Information Theory

Ruixin Guo

Department of Computer Science
Kent State University

November 9, 2023

Contents

1. Entropy

2. Data Compression

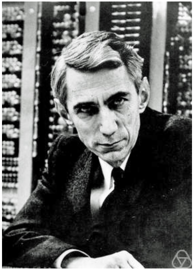
3. Data Transmission

The Story

The information theory was proposed in 1948 by Claude Shannon in his paper *A Mathematical Theory of Communication* [1].

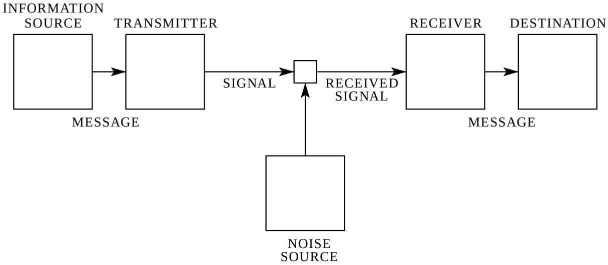
In the early 1940s, the researchers thought it is not possible to send information at a positive rate with negligible probability of error. Shannon surprised the communication theory community by proving the error could be made nearly zero for all communication rates below the channel capacity.

Information Theory has become the foundation of most communication technologies nowadays.



Claude Shannon

General Communication System



A general communication process is shown above. The information source has a set of messages $S = \{S_1, S_2, ..., S_n\}$ and each time it picks a message S_i to send. The message S_i is first encoded by the transmitter to a form X_i that is suitable for transferring. X_i is sent through a channel (usually noisy) and received by the receiver as Y_i. The receiver decode Y_i as S_i' and send it to the destination.

Note that if the channel is noisy, we may have $S_i \neq S_i'$, which means the message is not transferred correctly.

Contents

1. Entropy
2. Data Compression
3. Data Transmission

Units of Information

One idea that inspires Shannon is the general information definition proposed by Hartley [2].

The amount of information of a information source can be measured by the size of the message set S. Any monotonic function of $|S|$ can be considered as a measure of information, but the logarithm function is the most intuitive one.

Consider we have a storage device. The device can store 1 bit, and the bit can either be 0 or 1. Suppose the device store a unit of information. If we have n such devices, the amount of information should be n units, but the number of messages it can generate is 2^n. The logarithm function exactly builds such a mapping from 2^n to n.

Moreover, a message set of size $|S|$ means each message will have $1/|S|$ probability to be chosen. We can link the information of a message to the probability it occurs.

Units of Information

Definition 1.1 (Units of Information): If a message x occurs with probability $p(x)$, then the information it contains is $\log \frac{1}{p(x)}$.

The logarithm usually uses base 2. In this case a unit of information is called a bit. In general, to transfer a base 2 measure to a base b measure, we simply scale the quantity $\log_2 \frac{1}{p(x)}$ by a constant $\log_b 2$.

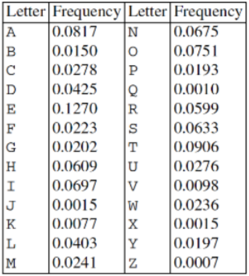
Intuitively, the units of information $\log \frac{1}{p(x)}$ is the minimum number of bits that required to describe x unambiguously.

$p(x)$ being small means we are more uncertain about x, thus $\log \frac{1}{p(x)}$ is large, which means x contains more information. If we are more certain about x, then x contains less information. If we are 100% sure about x, $p(x) = 1$, then the information x contains is 0.

Entropy

Assume the messages to be discrete. Let the set of all messages be $\mathcal{X} = \{x_1, x_2, ..., x_n\}$. Each $x_i \in \mathcal{X}$ is a message chosen from $\mathcal{X}$. Let the random variable X be the information source where $P(X = x_i) = p(x_i)$. How much information does X contain?

If X satisfies uniform distribution, $p(x_i) = \frac{1}{n}$ for every $i = 1, ..., n$, then the information of X is $\log n$. However, X does not always satisfy uniform distribution. For example, let X be a random variable of letters in English. It is known that the letters in English occur with different probabilities.



Letter	Frequency	Letter	Frequency
A	0.0817	N	0.0675
B	0.0150	O	0.0751
C	0.0278	P	0.0193
D	0.0425	Q	0.0010
E	0.1270	R	0.0599
F	0.0223	S	0.0633
G	0.0202	T	0.0906
H	0.0609	U	0.0276
I	0.0697	V	0.0098
J	0.0015	W	0.0236
K	0.0077	X	0.0015
L	0.0403	Y	0.0197
M	0.0241	Z	0.0007

Entropy

To measure the information of X of an arbitrary distribution, Shannon proposed entropy, which is the most basic concept in information theory.

Definition 1.2 (Entropy): The entropy $H(X)$ of a discrete random variable X is defined as

$$H(X) = \sum_{x \in \mathcal{X}} -p(x) \log p(x)$$

Remark: The entropy $H(X)$ can also be interpreted as the expectation of $\log \frac{1}{p(X)}$, where X is drawn from a distribution with the PMF $p(X)$:

$$H(X) = \mathbb{E}_X \left[\log \frac{1}{p(X)} \right]$$

Note that when $X = x$, $\log \frac{1}{p(x)}$ is the minimum number of bits that required to describe it. Therefore, entropy is the average of minimum number of bits to describe X. When X generates n messages, the total length of the messages is around $nH(X)$ as $n \to \infty$.

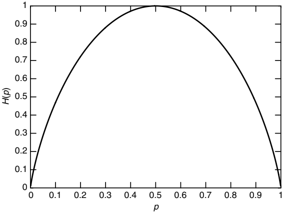
Entropy

Lemma 1.3: $H(X) \geq 0$.
Proof: $0 \leq p(x) \leq 1$ implies $\log \frac{1}{p(x)} \geq 0$.

Example: Suppose X satisfies Bernoulli distribution. $P(X=1)=p$ and $P(X=0)=1-p$. Then

$$H(X) = -p\log p - (1-p)\log(1-p) \overset{\text{def}}{=\!=} H(p)$$

The graph of the function $H(p)$ is shown on the right side. $H(p)=0$ when $p=0$ or 1, which means X has no uncertainty. $H(p)=1$ when $p=\frac{1}{2}$, which means X has most uncertainty when $P(X=0)=P(X=1)=\frac{1}{2}$.



Lemma 1.4: When $n = |\mathcal{X}|$ is finite, $H(X) \leq \log n$.
Proof: Since $\log$ is a concave function, by Jenson's Inequality (See Appendix 1),

$$H(X) = \mathbb{E}_X\left[\log\frac{1}{p(X)}\right] \leq \log \mathbb{E}_X\left[\frac{1}{p(X)}\right] = \log\sum_{i=1}^{n} p(x_i)\frac{1}{p(x_i)} = \log n$$

The equality is taken when $p(x_1)=p(x_2)=...=p(x_n)=\frac{1}{n}$.

Joint Entropy and Conditional Entropy

Definition 1.5 (Joint Entropy): The joint entropy $H(X,Y)$ of a pair of discrete random variables (X,Y) with a joint distribution $p(x,y)$ is defined as

$$H(X,Y) = -\sum_{x\in\mathcal{X}}\sum_{y\in\mathcal{Y}} p(x,y)\log p(x,y)$$

Definition 1.6 (Conditional Entropy): If $(X,Y)\sim p(x,y)$, the conditional entropy $H(Y|X)$ is defined as

$$\begin{aligned} H(Y|X) &= \sum_{x\in\mathcal{X}} p(x)H(Y|X=x) \\ &= -\sum_{x\in\mathcal{X}} p(x)\sum_{y\in\mathcal{Y}} p(y|x)\log p(y|x) \\ &= -\sum_{x\in\mathcal{X}}\sum_{y\in\mathcal{Y}} p(x,y)\log p(y|x) \\ &= \mathbb{E}_{X,Y}\left[\log\frac{1}{p(Y|X)}\right] \end{aligned}$$

The joint probability $H(X,Y)$ is the total information that the two random variables X and Y contains. The conditional probability $H(Y|X)$ is when X is known, the information (uncertainty) still remained in Y.

Chain Rule

The definition of joint entropy and conditional entropy naturally brings the following rule:

Theorem 1.7 (Chain Rule):

$$H(X,Y) = H(X) + H(Y|X)$$

Proof:

$$\begin{aligned} H(X,Y) &= -\sum_{x\in\mathcal{X}}\sum_{y\in\mathcal{Y}} p(x,y)\log p(x,y) \\ &= -\sum_{x\in\mathcal{X}}\sum_{y\in\mathcal{Y}} p(x,y)\log p(x)p(y|x) \\ &= -\sum_{x\in\mathcal{X}}\sum_{y\in\mathcal{Y}} p(x,y)\log p(x) - \sum_{x\in\mathcal{X}}\sum_{y\in\mathcal{Y}} p(x,y)\log p(y|x) \\ &= -\sum_{x\in\mathcal{X}} p(x)\log p(x) - \sum_{x\in\mathcal{X}}\sum_{y\in\mathcal{Y}} p(x,y)\log p(y|x) \\ &= H(X) + H(Y|X) \end{aligned}$$

Corollary 1.8: For random variables $X_1, X_2, ..., X_n$, we have

$$H(X_1, X_2, ..., X_n) = \sum_{i=1}^{n} H(X_i|X_{i-1}, ..., X_1)$$

Mutual Information

The Chain Rule says the total information of X and Y equals to the information of X plus the information of Y given X to be known.

Because X and Y are interchangeable, we have

$$H(X,Y) = H(X) + H(Y|X) = H(Y) + H(X|Y)$$

Here $H(Y|X) \neq H(X|Y)$. Let

$$I(X;Y) = H(X) - H(X|Y) = H(Y) - H(Y|X) \tag{1}$$

We call $I(X;Y)$ the mutual information of X and Y. It represents the common information that X and Y shares.

Definition 1.9 (Mutual Information): Consider two random variables X and Y with joint PMF $p(x,y)$ and marginal PMFs $p(x)$ and $p(y)$, then the mutual information $I(X;Y)$ is defined as

$$I(X;Y) = \sum_{x\in\mathcal{X}} \sum_{y\in\mathcal{Y}} p(x,y) \log \frac{p(x,y)}{p(x)p(y)} = \mathbb{E}_{X,Y}\left[\log \frac{p(X,Y)}{p(X)p(Y)}\right]$$

Mutual Information

Now we verify Eq (1):

$$\begin{aligned} I(X;Y) &= \sum_{x\in\mathcal{X}}\sum_{y\in\mathcal{Y}} p(x,y)\log\frac{p(x,y)}{p(x)p(y)} \\ &= \sum_{x\in\mathcal{X}}\sum_{y\in\mathcal{Y}} p(x,y)\log\frac{p(y|x)}{p(y)} \\ &= -\sum_{x\in\mathcal{X}}\sum_{y\in\mathcal{Y}} p(x,y)\log p(y) + \sum_{x\in\mathcal{X}}\sum_{y\in\mathcal{Y}} p(x,y)\log p(y|x) \\ &= H(Y) - H(Y|X) = H(X) - H(X|Y) \end{aligned}$$

Intuitively, $H(Y)$ is the uncertainty of Y, $H(Y|X)$ is the uncertainty remained in Y given a hint X. $I(X;Y) = H(Y) - H(Y|X)$ is the uncertainty of Y reduced by X, which is the information that X contains about Y.

Lemma 1.10: $I(X;Y) \geq 0$.
Proof: Using Jenson's Inequality. $-\log$ is a convex function. Thus,

$$\begin{aligned} I(X;Y) &= \mathbb{E}_{X,Y}\left[-\log\frac{p(X)p(Y)}{p(X,Y)}\right] \geq -\log\mathbb{E}_{X,Y}\left[\frac{p(X)p(Y)}{p(X,Y)}\right] \\ &= -\log\sum_{x\in\mathcal{X}}\sum_{y\in\mathcal{Y}} p(x,y)\frac{p(x)p(y)}{p(x,y)} = -\log\sum_{x\in\mathcal{X}} p(x)\sum_{y\in\mathcal{Y}} p(y) = 0 \end{aligned}$$

Mutual Information

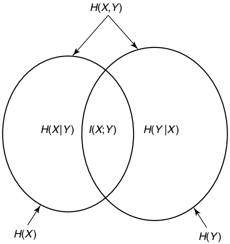
Corollary 1.11: $H(Y) \geq H(Y|X),\ H(X) \geq H(X|Y)$.

Corollary 1.12: $I(X;Y) = H(X) + H(Y) - H(X,Y)$.
Proof: Combine $I(X;Y) = H(Y) - H(Y|X)$ and $H(X,Y) = H(X) + H(Y|X)$.

Corollary 1.13: $I(X;X) = H(X) - H(X|X) = H(X)$.

Corollary 1.14: $I(X,Y) = 0$ if and only if X and Y are independent.

The following Venn diagram shows the relationship between entropy and mutual information.



Kullback–Leibler Divergence

The Kullback-Leibler Divergence, also called KL Divergence or relative entropy, measures the extra information required to describe a distribution $p(x)$ using the code of another distribution $q(x)$.

Definition 1.15 (KL Divergence): Let $x \in \mathcal{X}$ and $p(x), q(x)$ be two PMFs of x. The KL Divergence between $p(x)$ and $q(x)$ is defined as

$$D(\, p \,||\, q\,) = \sum_{x \in \mathcal{X}} p(x) \log \frac{p(x)}{q(x)} = \mathbb{E}_p \left[\log \frac{p(x)}{q(x)} \right]$$

Since $0 \log \frac{0}{q} = 0$ for any $q \geq 0$ [1], let $A = \{x : p(x) > 0\}$, we haves

$$D(\, p \,||\, q\,) = \sum_{x \in \mathcal{X}} p(x) \log \frac{p(x)}{q(x)} = \sum_{x \in A} p(x) \log \frac{p(x)}{q(x)}$$

Since $p \log \frac{p}{0} = \infty$ for any $p > 0$, we know that if there exists an $x \in \mathcal{X}$ such that $p(x) > 0$ and $q(x) = 0$, then $D(\, p \,||\, q\,)$ will be infinity.

[1] When $q = 0$, we define $0 \log \frac{0}{0} = 0$. This is because $\lim_{p \to 0, q \to 0} p \log \frac{p}{q} = \lim_{p \to 0} p \log p$ $- \lim_{p \to 0, q \to 0} p \log q = - \lim_{p \to 0, q \to 0} \log q^p$. We define $0^0 = 1$, as explained here https://math.stackexchange.com/questions/259514/how-to-define-the-00

Lemma 1.16: $D(\,p\,||\,q\,) \geq 0$. $D(\,p\,||\,q\,) = 0$ if and only if $p(x) = q(x)$ for all x.
Proof: Let $A = \{x : p(x) > 0\} \subseteq \mathcal{X}$. Using Jenson's Inequality. $-\log$ is a convex function. Thus,

$$D(\,p\,||\,q\,) = \sum_{x\in\mathcal{X}} p(x)\log\frac{p(x)}{q(x)} = \sum_{x\in A} p(x)\log\frac{p(x)}{q(x)}$$
$$\geq -\log\sum_{x\in A} p(x)\frac{q(x)}{p(x)} = -\log\sum_{x\in A} q(x) \geq -\log\sum_{x\in\mathcal{X}} q(x) = 0$$

The inequality holds only when $q(x)/p(x) = c$ for all $x \in A$ where c is a constant. Thus $\sum_{x\in A} q(x) = c\sum_{x\in A} p(x) = c$.
Since

$$c = \sum_{x\in A} q(x) \leq \sum_{x\in\mathcal{X}} q(x) = 1$$

And

$$-\log\sum_{x\in\mathcal{X}} p(x)\frac{q(x)}{p(x)} = -\log\sum_{x\in A} q(x) = \log\frac{1}{c}$$

To let $-\log\frac{1}{c} = 0$, we must have $c = 1$. Thus $p(x) = q(x)$. □

The KL Divergence can be considered as a distance between two probability distribution p and q. However, it is not a true distance since it is not symmetric, i.e., $D(\,p\,||\,q\,) \neq D(\,q\,||\,p\,)$. The distance also does not satisfy triangle inequality. When we want to use the distribution $q(x)$ to approximate the true distribution $p(x)$, we can use $D(\,p\,||\,q\,)$ to measure the error.

Note that the mutual information $I(X;Y)$ is a special case of KL Divergence: $I(X;Y) = D(\,p(x,y)\,||\,p(x)p(y)\,)$.

The KL Divergence $D(\,p\,||\,q\,)$ can be splited as

$$D(\,p\,||\,q\,) = \sum_{x\in\mathcal{X}} p(x)\log\frac{p(x)}{q(x)} = \underbrace{\sum_{x\in\mathcal{X}} p(x)\log\frac{1}{q(x)}}_{H(p,q)} - \underbrace{\sum_{x\in\mathcal{X}} p(x)\log\frac{1}{p(x)}}_{H(p)}$$

We call $H(p,q)$ the cross entropy of p and q [2], which means the average number of bits using the code of q to represent p. $H(p)$ is the entropy of p.

[2] If $X \sim p(x)$, we also write as $H(X)$ as $H(p)$, but $H(p,q)$ is denoted as the cross entropy not the joint entropy of p and q.

Conditional Mutual Information

Definition 1.17 (Conditional Mutual Information): The conditional mutual information of random variables X, Y and Z is defined by

$$I(X;Y|Z) = H(X|Z) - H(X|Y,Z) = \mathbb{E}_{X,Y,Z}\left[\log \frac{p(X,Y|Z)}{p(X|Z)p(Y|Z)}\right]$$

More precisely,

$$\begin{aligned}
I(X;Y|Z) &= H(X|Z) - H(X|Y,Z) \\
&= -\sum_{x,z} p(x,z)\log p(x|z) - \left(-\sum_{x,y,z} p(x,y,z)\log p(x|y,z)\right) \\
&= -\sum_{x,y,z} p(x,y,z)\log p(x|z) + \sum_{x,y,z} p(x,y,z)\log p(x|y,z) \\
&= \sum_{x,y,z} p(x,y,z)\log \frac{p(x|y,z)}{p(x|z)} = \sum_{x,y,z} p(x,y,z)\log \frac{p(x,y|z)}{p(x|z)p(y|z)} \\
&= \mathbb{E}_{X,Y,Z}\left[\log \frac{p(X,Y|Z)}{p(X|Z)p(Y|Z)}\right]
\end{aligned}$$

Chain Rule of Conditional Mutual Information

Theorem 1.18 (Chain Rule of Mutual Information):

$$I(X,Y;Z) = I(X;Z) + I(Y;Z|X)$$

Proof:

$$\begin{aligned} I(X,Y;Z) &= H(X,Y) - H(X,Y|Z) \\ &= H(X) + H(Y|X) - H(X|Z) - H(Y|X,Z) \\ &= [H(X) - H(X|Z)] + [H(Y|X) - H(Y|X,Z)] \\ &= I(X;Z) + I(Y;Z|X) \end{aligned}$$

Since X and Y are interchangeable, we also have

$$I(X,Y;Z) = I(Y;Z) + I(X;Z|Y)$$

Corollary 1.19: For random variables $X_1, X_2, ..., X_n$ and Y,

$$I(X_1, X_2, ..., X_n; Y) = \sum_{i=1}^{n} I(X_i; Y | X_{i-1}, ..., X_1)$$

Data Processing Inequality

Definition 1.20: We say random variables X, Y, Z form a Markov chain in that order if the distribution of Y is conditioned on X, the distribution of Z is conditioned on Y but not conditioned on X. Denoted as $X \to Y \to Z$. Specifically, the joint distribution of X, Y, Z can be written as

$$p(x, y, z) = p(x)p(y|x)p(z|y)$$

In fact, $X \to Y \to Z$ is a First Order Markov Chain [3]. Suppose we have two data processing function f_1, f_2 to process the data step by step. let $Y = f_1(X), Z = f_2(Y)$. Here Z only depends on Y. If Z also depends on X, the data processing function should be $Z = f_2(X, Y)$.

Lemma 1.21: $X \to Y \to Z$ if and only if X and Z are conditionally independent given Y.

Proof:

$$p(x, z|y) = \frac{p(x, y, z)}{p(y)} = \frac{p(x, y)p(z|y)}{p(y)} = p(x|y)p(z|y)$$

[3] https://stats.stackexchange.com/questions/2457/markov-process-that-depends-on-present-state-and-past-state

Data Processing Inequality

Lemma 1.22: $X \to Y \to Z$ implies $Z \to Y \to X$.

This is because X and Z are interchangeable since they are conditionally independent.

Theorem 1.23 (Data Processing Inequality): If $X \to Y \to Z$, then

$$I(X;Y) \geq I(X;Z)$$

Proof: By Theorem 1.18, we know

$$I(X;Y,Z) = I(X;Y) + I(X;Z|Y) = I(X;Z) + I(Y;Z|X)$$

By Lemma 1.21 and Corollary 1.14, X and Z are conditionally independent given Y, thus $I(X;Z|Y) = 0$. Since $I(Y;Z|X) \geq 0$, we have $I(X;Y) \geq I(X;Z)$. □

Corollary 1.24: If $X \to Y \to Z$, then

$$I(Y;Z) \geq I(X;Z)$$

Proof: By Lemma 1.22, $X \to Y \to Z$ implies $Z \to Y \to X$. Apply Theorem 1.23 to $Z \to Y \to X$. □

The Data Processing Inequality says, suppose Z is obtained from X, if there are more steps of data processing between X and Z, then fewer information will be shared by X and Z. No data processing methods can increase the shared information of X and Z.

Fano's Inequality

Suppose we wish to estimate a random variable X with an unknown distribution $p(x)$. We observe a random variable Y that is related to X by the conditional distribution $p(y|x)$. From Y, we calculate $\hat{X} = g(Y)$ as the estimator of X.

If $H(X|\hat{X}) = 0$, then we can estimate X from $\hat{X}$ with zero probability of error. The smaller $H(X|\hat{X})$ means we can estimate X from $\hat{X}$ with lower probability of error. Fano's Inequality gives an upper bound of $H(X|\hat{X})$ to measure how difficult we can estimate X from $\hat{X}$.

Theorem 1.25 (Fano's Inequality): For any estimator $\hat{X}$ such that $X \to Y \to \hat{X}$, with $P_e = P(X \neq \hat{X})$, we have

$$H(P_e) + P_e \log |\mathcal{X}| \geq H(X|\hat{X}) \geq H(X|Y)$$

Proof:
Define the error random variable

$$E = \begin{cases} 1 & \text{if } X \neq \hat{X} \\ 0 & \text{if } X = \hat{X} \end{cases}$$

Fano's Inequality

Using the chain rule, we have

$$\begin{aligned} H(E, X|\hat{X}) &= H(X|\hat{X}) + \underbrace{H(E|X, \hat{X})}_{=0} \\ &= H(E|\hat{X}) + H(X|E, \hat{X}) \end{aligned}$$

Since

$$H(E|\hat{X}) \leq H(E) = -P_e \log P_e - (1 - P_e) \log(1 - P_e) = H(P_e)$$

And

$$\begin{aligned} H(X|E, \hat{X}) &= P(E = 0) H(X|E = 0, \hat{X}) + P(E = 1) H(X|E = 1, \hat{X}) \\ &\leq (1 - P_e)0 + P_e H(X) \leq P_e \log |\mathcal{X}| \end{aligned}$$

We have $H(X|\hat{X}) \leq H(P_e) + P_e \log |\mathcal{X}|$. Since $H(X) = H(X|Y) + I(X;Y) = H(X|\hat{X}) + I(X;\hat{X})$ and by Theorem 1.23, $I(X;Y) \geq I(X;\hat{X})$, we have $H(X|Y) \leq H(X|\hat{X})$. □

Contents

1. Entropy
2. Data Compression
3. Data Transmission

Code

Definition 2.1 (Code): A code C for a random variable X is a mapping from $\mathcal{X}$, the set of all possible values of X, to $\mathcal{C}$, the set of codewords. For an $x \in \mathcal{X}$, we denote $C(x)$ as the codeword of x, and $l(x)$ as the length of $C(x)$.

A codeword is a finite length string of symbols. If one element in the string can represent d different symbols, we call the codeword d-ary.

For example, let $\mathcal{X} = \{a, b, c\}$, $C(a) = 00, C(b) = 10, C(c) = 11$, then $\mathcal{C} = \{00, 10, 11\}$. Since the symbol of each element in a codeword string can only be 0 or 1, $d = 2$. So the codeword is binary.

Different code methods include Huffman Code, Hamming Code, ASCII Code and so on.

Definition 2.2: The expected length of the codeword $C(x)$ for a random variable X with PMF $p(x)$ is defined as

$$L(C) = \sum_{x \in \mathcal{X}} p(x) l(x)$$

Code

Definition 2.3: A code C is nonsingular if for any $x, x' \in \mathcal{X}$,

$$x \neq x' \implies C(x) \neq C(x')$$

Nonsingularity is the sufficient condition for an unambiguous description of a single value of X.

Definition 2.4: The extension of code C is defined as the concatenation of codewords,

$$C(x_1 x_2 ... x_n) = C(x_1) C(x_2) ... C(x_n)$$

For example, if $C(x_1) = 00, C(x_2) = 11$, then $C(x_1 x_2) = 0011$.

Definition 2.5: A code is called uniquely decodable if its extension is non-singular.

For example, if a code C satisfies $C(x_1) = 0$, $C(x_2) = 01$, $C(x_3) = 10$. Then the extension code 010 can be decoded as either $x_1 x_3$ or $x_2 x_1$. We say C is not uniquely decodable.

Code

Definition 2.6: A code is called prefix code if no codeword is the prefix of any other codeword.

A prefix code can be decoded without reference the future codewords since the end of codeword is immediately recognizable.

Example 2.7: Huffman Code is prefix code. Suppose $\mathcal{X} = \{a, b, c, d\}$ where

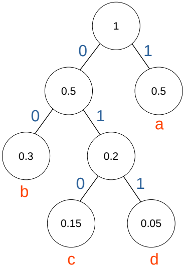
$$P(X = a) = 0.5,\ P(X = b) = 0.3,\ P(X = c) = 0.15,\ P(X = d) = 0.05$$

Construct a Huffman Tree as the right side showed, then we can get the codewords:

$$C(a) = 1,\ C(b) = 00,\ C(c) = 010,\ C(d) = 011$$

Given an extension codeword: 010001010000111

We decode it in this way: Scan the extension codeword from left to right. Starting from the root of the Huffman tree, when meet a 0, go to the left child; when meet a 1, go to the right child. When reach a leaf, return the symbol, and go back to the root. Continue the process until the end of the extension codeword.



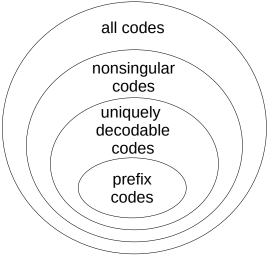
Therefore, the extension code is parsed as $010\ 00\ 1\ 010\ 00\ 011\ 1$ and decoded as $cbacbda$.

Example 2.8: The prefix code is uniquely decodable but not all uniquely decodable code is prefix code. For example, let the code be

$$C(a) = 10,\ C(b) = 00,\ C(c) = 11,\ C(d) = 110$$

This is not a prefix code because $C(c)$ is a prefix of $C(d)$. However, it is uniquely decodable. When the first two bits are 01 or 00, we can decode it immediately. When the first two bits are 11, if it follows odd number of 0s, then the first codeword must be 110; if it follows even number of 0s, the first codeword must be 11. Here we can see that the decoding process requires the future bits to decode the current codeword.

The relationships between nonsingular codes, uniquely decodable codes and prefix codes are shown in the right figure.



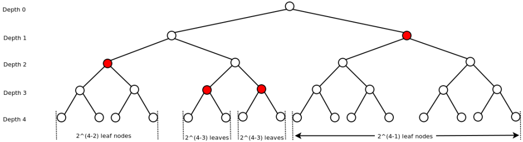
Kraft Inequality

Theorem 2.9 (Kraft Inequality): For any binary prefix code, the codeword lengths $l_1, l_2, ..., l_n$ must satisfy the inequality

$$\sum_{i=1}^{n} 2^{-l_i} \leq 1$$

Conversely, given a set of codeword lengths that satisfy this inequality, there exists an prefix code with these word lengths.

Proof:
$\Longrightarrow$: Let $l_{\mathsf{max}} \geq \max\{l_1, l_2, ..., l_n\}$ be the maximum depth of a full binary tree, then the tree has $2^{l_{\mathsf{max}}}$ leafs. Let a node A_i of depth l_i represent a codeword, then the subtree rooted at A_i will not contain another node to represent another codeword, since the code is prefix.



The subtree rooted at A_i has $2^{l_{\mathsf{max}}-l_i}$ leafs. Each codeword gives such a subtree, and the sum of all leafs in these subtrees will not exceed $2^{l_{\mathsf{max}}}$, thus

$$\sum_{i=1}^{n} 2^{l_{\mathsf{max}}-l_i} \leq 2^{l_{\mathsf{max}}} \implies \sum_{i=1}^{n} 2^{-l_i} \leq 1$$

$\impliedby$: For any $l_1, l_2, ..., l_n$ that satisfies Kraft Inequality, we can always construct a tree as shown in the previous page. Use the preorder traversal, label the first node of depth l_1 to be the codeword 1, then remove all the descendants of the node. Repeating this step to get all the codewords. □

Corollary 2.10: The Kraft Inequality for a d-ary code is

$$\sum_{i=1}^{n} d^{-l_i} \leq 1$$

since a d-ary code can be represented by a d-ary tree.

The Kraft Inequality gives the limitation of the codeword lengths of a prefix code.

Optimal Codes

Consider we have a random variable X satisfying the discrete distribution $p = \{p_1, p_2, ..., p_n\}$. We code the symbol of probability p_i by a codeword of length l_i. The optimal code is the one that minimizes the expected code length $L = \sum_{i=1}^n p_i l_i$.

Theorem 2.11: The expected length L of any binary code for a random variable X is greater or equal to the entropy $H(X)$. That is,

$$L \geq H(X)$$

with the equality if and only if $p_i = 2^{-l_i}$.

Proof: We can write the difference between the expected length and the entropy as

$$L - H(X) = \sum_{i=1}^n p_i l_i - \sum_{i=1}^n p_i \log \frac{1}{p_i} = -\sum_{i=1}^n p_i \log 2^{-l_i} + \sum_{i=1}^n p_i \log p_i$$

Define $r_i = \frac{2^{-l_i}}{\sum_{j=1}^n 2^{-l_j}}$, $c = \sum_{j=1}^n 2^{-l_j}$. Note that $r = \{r_1, r_2, ..., r_n\}$ is another distribution since $\sum_{i=1}^n r_i = 1$. Then

$$L - H(X) = -\sum_{i=1}^{n} p_i \log r_i c + \sum_{i=1}^{n} p_i \log p_i = \sum_{i=1}^{n} p_i \log \frac{p_i}{r_i} - \sum_{i=1}^{n} p_i \log c$$
$$= D(\, p \,||\, r \,) - \log c$$

By Theorem 1.16, $D(\, p \,||\, r \,) \geq 0$. By Theorem 2.9, $c \leq 1$. Thus, $L - H(X) \geq 0$.

$L - H(X) = 0$ if and only if $D(\, p \,||\, r \,) = 0$ and $c = 1$. That is, $p_i = 2^{-l_i}$ for every i. □

To let $L = H(X)$, we need $l_i = -\log p_i$ for every i. The codeword length l_i is a integer, but $\log \frac{1}{p_i}$ is not always a integer. Thus L usually cannot reach $H(X)$.

We can obtain the integer code length by round up the $\log \frac{1}{p_i}$ term, that is, each l_i is defined as

$$l_i = \left\lceil \log \frac{1}{p_i} \right\rceil \qquad (2)$$

Then the $l_1, l_2, ..., l_n$ obtained in this way forms a legal code since it satisfies the Kraft Inequality:

$$\sum_{i=1}^{n} 2^{-\left\lceil \log \frac{1}{p_i} \right\rceil} \leq \sum_{i=1}^{n} 2^{-\log \frac{1}{p_i}} \leq 1$$

By Eq (2) we have:

$$\log \frac{1}{p_i} \leq l_i \leq \log \frac{1}{p_i} + 1$$

Multiplying by p_i and summing over i, we get

$$H(X) \leq L \leq H(X) + 1 \tag{3}$$

Theorem 2.12 (Bound on the Optimal Code): Let $l_1^*, l_2^*, ..., l_n^*$ be optimal codeword lengths and $L^* = \sum_{i=1}^n p_i l_i^*$, then

$$H(X) \leq L^* \leq H(X) + 1$$

Proof: By letting $l_i = \left\lceil \log \frac{1}{p_i} \right\rceil$, and $L = \sum_{i=1}^n p_i l_i$. L satisfies Eq (3). L may not be an optimal length, so for the optimal length L^*, we have $L^* \leq L$. Since $H(X)$ is the minimum for any expected codeword length, we have

$$H(X) \leq L^* \leq L \leq H(X) + 1$$

Contents

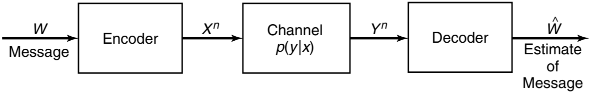
1. Entropy

2. Data Compression

3. Data Transmission

Channel

Definition 3.1 (Discrete Channel): A discrete channel is defined as an input set $\mathcal{X}$, an output set $\mathcal{Y}$ and a probability transition matrix $p(y|x)$ that expresses the probability of the output symbol y given the input symbol x we sent.



$p(y|x)$ is a $|\mathcal{X}| \times |\mathcal{Y}|$ matrix where $p(y_j|x_i)$ represents the probability that the input x_i transfers to the output y_j.

Definition 3.2 (Channel Capacity): The channel capacity of a discrete memoryless channel is defined as

$$C = \max_{p(x)} I(X;Y)$$

where the maximum is taken over all input distributions $p(x)$.

In general, we can consider $I(X;Y)$ as a function of $p(x)$ since

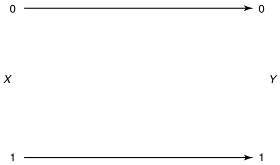
$$I(X;Y) = \sum_{x,y} p(x,y) \log \frac{p(x,y)}{p(x)p(y)} = \sum_{x,y} p(y|x)p(x) \log \frac{p(y|x)}{\sum_x p(y|x)p(x)}$$

where $p(y|x)$ is fixed, determined by the channel.

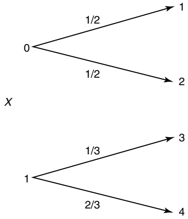
Channel

The channel can be noiseless and noisy. The noiseless channel and noisy channel with nonoverlapping outputs are errorless.

Consider we send message $X = \{0, 1\}$. Since given Y, we can map it back to X with probability 1 correctness, $I(X;Y) = H(X) - H(X|Y) = H(X)$. And the channel capacity is $C = \max I(X;Y) = 1$ bit, obtained by choosing the probability $p(x) = (\frac{1}{2}, \frac{1}{2})$.



Noiseless Channel



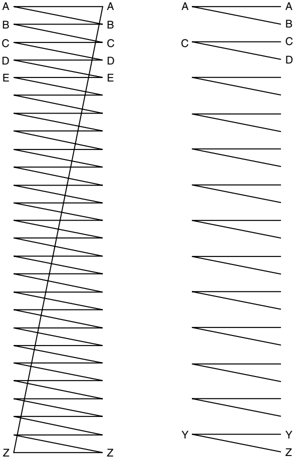
Noisy Channel with Nonoverlapping Outputs

Channel

The noisy typewriter is a channel that the input is either received unchanged in the output with probability $\frac{1}{2}$ or shift to the next character with probability $\frac{1}{2}$.

Suppose the input set has 26 characters, we know that $H(Y|X) = 2 * (-\frac{1}{2}\log\frac{1}{2})$ $+24 * 0 \log 0 = 1$ bit, and $C = \max I(X;Y) = \max(H(Y) - H(Y|X))$ $= \log 26 - 1 = \log 13$ bits.

This channel capacity is equivalent to choosing 13 characters as a subset of the input and transmit them without error.



Noisy channel

Noiseless subset of inputs

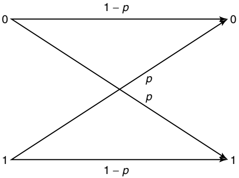
Noisy Typewriter

Channel

The binary symmetric channel has 0 and 1 as input symbols. Each symbol has p probability to shift to the other and $1-p$ probability to remain unchanged. Then

$$\begin{aligned} I(X;Y) &= H(Y) - H(Y|X) = H(Y) - \sum_x p(x) H(Y|X=x) \\ &= H(Y) - H(p) \sum_x p(x) = H(Y) - H(p) \leq 1 - H(p) \end{aligned}$$

Thus, $C = \max I(X;Y) = 1 - H(p)$.



Binary Symmetric Channel

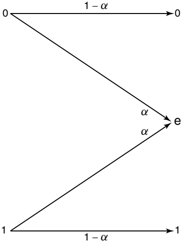
Channel

The binary erasure channel has two input symbols 0 and 1 and three output symbols $0, 1$ and e, where e means "erased". Each input symbol has $1-\alpha$ probability to remain unchanged and α probability to be erased.

The channel capasity is as follows:

$$C = \max_{p(x)} I(X;Y) = \max_{p(x)}(H(Y) - H(Y|X)) = \max_{p(x)} H(Y) - H(\alpha)$$

Now we want to find $p(x)$ to maximize $H(Y)$.



Binary Erasure Channel

Channel

Suppose $P(X=1)=\pi$ and $P(X=0)=1-\pi$, then $P(Y=0)=(1-\pi)(1-\alpha)$, $P(Y=e)=\alpha$, $P(Y=1)=\pi(1-\alpha)$. Thus,

$$\begin{aligned} H(Y) &= -(1-\pi)(1-\alpha)\log(1-\pi)(1-\alpha) - \alpha\log\alpha - \pi(1-\alpha)\log\pi(1-\alpha) \\ &= (1-\alpha)H(\pi) + H(\alpha) \end{aligned}$$

Therefore,

$$\begin{aligned} C &= \max_{p(x)} H(Y) - H(\alpha) \\ &= \max_{\pi}(1-\alpha)H(\pi) + H(\alpha) - H(\alpha) \\ &= (1-\alpha)\max_{\pi} H(\pi) \\ &= 1-\alpha \end{aligned}$$

Symmetric Channel

A channel is called symmetric if the rows of its transition matrix are permutations of each other, so are the columns. For example,

$$p(y|x) = \begin{pmatrix} \frac{1}{3} & \frac{1}{3} & \frac{1}{6} & \frac{1}{6} \\ \frac{1}{6} & \frac{1}{6} & \frac{1}{3} & \frac{1}{3} \end{pmatrix}$$

Because each row is the permutation of each other, the sum of each row will be the same, so as the columns. Since the sum of each row is 1, the sum of each column will be $\frac{|\mathcal{X}|}{|\mathcal{Y}|}$.

Let $\boldsymbol{r}$ be a row of the matrix, we have

$$I(X;Y) = H(Y) - H(Y|X) = H(Y) - H(\boldsymbol{r}) \leq \log|\mathcal{Y}| - H(\boldsymbol{r})$$

The probability $p(x) = \frac{1}{|\mathcal{X}|}$ is the maximizer of $I(X;Y)$. Observed that for each y,

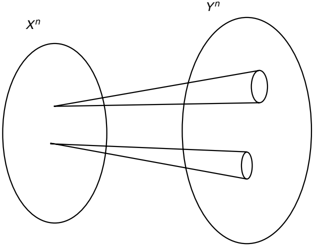
$$p(y) = \sum_{x\in\mathcal{X}} p(y|x)p(x) = \frac{1}{|\mathcal{X}|}\sum_{x\in\mathcal{X}} p(y|x) = \frac{1}{|\mathcal{X}|}\frac{|\mathcal{X}|}{|\mathcal{Y}|} = \frac{1}{|\mathcal{Y}|}$$

Preview of the Channel Coding Theorem

Analogous to the noisy typewriter channel. Suppose the input is a set of codewords of n-sequences, each symbol in the n-sequence contains $H(X)$ bits information, and the input set contains $nH(X)$ bits information. The number of codewords in the input set is $2^{nH(X)}$. Similarly, let the number of codewords in the output set is $2^{nH(Y)}$.

Suppose each input maps to a set of $2^{nH(Y|X)}$ outputs. To ensure zero decoding error, the sets of outputs should not be overlapped. We can have at most $2^{nH(Y)}/2^{nH(Y|X)} = 2^{nI(X;Y)}$ non-overlapping output sets, which means the number of distinguishable codewords in the input set is at most $2^{nI(X;Y)}$. So there are at most $nI(X;Y)$ bits information for n-sequence that can be transmitted without error. That is, each symbol in the n-sequence can have at most $I(X;Y)$ bits information.

We will rigorously prove this idea in the following slides.



Channel Coding

Definition 3.3: An (M, n) code for the channel $(\mathcal{X}, p(y|x), \mathcal{Y})$ consists of the following:

(1) An index set $\{1, 2, ..., M\}$

(2) An encoding function $X^n : \{1, 2, ..., M\} \to \mathcal{X}^n$, yielding codewords $x^n(1), x^n(2), ..., x^n(M)$.

(3) A decoding function $g : \mathcal{Y}^n \to \{1, 2, ..., M\}$

The channel communication process is: 1. Choose an index $i \in \{1, 2, ..., M\}$, encoded as $x^n(i)$; 2. Send $x^n(i)$ as the input of the channel, the channel will give an output $y^n \sim p(y^n|x^n(i))$. 3. Decode y^n by $i' = g(y^n)$. If $i = i'$, we say the message is correctly sent.

We say a channel is memoryless if $p(y^n|x^n) = \prod_{i=1}^n p(y_i|x_i)$. That is, the bits in the n-sequence is i.i.d. distributed. Note that $x_1, x_2, ..., x_n$ may be dependent, but for each x_i, the distribution $p(y|x_i)$ is independent.

Memoryless means the channel has no feedback. In general, a feedback channel satisfies $p(y^n|x^n) = \prod_{i=1}^n p(y_i|x_i, y_{i-1})$.

Error Definition

Definition 3.4: We define

$$\lambda_i = P(g(Y^n) \neq i | X^n = x^n(i)) = \sum_{y_n} p(y^n | x^n(i)) \mathbf{1}[g(y^n) \neq i]$$

as the conditional probability of error given that index i was sent, where **1** is the indicator function.

Given a index i, encoded as $x^n(i)$. The channel will map $x^n(i)$ to different y_n with different probability. Some of these y_ns can be decoded as i correctly while others cannot. λ_i represents the probability of incorrect decoding.

Definition 3.5: The maximum probability of error $\lambda^{(n)}$ for an (M, n) code is defined as

$$\lambda^{(n)} = \max_{i \in \{1,2,\ldots,M\}} \lambda_i$$

Code Rate

Definition 3.6: The rate R of an (M, n) code is defined as

$$R = \frac{\log M}{n} \qquad \text{bits per transmission}$$

Example 3.7: Suppose we have a symbol set $\{a, b, c, d\}$ where each symbol occurs with the same probability. Let X be the random variable of the symbol set, then $H(X) = 4 \cdot (-\frac{1}{4} \log \frac{1}{4}) = 2$ bits. We can use 2 bits to encode the symbol set: $a \to 00, b \to 01, c \to 10, d \to 11$. This gives the minimum average length of code. Each symbol contains 1 bit information.

We can also use $n = 4$ bits to encode the symbol set. For example, $a \to 0000, b \to 0101, c \to 1010, d \to 1111$. Then each symbol will contain $R = \frac{H(X)}{n} = \frac{2}{4} = \frac{1}{2}$ bits information on average.

Note that there are $2^4 = 16$ different 4-bit sequences. We only use $2^{nR} = 2^{H(X)} = 4$ of them for encoding.

Definition 3.8: A rate R is said to be achievable if there exists a sequence of $(2^{nR}, n)$ codes such that the maximum probability of error $\lambda^{(n)} \to 0$ as $n \to \infty$.

Jointly Typical Sequences

Consider a channel whose input is X and output is Y. Given Y, on what condition can we map it back to X (without any prior knowledge)? The condition is that X and Y are not independent. Equivalently, $p(x,y) \neq p(x)p(y)$, $I(X;Y) > 0$. [4]

Definition 3.9: The set $A_\epsilon^{(n)}$ of jointly typical sequences $\{(x^n, y^n)\}$ with respect to the distribution $p(x,y)$ is the set of n-sequences with empirical entropies ϵ-close to the true entropies.

$$
\begin{aligned}
A_\epsilon^{(n)} = \{ & (x^n, y^n) \in \mathcal{X}^n \times \mathcal{Y}^n : \\
& \left| -\frac{1}{n} \log p(x^n) - H(X) \right| < \epsilon, \\
& \left| -\frac{1}{n} \log p(y^n) - H(Y) \right| < \epsilon, \\
& \left| -\frac{1}{n} \log p(x^n, y^n) - H(X,Y) \right| < \epsilon \}
\end{aligned}
$$

[4] In cryptography, when the distribution of the plaintext X and cyphertext Y are independent, we call this Perfect Security, i.e., there is no way to know about X given Y. https://www.cs.purdue.edu/homes/hmaji/teaching/Fall2016/lectures/03.pdf

Theorem 3.10: Let (X^n, Y^n) be sequences n drawn i.i.d. according to $p(x^n, y^n) = \prod_{i=1}^n p(x_i, y_i)$, then:

(1) $P((X^n, Y^n) \in A_\epsilon^{(n)}) \to 1$ as $n \to \infty$.

(2) $|A_\epsilon^{(n)}| \leq 2^{n(H(X,Y)+\epsilon)}$

(3) If $\tilde{X}^n, \tilde{Y}^n$ are independent, $\tilde{X}^n \sim p(x^n)$ and $\tilde{Y}^n \sim p(y^n)$ where $p(x^n)$ and $p(y^n)$ are the marginal distributions of $p(x^n, y^n)$, then

$$P((\tilde{X}^n, \tilde{Y}^n) \in A_\epsilon^{(n)}) \leq 2^{-n(I(X;Y)-3\epsilon)}$$

Also, for sufficiently large n,

$$P((\tilde{X}^n, \tilde{Y}^n) \in A_\epsilon^{(n)}) \geq (1-\epsilon)2^{-n(I(X;Y)+3\epsilon)}$$

Proof:
(1) Since $-\frac{1}{n}\log p(x^n) = -\frac{1}{n}\sum_{i=1}^n \log p(x_i)$ and $\mathbb{E}[\log p(x_i)] = H(X)$, by Weak Law of Large Numbers, we know that $-\frac{1}{n}\log p(x^n) \xrightarrow{p} H(X)$ as $n \to \infty$. That is, for any $\epsilon > 0$,

$$\lim_{n\to\infty} P(|-\frac{1}{n}\log p(x^n) - H(X)| \geq \epsilon) = 0$$

Similar for $H(Y)$ and $H(X,Y)$.

Since $P((X^n, Y^n) \in A_\epsilon^{(n)}) \leq 1$, we have

$$\lim_{n\to\infty} P((X^n, Y^n) \in A_\epsilon^{(n)}) \leq 1$$

Moreover,

$$\begin{aligned}
&\lim_{n\to\infty} P((X^n, Y^n) \in A_\epsilon^{(n)}) = 1 - \lim_{n\to\infty} P((X^n, Y^n) \notin A_\epsilon^{(n)}) \\
&= 1 - \lim_{n\to\infty} P\left(\left|-\frac{1}{n}\log p(x^n) - H(X)\right| \geq \epsilon \cup \left|-\frac{1}{n}\log p(y^n) - H(Y)\right| \geq \epsilon \cup \right. \\
&\qquad \left. \left|-\frac{1}{n}\log p(x^n, y^n) - H(X,Y)\right| \geq \epsilon\right) \\
&\geq 1 - \left[\lim_{n\to\infty} P\left(\left|-\frac{1}{n}\log p(x^n) - H(X)\right| \geq \epsilon\right) + \lim_{n\to\infty} P\left(\left|-\frac{1}{n}\log p(y^n) - H(Y)\right| \geq \epsilon\right)\right. \\
&\qquad \left. + \lim_{n\to\infty} P\left(\left|-\frac{1}{n}\log p(x^n, y^n) - H(X,Y)\right| \geq \epsilon\right)\right] \\
&= 1
\end{aligned}$$

Thus,

$$\lim_{n\to\infty} P((X^n, Y^n) \in A_\epsilon^{(n)}) = 1$$

(2) Since

$$\begin{aligned}
&\left|-\frac{1}{n}\log p(x^n, y^n) - H(X,Y)\right| < \epsilon \\
\Longleftrightarrow & -\epsilon < -\frac{1}{n}\log p(x^n, y^n) - H(X,Y) < \epsilon \\
\Longleftrightarrow & 2^{-n(H(X,Y)+\epsilon)} < p(x^n, y^n) < 2^{-n(H(X,Y)-\epsilon)} \qquad (4)
\end{aligned}$$

We have

$$\begin{aligned}
1 = \sum p(x^n, y^n) \geq \sum_{(x^n,y^n)\in A_\epsilon^{(n)}} p(x^n, y^n) \geq \sum_{(x^n,y^n)\in A_\epsilon^{(n)}} 2^{-n(H(X,Y)+\epsilon)} \\
= |A_\epsilon^{(n)}| 2^{-n(H(X,Y)+\epsilon)}
\end{aligned}$$

Thus,

$$|A_\epsilon^{(n)}| \leq 2^{n(H(X,Y)+\epsilon)}$$

(3) By Eq (4), similarly, we have

$$2^{-n(H(X)+\epsilon)} < p(x^n) < 2^{-n(H(X)-\epsilon)}$$
$$2^{-n(H(Y)+\epsilon)} < p(y^n) < 2^{-n(H(Y)-\epsilon)}$$

Since $\tilde{X}^n \sim p(x^n)$ and $\tilde{Y}^n \sim p(y^n)$ and $\tilde{X}^n, \tilde{Y}^n$ are independent, $(\tilde{X}^n, \tilde{Y}^n) \sim p(x^n)p(y^n)$. And the probability that $(\tilde{X}^n, \tilde{Y}^n)$ is in $A_\epsilon^{(n)}$ is

$$\begin{aligned} P((\tilde{X}^n, \tilde{Y}^n) \in A_\epsilon^{(n)}) &= \sum_{(\tilde{X}^n, \tilde{Y}^n) \in A_\epsilon^{(n)}} p(x^n)p(y^n) \\ &\leq 2^{n(H(X,Y)+\epsilon)} 2^{-n(H(X)-\epsilon)} 2^{-n(H(Y)-\epsilon)} \\ &= 2^{-n(I(X;Y)-3\epsilon)} \end{aligned}$$

For sufficiently large n, we have $P((X^n, Y^n) \in A_\epsilon^{(n)}) \geq 1-\epsilon$, thus

$$1-\epsilon \leq \sum_{(x^n,y^n) \in A_\epsilon^{(n)}} p(x^n, y^n) \leq \sum_{(x^n,y^n) \in A_\epsilon^{(n)}} 2^{-n(H(X,Y)-\epsilon)} = |A_\epsilon^{(n)}| 2^{-n(H(X,Y)-\epsilon)}$$
$$\iff |A_\epsilon^{(n)}| \geq (1-\epsilon) 2^{n(H(X,Y)-\epsilon)}$$

Therefore,

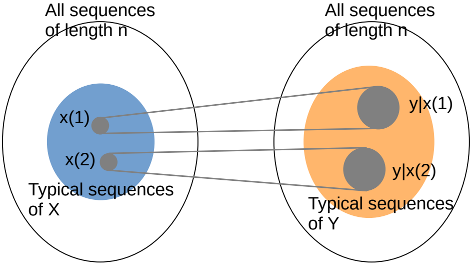
$$
\begin{aligned}
P((\tilde{X}^n, \tilde{Y}^n) \in A_\epsilon^{(n)}) &= \sum_{(\tilde{X}^n, \tilde{Y}^n) \in A_\epsilon^{(n)}} p(x^n) p(y^n) \\
&\geq (1-\epsilon) 2^{n(H(X,Y)-\epsilon)} 2^{-n(H(X)+\epsilon)} 2^{-n(H(Y)+\epsilon)} \\
&= (1-\epsilon) 2^{-n(I(X;Y)+3\epsilon)}
\end{aligned}
$$

□

Intuition of Joint Typical Sequences:

- For n-sequence, there will be 2^n possible messages. However, if each symbol contains $H(X)$ bits information, then only $2^{nH(X)}$ messages will be used for coding X^n. Note that for binary symbols, $H(X) \leq 1$. Similar for Y^n.
- The number of joint sequences $2^{nH(X;Y)} = 2^{nH(X)} 2^{nH(Y|X)}$. Each input message can be decoded to $2^{nH(Y|X)}$ output messages based on how noisy the channel is. If the channel is noiseless, then $H(Y|X) = 0$, any input will give a unique output. So $2^{nH(X;Y)}$ is the number of possible pairs of the input-output sequences. $(X^n, Y^n) \in A_\epsilon^{(n)}$ means the sequences X^n and Y^n form a pair.

- Suppose we have a bipartite graph where the left side has x nodes, right side has y nodes, and there are $D \leq xy$ edges. Each node on the right side has D/y edges on average connected to it. If we consider the nodes on the left side that connect to a fixed node on the right side, then there will be xy/D disjoint set on the left side. Note that D/xy is the probability that we find an edge from all combinations of the left side and right side nodes. Since the randomly chosen $(\tilde{X}^n, \tilde{Y}^n)$ has $2^{-nI(X;Y)}$ probability to be in $A_\epsilon^{(n)}$, there will be $2^{nI(X;Y)}$ messages that we can choose from all input messages without making confusion.
- Since the distribution $p(x)$ may not be the ideal distribution that gives $H(X)$, we add an error term ϵ.



Channel Coding Theorem

The characterization of the channel capacity as the maximum mutual information is the central and most famous success of information theory. Shannon proved that information can be sent reliably over a channel at all rates up to the channel capacity.

Theorem 3.11 (Channel Coding Theorem): For a discrete memoryless channel, all rates below capacity C are achievable.

(1) For every rate $R < C$, there exists a sequence of $(2^{nR}, n)$ codes with maximum probability of error $\lambda^{(n)} \to 0$.

(2) Conversely, any sequence of $(2^{nR}, n)$ codes with $\lambda^{(n)} \to 0$ must have $R < C$.

Proof of the Channel Coding Theorem

(1) Define the following events:

$$E_i = \{(X^n(i), Y^n) \in A_\epsilon^{(n)}\}, \qquad i \in \{1, 2, ..., 2^{nR}\}$$

Then $P(E_i) = P(g(Y^n) = i)$ and $P(E_i^c) = P(g(Y^n) \neq i)$.

Remember that $\lambda_i = P(g(Y^n) \neq i | X^n = x^n(i))$. Let $i = w$, the error happens when E_w is false or there exists E_i to be true for all $i \neq w$. That is, either Y^n cannot be decoded as w or Y^n can be decoded as both w and other indexes. Thus,

$$\begin{aligned}
\lambda_w &= P(E_1 \cup E_2 \cup ... \cup E_w^c \cup ... \cup E^{2^{nR}} | X^n = x^n(w)) \\
&\leq P(E_1^c | X^n = x^n(w)) + \sum_{i=1, i \neq w}^{2^{nR}} P(E_i | X^n = x^n(w)) \\
&\leq P(E_1^c | X^n = x^n(w)) + (2^{nR} - 1) 2^{-n(I(X;Y) - 3\epsilon)} \\
&\leq P(E_1^c | X^n = x^n(w)) + 2^{-n(I(X;Y) - R - 3\epsilon)}
\end{aligned}$$

By Theorem (1),

$$\lim_{n\to\infty} P(E_1^c|X^n = x^n(w)) = 1 - \lim_{n\to\infty} P(E_1|X^n = x^n(w)) = 0$$

To let $\lim_{n\to\infty} 2^{-n(I(X;Y)-R-3\epsilon)} = 0$, we need $I(X;Y) - R - 3\epsilon > 0$. For every $R < I(X;Y)$, there must exist $\epsilon > 0$ to make $R < I(X;Y) - 3\epsilon$. Thus we can choose n and ϵ to make $R < I(X;Y) - 3\epsilon$.

Therefore,

$$\lim_{n\to\infty} \lambda_w \leq \lim_{n\to\infty} P(E_1^c|X^n = x^n(w)) + \lim_{n\to\infty} 2^{-n(I(X;Y)-R-3\epsilon)} = 0$$

Since $\lim_{n\to\infty} \lambda_w = 0$ holds for any w, we must have $\lambda^{(n)} \to 0$.

Moreover, we can choose the input distribution $p(x)$ that maximizes $I(X;Y)$. This distribution corresponds to a code that enables $R < C$.

(2) Suppose the channel is a Markov Chain $W \to X^n(W) \to Y^n \to \hat{W}$. W is the random variable of the input set, encoded as $X^n(W)$. The channel takes the input $X^n(W)$ and outputs Y^n. Y^n is decoded as $\hat{W}$.

We first introduce a Lemma:

Lemma 3.12: Let Y^n be the result of passing X^n through a discrete memoryless channel of capacity C. Then

$$I(X^n; Y^n) \leq nC \qquad \text{for all } p(x^n)$$

Proof:

$$\begin{aligned} I(X^n; Y^n) &= H(Y^n) - H(Y^n|X^n) = H(Y^n) - \sum_{i=1}^{n} H(Y_i|Y_{i-1}, ..., Y_1, X^n) \\ &= H(Y^n) - \sum_{i=1}^{n} H(Y_i|X_i) \leq \sum_{i=1}^{n} H(Y_i) - \sum_{i=1}^{n} H(Y_i|X_i) \\ &= \sum_{i=1}^{n} I(X_i; Y_i) \leq nC \end{aligned}$$

The equality $=$ is because, since the channel is memoryless, Y_i only depends on X_i. □

Let $P_e^{(n)} = P(W \neq \hat{W})$. By Fano's Inequality,

$$H(W|\hat{W}) \leq H(P_e^{(n)}) + P_e^{(n)} nR \leq 1 + P_e^{(n)} nR$$

Therefore,

$$\begin{aligned} nR &= H(W) \\ &= H(W|\hat{W}) + I(W;\hat{W}) \\ &\leq 1 + P_e^{(n)} nR + I(W;\hat{W}) \\ &\leq 1 + P_e^{(n)} nR + I(X^n;Y^n) \qquad (5) \\ &\leq 1 + P_e^{(n)} nR + nC \qquad (6) \end{aligned}$$

Eq (5) is because of the data processing inequality. Since $W \to X^n(W) \to Y^n \to \hat{W}$, we have

$$I(W;\hat{W}) \leq I(W;Y^n) \leq I(X^n;Y^n)$$

Since

$$P_e^{(n)} = P(W \neq \hat{W}) = \sum_{w=1}^{2^{nR}} P(W \neq \hat{W} | W = w) P(W = w)$$

$$= \sum_{w=1}^{2^{nR}} \lambda_w P(W = w) \leq \lambda^{(n)} \sum_{w=1}^{2^{nR}} P(W = w) = \lambda^{(n)}$$

So $\lambda^{(n)} \to 0$ implies $P_e^{(n)} \to 0$.

By Eq (6) we have

$$R \leq \frac{1}{n} + P_e^{(n)} R + C$$

If there exists a code such that $\lambda^{(n)} \to 0$, when $n \to 0$, we have $\frac{1}{n} \to 0$ and $P_e^{(n)} \to 0$, thus $R \leq C$.

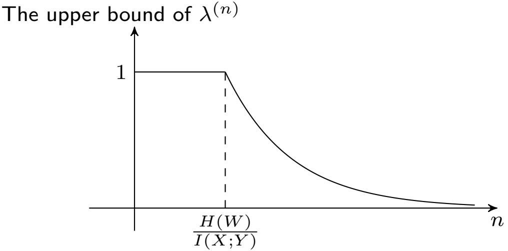
Intuition of the Channel Coding Theorem

For simplicity, suppose $\epsilon = 0$, we have

$$\lambda^{(n)} \leq p_n + 2^{-n(I(X;Y)-R)} = p_n + 2^{H(W)-nI(X;Y)}$$

where $p_n \to 0$ as $n \to \infty$ and $H(W) = nR = \log M$ is the entropy of the message set $\{1, 2, ..., M\}$. $I(X;Y)$ is the mutual information of a single symbol in the n-sequence.

Since $\lambda^{(n)}$ is the probability of error, $\lambda^{(n)} \leq 1$. It is possible for $\lambda^{(n)} < 1$ only when $R \leq I(X;Y) \Leftrightarrow n \geq \frac{H(W)}{I(X;Y)}$. Starting from the threshold $\frac{H(W)}{I(X;Y)}$, $\lambda^{(n)}$ will converge to 0 as $n \to \infty$.



Hamming Code

The channel coding theorem promises the existence of block codes that will allow us to transmit information at rates below capacity with an arbitrarily small probability of error if the block length is large enough.

An obvious way to construct such a coding is to simply repeat the information. For example, we encode 1 as 11111, 0 as 00000. Here the code length is $n = 5$, and the rate is $R = \frac{1}{5}$ bit per symbol. When the code passes a channel and gives an output, if there are more than 3 symbols to be 1, we decode it as 1; otherwise, we decode it as 0. error occurs if and only if more than three of the bits are changed. However, this code is not efficient since R goes to 0 with n increasing.

Hamming Code: We consider a binary code of block length 7. All operations will be done in a module 2 field. Consider the set of all nonzero binary vectors of length 3. Arrange them in columns to form a matrix:

$$H = \begin{bmatrix} 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 \end{bmatrix}$$

Hamming Code

The dimension of the range of H (i.e., rank of H) is 3. Thus, the dimension of the nullspace of H is 4. And we will have $2^4 = 16$ codewords in the nullspace:

$$\begin{array}{cccc}
0000000 & 0001111 & 0010110 & 0011001 \\
0100101 & 0101010 & 0110011 & 0111100 \\
1000011 & 1001100 & 1010101 & 1011010 \\
1100110 & 1101001 & 1110000 & 1111111
\end{array}$$

Note that no pair of columns in H is linearly dependent. While any two columns sum to a third column, giving a triple of linearly dependent column. This means we must choose at least three columns to make their sum to be 0, thus each codeword will have at least three 1s. We say the minimum weight of the codewords is 3.

Since the nullspace is linear and closed, the difference of any two codewords is also a codeword, which will also have at least three 1s. Each 1 means the symbols of the two codewords in that index is different, thus there will be at least 3 bits different for any two codewords in the nullspace. We say the minimum distance of the codewords is 3.

Hamming Code

This means, if a codeword has 1 bit error, we can correct it. If a codeword has 2 bits error, we can detect it but not able to correct it. Let e_i be the error vector where the ith index is 1 and other indexes are 0. Suppose we send c through a channel and receives $c + e_i$. We calculate

$$H(c + e_i) = Hc + He_i = He_i$$

and He_i is the ith column of H. In this way we can know which bit in c is corrupted and can fix the bit by reversing it.

Suppose each codeword occurs with the same probability, then the information of the codewords is $\log 16 = 4$ bits, and each symbol contains $\frac{4}{7}$ bits information.

We call this particular Hamming Code as $[7, 4]$ Hamming Code[5]. In general, for any r, we can construct Hamming Code $[2^r - 1, 2^r - r - 1]$. The matrix H is of size $r \times (2^r - 1)$. The codeword length is $2^r - 1$. The total number of messages is $2^{2^r - r - 1}$ and the information is $2^r - r - 1$ bits. The rate is $R = \frac{2^r - r - 1}{2^r - 1} = 1 - \frac{r}{2^r - 1}$.

[5] https://en.wikipedia.org/wiki/Hamming(7,4)

References

[1] Claude Elwood Shannon. A mathematical theory of communication. The Bell system technical journal, 27(3):379–423, 1948. `https://people.math.harvard.edu/~ctm/home/text/others/shannon/entropy/entropy.pdf`

[2] Hartley, R. V. L.. Transmission of Information. Bell System Technical Journal, p. 535, 1928.

[3] Thomas Cover and Thomas Joy. Elements of Information Theory, Second Edition. 2006. (Important!).

[4] `https://web.stanford.edu/class/ee376a/files/scribes/lecture15.pdf`

[5] `https://users.math.msu.edu/users/halljo/classes/codenotes/Hamming.pdf`

[6] John Duchi. Lecture Notes on Statistics and Information Theory. `https://web.stanford.edu/class/stats311/lecture-notes.pdf`

References

[7] https://modelingsimulation.github.io/TeachingWriting2020/Resources/reports/ShengLiu.pdf

[8] Kolmogorov Complexity.
https://www.cs.princeton.edu/courses/archive/fall11/cos597D/L10.pdf

[9] Fenchel Duality
https://ocw.mit.edu/courses/6-253-convex-analysis-and-optimization-spring-2012/8508c98dd2a400f91035eb2241727d7a_MIT6_253S12_lec01.pdf

[10] Fenchel-Moreau Theorem https://www2.math.ethz.ch/education/bachelor/lectures/hs2015/math/mf/lecture7notes

Appendix 1: Jensen's Inequality

Theorem A.1 (Jensen's Inequality): If f is a convex function and X is a random variable, then

$$\mathbb{E}[f(X)] \geq f(\mathbb{E}[X])$$

Moreover, if f is strictly convex, $\mathbb{E}[f(X)] = f(\mathbb{E}[X])$ implies $X = \mathbb{E}[X]$ with probability 1, i.e., X is a constant.

Proof: Consider discrete distribution, for two points x_1, x_2, let p_1, p_2 be the probabilities where $p_1 + p_2 = 1$. By the definition of convex function, we have

$$p_1 f(x_1) + p_2 f(x_2) \geq f(p_1 x_1 + p_2 x_2)$$

By induction, suppose there exist $k-1$ points $x_1, ..., x_k$ such that for any probabilities $p_1, ..., p_{k-1}$ where $\sum_{i=1}^{k-1} p_i = 1$,

$$\sum_{i=1}^{k-1} p_i f(x_i) \geq f(\sum_{i=1}^{k-1} p_i x_i)$$

holds, then for k points, let $p_i' = p_i/(1-p_k)$ for $i = 1, 2, ..., k-1$ such that $\sum_{i=1}^{k-1} p_i' = 1$,

Appendix 1: Jensen's Inequality

$$
\begin{aligned}
\sum_{i=1}^{k} p_i f(x_i) &= p_k f(x_k) + \sum_{i=1}^{k-1} p_i f(x_i) \\
&= p_k f(x_k) + (1-p_k) \sum_{i=1}^{k-1} p'_i f(x_i) \\
&\geq p_k f(x_k) + (1-p_k) f(\sum_{i=1}^{k-1} p'_i x_i) \\
&\geq f(p_k x_k + \sum_{i=1}^{k-1} p'_i x_i) \geq f(\sum_{i=1}^{k} p_i x_i)
\end{aligned}
$$

When f is μ-strongly convex, for 2 points, by definition,

$$p_1 f(x_1) + p_2 f(x_2) \geq f(p_1 x_1 + p_2 x_2) + \mu \frac{p_1 p_2}{2} \|x_1 - x_2\|^2$$

Thus the equality can only be taken when $x_1 = x_2$. Similar for ≥ 2 points.

Appendix 2: Variational Representation of KL Divergence

The KL Divergence can be represented as variation, which simplifies the calculation.

Theorem A.2.1 (Donsker-Varadhan Representation): Let $g(x)$ be any measurable function, $p(x)$ and $q(x)$ be two distributions, and e be the base of the logarithm. The KL Divergence equals to the following variational representation:

$$D(\,p\,||\,q\,) = \sup_{g(x)}\{\mathbb{E}_{p(x)}[g(x)] - \log \mathbb{E}_{q(x)}[e^{g(x)}]\}$$

Proof: Denote

$$t(x) = \frac{e^{g(x)}q(x)}{\sum_x e^{g(x)}q(x)} = \frac{e^{g(x)}q(x)}{\mathbb{E}_{q(x)}[e^{g(x)}]}$$

Then $t(x)$ is a distribution function. Note that

$$\mathbb{E}_{p(x)}\left[\log \frac{t(x)}{q(x)}\right] = \mathbb{E}_{p(x)}\left[\log \frac{e^{g(x)}q(x)}{\mathbb{E}_{q(x)}[e^{g(x)}]q(x)}\right]$$
$$= \mathbb{E}_{p(x)}[\log e^{g(x)}] - \mathbb{E}_{p(x)}[\log \mathbb{E}_{q(x)}[e^{g(x)}]] = \mathbb{E}_{p(x)}[g(x)] - \log \mathbb{E}_{q(x)}[e^{g(x)}]$$

Appendix 2: Variational Representation of KL Divergence

The above inequality uses the fact that $\mathbb{E}_{q(x)}[e^{g(x)}]$ is a constant. Therefore,

$$\begin{aligned} & D(\,p\,||\,q\,) - \left(\mathbb{E}_{p(x)}[g(x)] - \log \mathbb{E}_{q(x)}[e^{g(x)}]\right) \\ &= \mathbb{E}_{p(x)}\left[\log \frac{p(x)}{q(x)}\right] - \mathbb{E}_{p(x)}\left[\log \frac{t(x)}{q(x)}\right] \\ &= \mathbb{E}_{p(x)}\left[\log \frac{p(x)}{t(x)}\right] = D(\,p\,||\,t\,) \geq 0 \end{aligned}$$

Thus,

$$D(\,p\,||\,q\,) \geq \mathbb{E}_{p(x)}[g(x)] - \log \mathbb{E}_{q(x)}[e^{g(x)}]$$

The equality can be taken only when we choose $g(x)$ such that $t(x) = p(x)$. In other words, $g(x) = \log \frac{p(x)}{q(x)} + C$ for some constant $C \in \mathbb{R}$.

Appendix 2: Variational Representation of KL Divergence

Definition A.2.2 (f-Divergence): Let $f : [0, +\infty) \to \mathbb{R}$ be a convex function and $f(1) = 0$. Let $p(x)$ and $q(x)$ be two PMFs of $x \in \mathcal{X}$ The f-Divergence between $p(x)$ and $q(x)$ is defined as

$$D_f(\, p \,||\, q \,) = \sum_{x \in \mathcal{X}} q(x) f\left(\frac{p(x)}{q(x)}\right) = \mathbb{E}_{q(x)}\left[f\left(\frac{p(x)}{q(x)}\right)\right]$$

Note that KL Divergence is a special case of the f-divergence. Let $f(x) = -\log x$, we have $D(\, q \,||\, p \,) = D_f(\, p \,||\, q \,)$.

Theorem A.2.3 (Variational Representation of f-Divergence): Let $g(x)$ be any measurable function, $p(x)$ and $q(x)$ be two distributions, and e be the base of the logarithm. Let f be convex and lower semicontinuous. The f-Divergence equals to the following variational representation:

$$D_f(\, p \,||\, q \,) = \sup_{g(x)} \{\mathbb{E}_{p(x)}[g(x)] - \mathbb{E}_{q(x)}[f^*(g(x))]\}$$

where $f^*(y) = \sup_{x \in \mathrm{dom} f}\{yx - f(x)\}$ is the convex conjugate of f.

Appendix 2: Variational Representation of KL Divergence

Proof: The proof will utilize convex conjugate and Fenchel-Moreau Theorem.

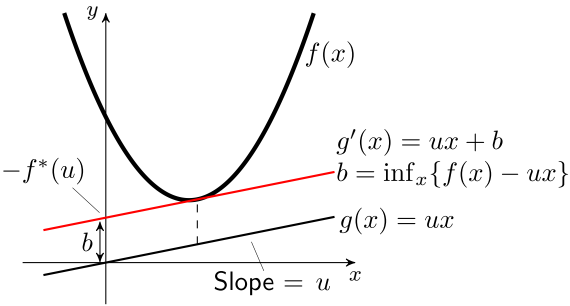
Definition A.2.4 (Convex Conjugate): For any function $f : \mathcal{X} \to \mathbb{R}$, its convex conjugate is defined as

$$f^*(u) = \sup_x \{ux - f(x)\} = -\inf_x \{f(x) - ux\}$$

Given u, we find x to minimize $f(x) - g(x)$ where $g(x) = ux$.

Draw the tangent of slope u of $f(x)$, denoted as $g'(x)$. Then the distance between $g(x)$ and $g'(x)$ is the minimum of $f(x) - g(x)$.

$g'(x)$ crosses the y-axis at $(0, -f^*(u))$.



Appendix 2: Variational Representation of KL Divergence

Analogous to that the signal can be described either in time domain or frequency domain, the function $f(x)$ and its convex conjugate $f^*(u)$ are just two different way to describe the epigraph of $f(x)$. For all the tangents of $f(x)$, f^* is a function of the crossing point of the tangent on y-axis with the slope of the tangent u.

Theorem A.2.5 (Fenchel-Moreau Theorem): Let f^{**} be the biconjugate function of f, i.e, f^{**} is the conjugate of f^*. If f is lower semicontinuous and convex, then $f^{**} = f$.

Proof: Since

$$\begin{aligned} f^{**}(x) &= \sup_u \{xu - f^*(u)\} \\ &= \sup_u \{xu - \sup_x \{ux - f(x)\}\} \\ &\geq \sup_u \{xu - ux + f(x)\} = f(x) \end{aligned}$$

We know that $f^{**}(x) \geq f(x)$.

Appendix 2: Variational Representation of KL Divergence

If f is convex and semicontinuous, then its epigraph is closed. We can write $f(x)$ as

$$f(x) = \sup_{a \leq f} \{a(x)\}$$

where $a(x) = kx + b, k \in \mathbb{R}, b \in \mathbb{R}$ is an affine function. This means that $f(x)$ can be expressed as the tangents of its epigraph.

For any $a \leq f$, consider

$$f^*(u) = \sup_x \{ux - f(x)\}$$
$$a^*(u) = \sup_x \{ux - a(x)\}$$

We have $a^*(u) \geq f^*(u)$. Similarly, $a^{**}(x) \leq f^{**}(x)$.
Note that

$$a^*(u) = \sup_x \{ux - kx - b\} = \begin{cases} -b & \text{if } u = k \\ +\infty & \text{if } u \neq k \end{cases}$$

Appendix 2: Variational Representation of KL Divergence

And

$$a^{**}(x) = \sup_u \{xu - a^*(u)\} = xk + b$$

The supremum is taken only when $u = k$ such that $a^*(u) = -b$. Otherwise, $xu - a^*(u) = -\infty$ and cannot get the supremum.

Therefore, we know that $a^{**}(x) = a(x)$, hence $a(x) \leq f^{**}(x)$. Note that this holds for any $a(x) \leq f(x)$. Thus, $f(x) = \sup_{a \leq f}\{a(x)\} \leq f^{**}(x)$. Therefore, $f^{**} = f$. □

Let's go back to the proof of Theorem A.2.3. By Theorem A.2.5, we can write

$$f(x) = \sup_u \{xu - f^*(u)\}$$

Thus,

$$D_f(\, p \,||\, q\,) = \mathbb{E}_{q(x)}\left[f\left(\frac{p(x)}{q(x)}\right)\right] = \mathbb{E}_{q(x)}\left[\sup_u \left\{u\frac{p(x)}{q(x)} - f^*(u)\right\}\right]$$

Appendix 2: Variational Representation of KL Divergence

$$
\begin{aligned}
&\geq \sup_u \left\{ \mathbb{E}_{q(x)} \left[u \frac{p(x)}{q(x)} - f^*(u) \right] \right\} = \sup_u \left\{ \mathbb{E}_{q(x)} \left[u \frac{p(x)}{q(x)} \right] - \mathbb{E}_{q(x)}[f^*(u)] \right\} \\
&= \sup_u \left\{ \sum_{x \in \mathcal{X}} q(x) u \frac{p(x)}{q(x)} - \mathbb{E}_{q(x)}[f^*(u)] \right\} = \sup_u \left\{ \sum_{x \in \mathcal{X}} p(x) u - \mathbb{E}_{q(x)}[f^*(u)] \right\} \\
&= \sup_u \left\{ \mathbb{E}_{p(x)} u - \mathbb{E}_{q(x)}[f^*(u)] \right\}
\end{aligned}
$$

Let $u = g(x)$, we proved the theorem. □

Corollary A.2.6:

$$
D(\, p \,||\, q \,) = \sup_{g(x)} \{ \mathbb{E}_{p(x)}[g(x)] - \mathbb{E}_{q(x)}[e^{g(x)-1}] \}
$$

Proof: Let $f(x) = x \log x$, then f is a convex and continuous function.

$$
D_f(\, p \,||\, q \,) = \sum_{x \in \mathcal{X}} q(x) \frac{p(x)}{q(x)} \log \left(\frac{p(x)}{q(x)} \right) = \sum_{x \in \mathcal{X}} p(x) \log \left(\frac{p(x)}{q(x)} \right) = D(\, p \,||\, q \,)
$$

Appendix 2: Variational Representation of KL Divergence

Since

$$f^*(u) = \sup_x \{ux - f(x)\} = \sup_x \{ux - x\log x\}$$

Let $g(x) = ux - x\log x \Rightarrow g'(x) = u - \log x - 1, g''(x) = -\frac{1}{x} \leq 0$. $g'(x) = 0$ if and only if $\log x = u - 1 \Rightarrow x = e^{u-1}$. Thus $f^*(u) = e^{u-1}$. By Theorem A.2.3,

$$D_f(\,p\,||\,q\,) = \sup_{g(x)}\{\mathbb{E}_{p(x)}[g(x)] - \mathbb{E}_{q(x)}[f^*(g(x))]\} = \sup_{g(x)}\{\mathbb{E}_{p(x)}[g(x)] - \mathbb{E}_{q(x)}[e^{g(x)-1}]\}$$

$\square$

Mutual Information Neural Estimation [6]: Consider we have two random variables X and Z, how to estimate the mutual information $I(X;Z)$?

For example, consider a real world dataset, the features $\{x_1, ..., x_n\}$ are sampled from an unknown distribution $p(x)$, and the labels $\{z_1, ..., z_n\}$ are sampled from an unknown distribution $p(z)$. Let X be the random variable of $p(x)$, Z be the random variable of $p(z)$, we would like to estimate $I(X;Z)$.

[6] Mohamed Ishmael Belghazi, Aristide Baratin, Sai Rajeshwar, Sherjil Ozair, Yoshua Bengio, Aaron Courville, Devon Hjelm Proceedings of the 35th International Conference on Machine Learning, PMLR 80:531-540, 2018.

Appendix 2: Variational Representation of KL Divergence

Remember that

$$I(X;Z) = \sum_{x,z} p(x,z) \log \frac{p(x,z)}{p(x)p(z)}$$

Since we don't know $p(x), p(z)$ and $p(x,z)$, estimating $I(X;Z)$ will be difficult.

However, remember that $I(X;Z) = D(\,p(x,z)\,||\,p(x)p(z)\,)$. And by Theorem A.2.1,

$$D(\,p(x,z)\,||\,p(x)p(z)\,) = \sup_{g(x,z)} \{\mathbb{E}_{p(x,z)}[g(x,z)] - \log \mathbb{E}_{p(x)p(z)}[e^{g(x,z)}]\}$$

Since $g(x,z)$ is from all measuable functions. We can let $g(x,z)$ be a neural network $T_\theta : X \times Z \to \mathbb{R}$. T_θ is parameterized by θ. Let $\mathcal{F}$ be the hypothesis space of T_θ, then $\mathcal{F}$ is a subset of all measuable functions, thus

$$D(\,p(x,z)\,||\,p(x)p(z)\,) \geq \sup_{T_\theta(x,z) \in \mathcal{F}} \{\mathbb{E}_{p(x,z)}[T_\theta(x,z)] - \log \mathbb{E}_{p(x)p(z)}[e^{T_\theta(x,z)}]\} \quad (7)$$

The right hand side of the above inequality can be an estimator of $I(X;Z)$.

Appendix 2: Variational Representation of KL Divergence

Since the neural network T_θ is an universal approximator, we can train T_θ to find the θ that reaches the supremum. $\mathbb{E}_{p(x,z)}$ and $\mathbb{E}_{p(x)p(z)}$ can be empirically calculated using the samples of the dataset.

Similar to Eq (7), by Corollary A.2.6, we have

$$D(\,p(x,z)\,||\,p(x)p(z)\,) \geq \sup_{T_\theta(x,z)\in\mathcal{F}} \{\mathbb{E}_{p(x,z)}[T_\theta(x,z)] - \mathbb{E}_{p(x)p(z)}[e^{T_\theta(x,z)-1}]\} \quad (8)$$

We usually use Eq (7) instead of Eq (8) to estimate $I(X;Z)$, because the right hand side of Eq (7) is closer to $I(X;Z)$ than the right hand side of Eq (8), based on the fact that $\log x \leq \frac{x}{e}$ for any $x > 0$.

Appendix 3: Kolmogrov Complexity

Recall that Shannon describes the information of an object by linking it to the probability it occurs with. For a random variable X, suppose the event $X = x$ is of probability $p(x)$, then the information the event has is $\log \frac{1}{p(x)}$. The information is one way to describe the complexity of the event.

In 1965, the mathematician Andrey Kolmogorov proposed another definition of complexity of an object, called Kolmogrov Complexity. This complexity is about not how likely the object will occur but how hard to describe the object. Consider we generate the object with an algorithm in a computer, Kolmogrov Complexity is defined as the length of the shortest binary computer program that describes the object.

Example: Consider we have 3 strings:

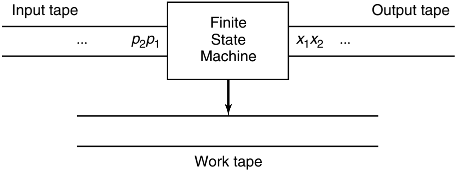
1. 0101010101010101010101010101010101010101010101010101010101010101
2. 0110101000001001111001100110011111110011101111001100100100001000
3. 1101111001110101111101101111101110101101111000101110010100111011

String 1 is easy to describe since it repeats 01. String 2 seems random and hard to describe, but it is the decimal part of $\sqrt{2} - 1$, so describing it is easy. String 3 is random and hard to describe. We can consider it as describing a length-n string with k 1s, so the complexity would be $\log n + nH(\frac{k}{n})$, see Example A.3.8.

Appendix 3: Kolmogrov Complexity

Turing Machine: The Turing machine is a conceptual computer that can simulate any computer algorithm, no matter how complicated it is. We also call the Turing machine a universal machine.

Consider a finite state Turing machine with an input tape, an output tape and a work tape. The input tape feeds a binary program $p_1 p_2 ...$ of finite length. The machine runs the program bu changing its state and record the state on the work tape. The output tape produces a binary string $x_1 x_2 ...$ of finite length or infinite length.



Appendix 3: Kolmogrov Complexity

Definition A.3.1 (Kolmogrov Complexity): Let x be a finite-length binary string and $\mathcal{U}$ be a universal computer. Let $l(x)$ denote the length of the string x. Let $\mathcal{U}(p)$ denote the output of the computer $\mathcal{U}$ when fed a binary program p. The Kolmogorov Complexity $K_{\mathcal{U}}(x)$ of a string x with respect to a universal computer $\mathcal{U}$ is defined as

$$K_{\mathcal{U}}(x) = \min_{p:\mathcal{U}(p)=x} l(p)$$

the minimum length over all programs that print x and halt. Thus, $K_{\mathcal{U}}(x)$ is the shortest description length of x over all descriptions interpreted by computer $\mathcal{U}$. Sometimes we simplify $K_{\mathcal{U}}(x)$ as $K(x)$.

Definition A.3.2 (Conditional Kolmogrov Complexity): Suppose $l(x)$ is known by the computer, we can define the conditional Kolmogrov Complexity knowing $l(x)$ as

$$K_{\mathcal{U}}(x|l(x)) = \min_{p:\mathcal{U}(p,l(x))=x} l(p)$$

which means we take both the program p and the length $l(x)$ as the input, the minimum $l(p)$ required to describe x.

Appendix 3: Kolmogrov Complexity

Theorem A.3.3: If $\mathcal{U}$ is a universal computer, then for any other computer $\mathcal{A}$ there exists a constant $c_{\mathcal{A}}$ such that

$$K_{\mathcal{U}}(x) \leq K_{\mathcal{A}}(x) + c_{\mathcal{A}}$$

Proof: Since $\mathcal{U}$ is a universal computer, it can run any algorithm, including the algorithm that simulates $\mathcal{A}$. The algorithm that for $\mathcal{U}$ to describe $\mathcal{A}$ is a constant, defined as $c_{\mathcal{A}}$. We can let $\mathcal{U}$ output x by running such a program that $\mathcal{U}$ simulates $\mathcal{A}$ and $\mathcal{A}$ outputs x. Thus the total length of the program will be $K_{\mathcal{A}}(x) + c_{\mathcal{A}}$. Any simpler programs for $\mathcal{U}$ to output x will not exceed this length.

Theorem A.3.4: Conditional complexity is less than the length of the sequence, i.e.,

$$K(x|l(x)) \leq l(x) + c$$

Proof: A program that prints x can be as follows:

`Print the following` l`-bit sequence:` $x_1 x_2 ... x_{l(x)}$

Since $l(x)$ is provided, the end of the program is clearly defined. Thus the length of the program is $l(x) + c$.

Appendix 3: Kolmogrov Complexity

Theorem A.3.5: $K(x) \leq K(x|l(x)) + 2\log l(x) + c$
Proof: We can design a program by combining both $l(x)$ and the program in Theorem A.3.4. $l(x)$ can be transferred to a binary code of $\log l(x)$ length. But if we simply send $\log l(x)$ bits, the program will not know where is the end of the length code (unless we know the length of the length code).

Instead, we repeat each bit of $l(x)$ twice, and add 01 to the end as an indication of the end of the length code. For example, if $l(x) = 5$, first convert it to binary code 101, then repeat each bit twice and get 110011, and finally add 01 to the end and get 11001101. Therefore, $l(x)$ can be represented by a binary code of length $2\log l(x) + 2$. Thus,

$$K(x) \leq K(x|l(x)) + 2\log l(x) + 2 + c_0 = K(x|l(x)) + 2\log l(x) + c$$

Theorem A.3.6: The number of strings x with complexity $K(x) < k$ satisfies

$$|\{x \in \{0,1\}^* : K(x) < k\}| < 2^k$$

Proof: The number of binary programs with length $< k$ is $\sum_{i=1}^{k-1} 2^i = 2^k - 1 < 2^k$. Each program corresponds to a string, so the total number of strings is $< 2^k$.

Appendix 3: Kolmogrov Complexity

Theorem A.3.7: Kolmogorov complexity is not computable.

Proof: Suppose there exist a program $Q(x)$ that takes the input string x and output $K(x)$. Then we can construct a program $P(n)$ as follows:

```
$P(n)$:
For all $x \in \{0,1\}^n$
   If $x$ is the first string of $K(x) > n$, use $Q$ to check this.
     Output $x$ and halt.
```

Suppose P takes the input n and outputs x_n, then $K(x_n) > n$. However, P is a program that describes x_n in $2 \log n + c$ length, because we encode n in $2 \log n$ bits and Q is of constant length. Thus $K(x_n) \leq 2 \log n + c$. This makes a contradiction. So Q does not exist.

Appendix 3: Kolmogrov Complexity

Example A.3.8: The Kolmogrov Complexity of a sequence of n bits of k ones.

We can design such a program to print any sequence of n bits of k ones:

```
Generate all sequence with $k$ ones in lexicographic order.
Of these sequences, print the $i$th sequence.
```

Suppose n is known by the computer. The only variable in the program is k and i. k is in the range $\{0, 1, ..., n\}$, which requires $\log n$ bits to describe. i is in the range $\{0, 1, ..., \binom{n}{k}\}$, which requires $\log \binom{n}{k}$ bits to describe. Thus, the total length of the program is

$$l(p) = c + \log n + \log \binom{n}{k}$$

Consider the Stirling's approximation[7]:

$$\sqrt{2\pi n}\left(\frac{n}{e}\right)^n < n! < \sqrt{2\pi n}\left(\frac{n}{e}\right)^n e^{\frac{1}{12n}}$$

[7] https://en.wikipedia.org/wiki/Stirling's_approximation

Appendix 3: Kolmogrov Complexity

Thus for any $n \geq 1$, using base 2 logarithm, we have

$$
\begin{aligned}
\binom{n}{k} &= \frac{n!}{(n-k)!k!} \leq \frac{\sqrt{2\pi n}\left(\frac{n}{e}\right)^{n} e^{\frac{1}{12n}}}{\sqrt{2\pi(n-k)}\left(\frac{n-k}{e}\right)^{n-k}\sqrt{2\pi k}\left(\frac{k}{e}\right)^{k}} \\
&= \sqrt{\frac{n}{2\pi(n-k)k}} \cdot \frac{n^n}{(n-k)^{n-k}k^k} \cdot e^{\frac{1}{12n}} \\
&= \sqrt{\frac{n}{2\pi(n-k)k}} \cdot 2^{n\log n-(n-k)\log(n-k)-k\log k+\frac{1}{12n}\log e} \\
&= \sqrt{\frac{n}{2\pi(n-k)k}} \cdot 2^{n\left[\log n-(1-\frac{k}{n})\log(n-k)-\frac{k}{n}\log k\right]+\frac{1}{12n}\log e} \\
&= \sqrt{\frac{n}{2\pi(n-k)k}} \cdot 2^{n\left[-(1-\frac{k}{n})\log(1-\frac{k}{n})-\frac{k}{n}\log\frac{k}{n}\right]+\frac{1}{12n}\log e} \\
&= \sqrt{\frac{n}{2\pi(n-k)k}} \cdot 2^{nH(\frac{k}{n})+\frac{1}{12n}\log e} = \sqrt{\frac{n}{\pi(n-k)k}} \cdot 2^{nH(\frac{k}{n})+\frac{1}{12n}\log e-\frac{1}{2}} \\
&\leq \sqrt{\frac{n}{\pi(n-k)k}} \cdot 2^{nH(\frac{k}{n})}
\end{aligned}
$$

Appendix 3: Kolmogrov Complexity

The last inequality is because $\log e \approx 1.442$ and $\frac{1}{12n}\log e - \frac{1}{2} \leq \frac{1}{12}\log e - \frac{1}{2} \approx -0.38 < 0$. For $k \neq 0$ or n and $n \geq 1$, we have

$$\log\sqrt{\frac{n}{\pi(n-k)k}} \leq \log\sqrt{\frac{n}{\pi(n-1)}} \leq \log\sqrt{\frac{2}{\pi}} \leq 1$$

Therefore,

$$\log\binom{n}{k} \leq \log\sqrt{\frac{n}{\pi(n-k)k}} + nH(\frac{k}{n}) \leq nH(\frac{k}{n}) + 1$$

Thus, for any string $x^n = x_1x_2...x_n$ of length n where $x_i \in \{0,1\}$ and $k = \sum_{i=1}^{n} x_i$, $k \neq 0$ or n and $n \geq 1$,

$$K(x^n|n) \leq c + \log n + \log\binom{n}{k} \leq c' + \log n + nH(\frac{k}{n}) \qquad (9)$$

If $k = 0$ or n, $K(x^n|n) = c$, because we can design a program to print fixed number of 0s or 1s.

Appendix 3: Kolmogrov Complexity

Theorem A.3.9: For any computer $\mathcal{U}$,

$$\sum_{p:\mathcal{U}(p)\ \text{halts}} 2^{-l(p)} \leq 1$$

Proof: For all the program that halts, no program will be the prefix of another program, otherwise the latter will never be executed to the end and halts. Thus such programs are prefix code and satisfy Kraft Inequality.

Theorem A.3.10 (Relationship of Kolmogorov Complexity and Entropy): Let the stochastic process $\{X_i\}$ be drawn i.i.d according to the PMF $p(x)$, where $x \in \{0, 1\}$. Let $p(x^n) = \prod_{i=1}^{n} p(x_i)$. Then there exists a constant c such that

$$H(X) \leq \frac{1}{n} \sum_{x^n} p(x^n) K(x^n|n) \leq H(X) + \frac{\log n}{n} + \frac{c}{n}$$

for all n. Consequently,

$$\mathbb{E}_{p(x^n)} \left[\frac{1}{n} K(x^n|n) \right] \rightarrow H(X)$$

as $n \rightarrow \infty$.

Appendix 3: Kolmogrov Complexity

Proof: Consider the lower bound. Since $K(x^n|n)$ is a length of the codeword of x^n, and $\mathbb{E}_{p(x^n)}\left[K(x^n|n)\right] \to H(X)$ is the average code length of x^n. By Theorem 2.11, the expected code length will not be smaller than the entropy, thus

$$\frac{1}{n}\sum_{x^n} p(x^n)K(x^n|n) = \frac{1}{n}\mathbb{E}_{p(x^n)}\left[K(x^n|n)\right] \geq \frac{1}{n}H(X_1, X_2, ..., X_n) = H(X)$$

Now consider the upper bound. Suppose $x_1, x_2, ..., x_n$ are i.i.d. drawn from Bernoulli(θ), the number of 1s in the string $x = x_1x_2...x_n$ is $k = \sum_{i=1}^{n} x_i$. By Eq (9),

$$K(x^n|n) \leq nH\left(\frac{1}{n}\sum_{i=1}^{n} x_i\right) + \log n + c$$

Thus,

$$\begin{aligned}\frac{1}{n}\sum_{x^n} p(x^n)K(x^n|n) &= \frac{1}{n}\mathbb{E}_{p(x^n)}\left[K(x^n|n)\right] \\ &\leq \mathbb{E}_{p(x^n)}\left[H\left(\frac{1}{n}\sum_{i=1}^{n} x_i\right)\right] + \frac{\log n}{n} + \frac{c}{n}\end{aligned}$$

Appendix 3: Kolmogrov Complexity

Since for $x \in [0,1]$, $H(x) = -x\log x - (1-x)\log(1-x)$ is a concave function of x, we have

$$\begin{aligned}\mathbb{E}_{p(x^n)}\left[H\left(\frac{1}{n}\sum_{i=1}^{n} x_i\right)\right] &\le H\left(\mathbb{E}_{p(x^n)}\left[\frac{1}{n}\sum_{i=1}^{n} x_i\right]\right) = H\left(\frac{1}{n}\sum_{i=1}^{n}\mathbb{E}_{p(x^n)}[x_i]\right)\\ &= H\left(\frac{1}{n}\sum_{i=1}^{n}\theta\right) = H(\theta) = H(X)\end{aligned}$$

Thus we have

$$\frac{1}{n}\sum_{x^n} p(x^n)K(x^n|n) \le H(X) + \frac{\log n}{n} + \frac{c}{n}$$

And by Squeeze Theorem,

$$\lim_{n\to\infty}\frac{1}{n}\sum_{x^n} p(x^n)K(x^n|n) = H(X)$$

□

Appendix 3: Kolmogrov Complexity

Consider a monkey randomly flipping coins to generate binary strings and send them to a universal computer as input programs. In most cases, the strings are not valid programs and the computer will output nothing. But, with certain probability, the monkey will input a valid program and the computer will output a meaningful string. We use the universal probability to describe the probability to make a meaningful output string by sending randomly generated binary program to a universal computer.

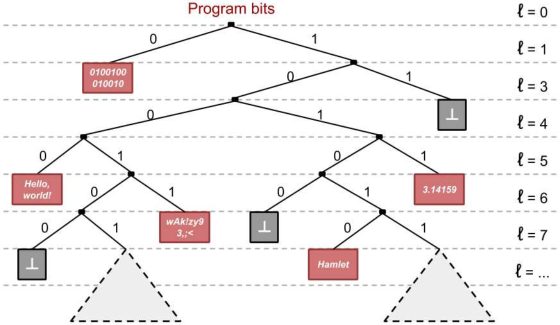
Definition A.3.11 (Universal Probability): The universal probability of a string x is

$$P_{\mathcal{U}}(x) = \sum_{p:\mathcal{U}(p)=x} 2^{-l(p)} = P\left(\mathcal{U}(p) = x\right)$$

which is the probability that a program randomly drawn as a sequence of fair coin flips p_1, p_2... will print out the string x.

Appendix 3: Kolmogrov Complexity

Intuition of the Universal Probability: Suppose all programs are prefixed. A valid program will halt with a meaningful string and a invalid program will halt with nothing. Consider the following tree structure[8], a program that halts with length $l(p)$ is of probability $2^{-l(p)}$.



[8] Figure from https://homepages.dcc.ufmg.br/~msalvim/courses/infotheory/L09_KolmogorovComplexity_UniversalProbability[still]

Appendix 3: Kolmogrov Complexity

Comparing with Theorem A.3.9, we know that the probability must be less or equal to 1,

$$P\left(\mathcal{U}(p)=x\right)=\sum_{p:\mathcal{U}(p)=x}2^{-l(p)}\leq\sum_{p:\mathcal{U}(p)\text{ halts}}2^{-l(p)}\leq 1$$

In all programs that halts, some of them will output x. If we randomly drawn these programs, a shorter program will have higher probability to be selected than a longer one.

Suppose that our world is some sort of data (string) generated by a law of physics (program). Scientists observe the phenomenons in the world to infer the law of physics that causes them. If there are two proposed laws of physics that explain the phenomenons equally well, the simpler law of physics (the shorter program) will be adopted since it has higher probability to be the true law. This is known as the Occam's Razor [9].

[9] I would like to share one interesting book that mentioned Occam's Razor and complexity in physics. *The quark and the jaguar* by Murray Gell-Mann.