's message from Bob. Such an algorithm is simple a function $Enc : \mathcal{M} \times \mathcal{K} \to \mathcal{C}$ where $\mathcal{M}$ is the set of possible messages, $\mathcal{K}$ is the set of *encryption keys* and $\mathcal{C}$ is what is called the *ciphertext*. That is, for a message $M = m$ and key $K = k$ it outputs ciphertext $c = Enc(m, k)$. Clearly, an encryption algorithm is only useful if Bob can at all recover the message m from the cipertext when he has the key k. That is, there exists some *decryption algorithm* $Dec : \mathcal{C} \times \mathcal{K} \to \mathcal{M}$ outputting $m = Dec(c, k) = Dec(Enc(m, k), k)$. In this context, the message is also often called the *plaintext*.

1.1 Shannon secrecy

Shannon first formulated what it means for an encryption scheme to be secure. In essence, his condition states that the eavesdroppper Eve should not get any additional information about the message by looking at the ciphertext than what she already knew before. Again note that Eve knows the algorithm used, as well as the scope of the problem. For example, she may know that the messages can only be "attack" or "retreat". Recall from the discussion in our previous lecture that the idea that Eve (who has ciphertext C) should not gain any extra information means that

$$P_M(m) = P_{M|C}(m|c) \ , \tag{1}$$

for all messages m and ciphertexts c. This is indeed what is commonly known as *Shannon secrecy*. Before formally stating this notion, let us first consider what $P_{M|C}(m|c)$ actually refers to. How can we write down this distribution? First of all, let us consider the reverse $P_{C|M}(c|m)$. Note that Enc only outputs one ciphertext for each message m and key k. When we fix the message, the distribution over possible ciphertexts thus only depends on the distribution over possible keys k[1]. Thus

$$P_{C|M}(c|m) = \sum_{\substack{k\in\mathcal{K} \\ Enc(k,m)=c}} P_K(k) \ . \tag{2}$$

We can now write

$$P_C(c) = \sum_{m\in\mathcal{M}} P_M(m)P_{C|M}(c|m) \ . \tag{3}$$

Applying Bayes' rule we can now express the desired distribution as (see examples in the previous lecture notes!)

$$P_{M|C}(m|c) = \frac{P_{MC}(m,c)}{P_C(c)} = \frac{P_M(m)P_{C|M}(c|m)}{P_C(c)} \tag{4}$$

Understanding (1) we can now state Shannon secrecy as

Definition 1.1. *An encryption scheme satisfies* Shannon secrecy *if and only if for all distributions* $\vec{\rho}_M$ *over messages* $\mathcal{M}$

$$\vec{\rho}_{MC} = \vec{\rho}_M \otimes \vec{\rho}_C \ . \tag{5}$$

You should convince yourself that this does indeed capture (1).

1.2 Perfect secrecy

Another way to capture the idea of secure message transmission is known as *perfect secrecy*. Intuitively, it states that given the ciphertext C, all possible messages are equally likely to have lead to said ciphertext. More formally, we can use (2) to say that

Definition 1.2. *An encryption scheme satisfies* perfect secrecy *if for all pairs of messages* $m_1, m_2 \in \mathcal{M}$ *and all ciphertexts* $c \in \mathcal{C}$ $P_{C|M}(c|m_1) = P_{C|M}(c|m_2)$

$$\vec{\rho}_{C|M=m_1} = \vec{\rho}_{C|M=m_2} \ , \tag{6}$$

where the probability taken over the possible keys $\mathcal{K}$ *(see* (2)*).*

Think about this definition carefully. In particular, can we say that for all keys $k \in \mathcal{K}$ it implies that $Enc(k, m_1) = Enc(k, m_2)$? Can you write down the definition in vector notation as above?

Which of these definitions makes more sense? It turns out that they are completely equivalent! Let's convince ourselves that this is indeed true.

[1]Note that an encryption scheme could involve internal randomness. Since, however, this changes nothing in any of our arguments below we will ignore this fact. If you want, you can redo everything below making such randomness explicit which is straightforward - you can think of it as a larger "key".

Lemma 1.3. *An encryption scheme satisfies Shannon secrecy if and only if it satisfies perfect secrecy.*

Proof. Let's first show that if a scheme satisfies Shannon secrecy then it satisfies perfect secrecy. Note that using Bayes rule, the condition of perfect secrecy $P_{C|M}(c|m_1) = P_{C|M}(c|m_2)$ is equivalent to

$$\frac{P_{M|C}(m_1|c)}{P_M(m_1)} = \frac{P_{M|C}(m_2|c)}{P_M(m_2)} \ . \tag{7}$$

Hence if $P_{M|C}(m|c) = P_M(m)$ for all m (Shannon secrecy) then also perfect secrecy holds.

Conversely, let us now show that if a scheme satisfies perfect secrecy, then it satisfies Shannon secrecy. Note that perfect secrecy implies that for all $\hat{m} \in \mathcal{M}$

$$P_C(c) = \sum_{m \in \mathcal{M}} P_M(m) P_{C|M}(c|m) = P_{C|M}(c|\hat{m}) \tag{8}$$

Hence by Bayes' rule we have for all $\hat{m}$

$$P_{M|C}(\hat{m}|c) = \frac{P_M(\hat{m}) P_{C|M}(c|\hat{m})}{P_C(c)} = P_M(\hat{m}) \ , \tag{9}$$

which is Shannon secrecy. □

To get more intuition about what this definition really means, let us consider a small game between Alice and the eavesdropper Eve.

- Eve may choose two possible message m_1 and $m_2 \in \mathcal{M}$ and request an encryption from Alice.
- Alice picks a random $r \in \{0, 1\}$ and sends Eve $Enc(k, m_r) = c$ for a randomly chosen k.
- Eve outputs a guess g for r.

We say that Eve wins the game if $g = r$ - that is, her guess was correct. What is the probability that she succeeds? Note that if $r = 0$, then Eve receives a c chosen according to the distribution $P_{C|r=0}(c) = \sum_{k,c=Enc(k,m_0)} P_K(k) = P_{C|M=m_0}$ otherwise c is chosen according to $P_{C|r=1}(c) = \sum_{k,c=Enc(k,m_1)} P_K(k) = P_{C|M=m_1}$. I.e., Eve needs to determine which distribution was used in order to guess r. But this is exactly what you considered in the example of dice in the homework! As you have shown, her probability is given by

$$p_{\text{win}} = \frac{1}{2} + \frac{\Delta(\vec{\rho}_{C|r=0}, \vec{\rho}_{C|r=1})}{2} \ . \tag{10}$$

What can you say about $\Delta(\vec{\rho}_{C|r=0}, \vec{\rho}_{C|r=1})$ for a perfectly secure encryption scheme? How about if Eve got to choose three messages instead of 2?

We thus see that perfect security is an extremely robust notion: Even if Eve gets to choose the messages m_0 and m_1 herself she can no longer recover which one it was from the ciphertext c.

1.3 One time pad

Let us know establish one crucial fact about encryption algorithms. To this end, let us first consider a very simple one known as the one time pad, or Vernam cipher. Let's suppose we want to encrypt messages which are n-bit long strings, i.e., $m = (m_1, \ldots, m_n) \in \mathcal{M} = \{0,1\}^n$. Also, assume that Alice and Bob share an n-bit key $k = (k_1, \ldots, k_n) \in \mathcal{K} = \{0,1\}^n$ chosen uniformly at random $P_K(k) = 1/2^n$. To encrypt a message, Alice now simply XORs each message bit with a key bit. I.e.

$$Enc(m,k) = m \oplus k = (m_1 \oplus k_1, \ldots, m_n \oplus k_n) = c \ . \tag{11}$$

To decrypt, Bob simply XORs the message with the key again to get

$$Dec(m,k) = c \oplus k = m \oplus k \oplus k = m \ . \tag{12}$$

In your homework, you will prove that the one time pad obeys perfect secrecy! This is all very well, but you may notice that Alice and Bob need a very large key. In fact, it is just as long as the message! A natural question is thus: can we find any encryption scheme in which Alice and Bob use a shorter key and yet achieve perfect security?

It turns out that we really do need such a large key. Can you make a guess why? Let us investigate. For simplicity, let us consider the usual scenario where the messages are n bit strings ($\mathcal{M} = \{0,1\}^n$) and the keys are also bit strings $\mathcal{K} = \{0,1\}^\ell$.

Lemma 1.4. *For any perfectly secure encryption scheme $\ell \geq n$, i.e., the key needs to be as long as the message.*

Proof. Let's suppose by contradiction that $\ell = n - 1 < n$ and the scheme was perfectly secure. Fix a message m_1 and consider the set S of ciphertexts that can occur for some key k, i.e. $S = \{c \in \mathcal{C} \mid \exists k \in \{0,1\}^{n-1}\ Enc(k, m_1) = c\}$. Note that $|S| \leq 2^{n-1}$ since there are only 2^{n-1} possible keys.

Now, note that Enc is one-to-one, i.e., for each key k every message is mapped to a differnet ciphertext as otherwise there would be no way to decrypt the message again correctly. Since there are 2^n possible messages, there are 2^n possible ciphertexts for each key k. However, this means that for every k there must be some message m_2 such that $Enc(k, m_2) \notin S$. Let us fix such a message m_2 and key k. It follows that $\vec{\rho}_{C|M=m_2} \neq \vec{\rho}_{C|M=m_1}$ quite simply because latter distribution is over a smaller set. □

Note that this also means that we cannot ever reuse our key again. Given what we discussed in the previous lecture you may be wondering - what if we relax the problem? That is, what if we only demand that $\vec{\rho}_{C|M=m} \approx_\varepsilon \vec{\rho}_{C|M=m'}$ for some small error ε? Maybe we could get away with a much shorter key? In the homework, you will show that this is indeed not the case! In other words, even for an error ε we still need essentially n bits of key.

1.4 Key exchange

Given that key is such a scarce resource, how can Alice and Bob obtain one? Unfortunately, it turns out that this is impossible to achieve from scratch using only classical communication, even if we do for now assume that Alice and Bob have access to an authenticated channel. I.e., Alice is always sure she's talking to Bob and vice versa (Can you imagine why authentication might be a problem?). It is possible to use quantum communication to create obtain a key, which is known as

quantum key distribution. Yet, even classically it is possible for Alice and Bob to generate a key if they have access to some other resources. In some of the lectures to come, we will investigate such resources. Incidently, one can understand quantum protocols as building up such resources.

However, let us first consider what we mean by a key distribution protocol. You have already explored this in your homework, so let us briefly recap. A protocol is deemed a secure key distribution scheme if at the end of the day Alice holds some key K, Bob holds the same key K, and Eve knows (almost) nothing. In other words,

$$\vec{\rho}_{KE} \approx_\varepsilon \frac{\mathbb{I}}{|K|} \otimes \vec{\rho}_E \ . \tag{13}$$

Any protocol that achieves the condition above is called a ε-secure key distribution protocol. In the homework, you investigated a toy magical device that could be used to distribute a key. Let us know consider a slightly more complicated device, that Alice and Bob can use many times in a row. In each round i, Alice can give an input bit x_i to the device, which will output x_i to Bob (but noone else). However, with some probability p it will broadcast the bit x_i to the world, including any eavesdropper Eve. Can Alice and Bob still use this device to generate a key?

Let's consider the distribution $\vec{\rho}_{X_iE_i}$ for just on round. Note that $E_i \in \{0, 1, \perp\}$ where $\perp$ means that Eve did not get the bit in this round. We have

$$P_{X_iE_i}(x_i, x_i) = pP_{X_i}(x_i) \ , \tag{14}$$

$$P_{X_iE_i}(x_i, \perp) = 1 - p \ . \tag{15}$$

Furthermore, note that the box acts independently in each round. In particular, if Alice chooses her bits uniformly and independently we have for $x = (x_1, \ldots, x_n)$ that

$$P_{XE}(x, x) = \frac{1}{2^n} p^n \ . \tag{16}$$

Can you write down the probability that Eve's output coincides with the string in ℓ positions? It's clear that simply choosing $K = X$ certainly does not yield a key in this case. Could Alice and Bob still use the device to get a key? We will return to this question in the lecture dedicated to key distribution, but to do this, we need to understand a bit more about Eve's information.

Clearly, if $p = 1$ Eve gets everything and there is no way for Alice and Bob to make a key. However, what if $p < 1$? To this end, let us now consider how we might quantify Eve's information about X further.

2 Shannon entropy

Let's for the moment forget the fact that Eve holds any side information, but let's consider the distribution on X alone. For a key K, we would want X to be uniform. One way of measuring the "randomness" of X that goes beyond the distance from uniform is using entropies. From your undergraduate studies, you may indeed be familiar with what is known as the *Shannon entropy*. For a distribution $\vec{\rho}_X$ it is defined as

$$H(X) := H(\vec{\rho}_X) = -\sum_{x \in \mathcal{X}} P_X(x) \log_2 P_X(x) \ . \tag{17}$$

Throughout we will only consider the binary logarithm, and hence omit the subscript $\log = \log_2$ from now on. All the way into the '90s the Shannon entropy was commonly used in cryptography, and indeed still features in many textbooks. However, it finds little use in modern textbooks. Let us consider a simple example illustrating why this is the case. First of all, let's consider X being uniformly distributed over $\mathcal{X} = \{0,1\}^n$. You should convince yourself that for the uniform distribution $H(X) = n$. Let's now consider a slightly different distribution, by making our example above even simpler. Let's suppose that a "key generation box" ouputs $X = 1\ldots1$ with probability $1/2$, and with probability $1/2$ it outputs one of the other strings with equal probability. That is,

$$P_X(1\ldots1) = \frac{1}{2}\ , \tag{18}$$

$$P_X(x) = \frac{1}{2}\frac{1}{2^n - 1} \text{ for } x \neq 1\ldots1\ . \tag{19}$$

A small calculation (do it!) shows that for this new distribution $H(X) \approx n/2$. At first glance, this may not seem quite so bad - the entropy is still reasonably high and indeed there seems to be a lot of randomness on the remaining strings.

What do you think? For a moment, imagine you are the eavesdropper Eve who does not have side information E, but does in fact know the distribution $\vec{\rho}_X$. Would you say that X is a secure key?

3 Min-entropy

After a moments thought you might object that we can indeed guess X very well - after all it's $1\ldots1$ with probability $1/2$. Hence, if Alice were to use X as her key, you could try to decrypt the ciphertext using $1\ldots1$ as your key and you'd be right with probability $1/2$ no matter how long the message would be! Yet, the Shannon entropy scales with n. It's thus clear that we would like to have a measure that does not depend on the average of the distribution, but rather captures "worst case behaviour".

Such a measure is given by the so-called *min-entropy*, which is defined as

$$H_{\min}(X) := H_{\min}(\vec{\rho}_X) = -\log \max_x P_X(x)\ . \tag{20}$$

It thus only depends on the largest peak of the distribution. When guessing which outcome occured, we do indeed output the element with the highest probability. That is we can also think of $\max_x P_X(x) = P_{\text{guess}}(X)$ as the probability of correctly guessing x when we only know its probability distribution.

For our little example above, note that $H_{\min}(X) = 1$! That is, even though that Shannon entropy is large (indeed we can make it arbitrarily large by increasing n), the min-entropy is a constant independent of n! This is indeed consistent with our observations above of how secure X would be as a key.

3.1 Conditional min-entropy

Let us now add Eve's side information E back into the problem, and define what we mean by the entropy of X *conditioned* on side information E. Do you have some ideas how we might define it?

Attempt 1 A first attempt might be to define the conditional min-entropy as the average. I.e. $H_{\min}(X|E) = \sum_e P_E(e) H_{\min}(X|E=e)$. Yet, note that this does not coincide with our notion of guessing. Guessing according to the maximum likelihood approach above would give us

$$P_{\text{guess}}(X|E) = \sum_e P_E(e) P_{\text{guess}}(X|E=e) = \sum_e P_E(e) \max_x P_{X|E=e}(x) \ . \tag{21}$$

Clearly, our definition does not have the property that $H_{\min}(X|E) = -\log P_{\text{guess}}(X|E)$ as in the uncondition case above.

Attempt 2 A second attempt might this to define

$$H_{\min}(X|E) = -\log \sum_e P_E(e) 2^{-H_{\min}(X|E=e)} = -\log P_{\text{guess}}(X|E) \ . \tag{22}$$

It turns out that this is indeed the "right" definiton of the conditional min-entropy, and we will see in the next lecture that it is a very useful measure to capture the inherent randomness contained in X.

3.2 Properties

Let us now establish some very useful properties of the conditional min-entropy - you will prove some of them in your homework!

Lemma 3.1. *The conditional min-entropy has the following properties for all $\vec{\rho}_{XYE}$ where X, Y and E are classical*[2]

1. $\log |X| \geq H_{\min}(X|E) \geq 0$
2. *(Monotonicity)* $H_{\min}(XY|E) \geq H_{\min}(X|E)$.
3. *(Data processing)* $H_{\min}(X|f(E)) \geq H_{\min}(X|E)$ *for any function f applied to E.*
4. *(Chain rule)* $H_{\min}(X|E) \geq H_{\min}(XE) - \log |E|$.

[2]Quantum information behaves rather differently.