Stochastic methods for image and signal analysis

Agnès Desolneux

Master MVA - ENS Cachan

Introduction : What is Pattern Theory ?

The term **“Pattern Theory”** was coined by Ulf Grenander to distinguish his approach to the analysis of patterned structures in the world from “Pattern Recognition”.

Here, we will follow the approach of D. Mumford and use the term “Pattern Theory” in a rather broad sense to include the statistical methods used in analyzing all “signals” generated by the world, whether they be images, sounds, written text, DNA or protein strings, spike trains in neurons, time series of prices or weather, or etc.

1. The manifesto of Pattern Theory

1. A wide variety of signals result from observing the world, all of which show patterns of many kinds, which are caused by objects, processes and laws present in the world but at least *partially hidden* from direct observation. These patterns can be used to *infer information* about these unobserved factors.
2. Observations are affected by many variables which are not conveniently modeled deterministically because they are too complex or too hard to observe and often belong to other categories of events which are irrelevant to the observations of interest. To make inferences in real time or with a model of reasonable size, we must model our observations *partly stochastically and partly deterministically*.
3. Accurate stochastic models which capture the patterns present in the signal, while respecting their natural structures, i.e. symmetries, independences of parts, marginals on key statistics, are needed. *These models should be learnt from the data and validated by sampling : inferences from them can be made using Bayes’ rule provided that samples from them resemble real signals.*

Example

Example a microphone or an ear responds to the pressure wave $p(t)$ transmitted by the air from a speaker's mouth.

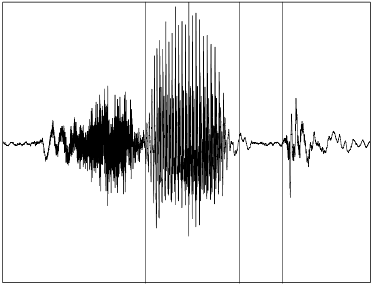


FIGURE: A second of the raw acoustic signal during the pronunciation of the word 'sheep'. Note the 4 phones : (i) white noise during the phoneme 'sh', (ii) a harmonic sound indicated by the periodic peaks (caused by the vocal chord openings) during for the vowel 'ee', (iii) silence while the mouth is closed during the first part of the stop consonant 'p', (iii) a burst when the mouth opens and the rest of 'p' is pronounced.

This function $p(t)$ with many obvious patterns. These patterns in p encode in a noisy, highly variable way the sequence of phones being pronounced and the word which these phones make up. We cannot *observe* the phones or the word directly, hence they are called *hidden variables*, but must infer them.

4. The various objects, processes and rules of the world produce patterns which can be described as precise *pure patterns* distorted and transformed by a limited family of *deformations*, similar across all modalities.
5. When all the stochastic factors affecting any given observation are suitably identified, they show a large amount of *conditional independence.*

2. Pattern Theory and Pattern Recognition

In what ways is Pattern Theory different from the better known field of Statistical Pattern Recognition ?

Traditionally, the focus of Statistical Pattern Recognition was the study of one or more data sets $\{\vec{x}_\alpha \in \mathbb{R}^k\}_{\alpha \in I}$ with the goal of (a) fitting (parametric and non-parametric) probability distributions to each data set, (b) finding optimal decision rules for classifying new data into the correct data set and (c) separating a single data set into clusters when it appears to be a mixture.

The essential issue is the **bias-variance** trade off : to model fully the complexities of the data source but not the accidental variations of the specific data set.

When Grenander first proposed Pattern Theory as a distinct enterprise, there were several very novel aspects of his approach :

- Firstly, he proposed that to describe the patterns in typical datasets, one should always look for appropriate *hidden variables*, in terms of which the patterns were more clearly described.
- Secondly, he proposed that the set of variables, observed and hidden, typically formed the vertices of a graph, as in Gibbs models, and that one must formulate *prior probability distributions for the hidden variables as well as models for the observed variables*.
- Thirdly, he proposed that *this graph itself might be random* and its variability must then be modeled.
- Fourthly, he proposed that one could list the different types of *deformations* which patterns were subject to, thus creating the basic classes of stochastic models that can be applied.
- Fifthly, he proposed that these models should be used for *pattern synthesis as well as analysis*.

As the subject evolved, Statistical Pattern Recognition merged with the area of neural nets and the first two ideas were absorbed into Statistical Pattern Recognition. So called ‘graphical models’ are now seen as the bread and butter of the field and discovering these hidden variables is a challenging new problem. And the use of prior models has become the mainstream approach in vision and expert systems as it has been in speech since the 60’s. However, the other aspects of Pattern Theory are still quite distinctive.

3. The basic types of patterns

Let us be more precise about the kinds of patterns and deformations referred to point 4 above. Real world signals show two very distinct types of patterns. We want to call these : i) *value patterns* and ii) *geometrical patterns*.

Signals, in general, are some sort of functions $f : X \to V$.
- In the case of value patterns, we mean that the features of this pattern are computed from the values of f or from some linear combinations of them (for example, power in some frequency band).
- In the case of geometric patterns, the function f can be thought of as producing geometrical patterns in its domain (for example, the set of its points of dis continuity).

The distinction affects which extra random variables you need to describe the pattern :
- For value patterns, we typically add coefficients in some expansion in order to describe the particular signal.
- For geometric patterns, we add certain points or subsets of the domain or features of such subsets. Traditional Statistical Pattern Recognition and the traditional theory of stationary processes deals only with the values of f, not the geometry of its domain.

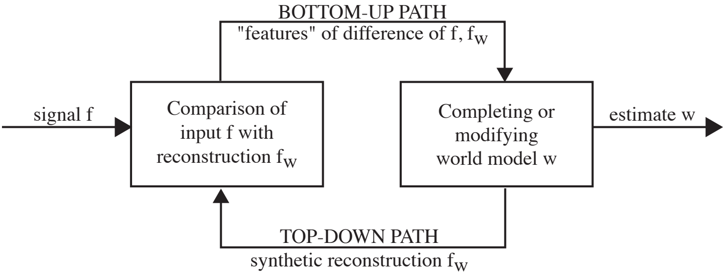

BOTTOM-UP PATH
"features" of difference of f, f_w
signal f
Comparison of input f with reconstruction f_w
Completing or modifying world model w
estimate w
TOP-DOWN PATH
synthetic reconstruction f_w

FIGURE: The fundamental architecture of Pattern Theory

First part : English Text and Markov Chains

In this chapter, we begin with a well-known type of signal and the simplest patterns in it. The ‘signals’ are just the strings of English characters that you find in any written English text.

Such signals are examples of *discrete one-dimensional* signals : they are made up of sequences $\{a_n\}$ whose elements lie in a finite set.

The general setting for our study will be signals made up of strings of letters taken from an arbitrary **finite alphabet** S. We will denote the size of the alphabet by $s = \#S$.
Examples of alphabets :

- Strings of English : $S = \{a, b, ..., z, space\}$, $s = 27$ (if we ignore caps, numbers and punctuation).
- Strings of phonetic transcriptions of speech. Using one version of the International Phonetic Alphabet for the phonemes used in English, we have 35 symbols.
- Machine code : S={ **0,1** }, $s = 2$, with which your hard disk is filled.
- Morse code : S={·,**–** ,*pause* }, $s = 3$.
- DNA : S={ **A,G,T,C** }, the 4 ‘codons’ of life which make up the genome string.
- Proteins : S = {Leu, Ile, Val, Met, Phe, Tyr, Gly, Ala, Lys, Arg, His, Glu, Asp, Gln, Asn, Ser, Thr, Cys, Trp, Pro}, the 20 amino acids.

We assume we have access to a very large set of very long signals in the alphabet S.

A substring of length n will be called an n-**gram**. Let $\sigma = (a_1 ... a_n)$, $a_i \in S$, be an n-gram. Let Ω_n be the set of all such strings : there are s^n of them. We define

$$P_n(\sigma) = \text{ the frequency of } \sigma \text{ in messages.}$$

Then $P_n(\sigma) \geq 0$ and $\sum_\sigma P_n(\sigma) = 1$, i.e. P_n is a probability distribution on Ω_n.

As n gets larger, this table of frequencies captures more and more of the patterns in these signals.

What we do assume is that the statistics of long signals are *stationary*, i.e. if we take any piece σ of length n out of a signal of length N, the frequency of occurrences of the resulting n-grams does not depend on where it is found in the long signal :

Definition

Given a finite alphabet S, a **language on** S *is a set of probability distributions P_n on the set of strings Ω_n of length n for all $n \geq 1$ such that for all pairs of integers $n_1 < n_2$ and all i with $1 \leq i \leq n_2 - n_1 + 1$, P_{n_1} is the marginal of P_{n_2} under the map $\Omega_{n_2} \to \Omega_{n_1}$ given by $\{a_1 \cdots a_{n_2}\} \to \{a_i \cdots a_{i+n_1-1}\}$.*

We can construct a full language using only the n-gram statistics for a small fixed n as follows.

Notice first that the frequency table P_n on n-grams gives, as marginals, frequency tables P_k on k-grams, $k < n$, hence also **conditional probabilities** :

$$P_n(a_n|a_1...a_{n-1}) = \frac{P_n(a_1...a_n)}{P_{n-1}(a_1...a_{n-1})} = \frac{P_n(a_1...a_n)}{\sum_b P_n(a_1...a_{n-1}b)}.$$

The realization that these *conditional probabilities* were very informative in describing the nature of the language goes back to A.A.Markov. In 1913, he published a paper analyzing the poem *Eugene Onyegin* by Pushkin and comparing it with other texts. Having no computers, he reduced the alphabet to two symbols : vowel v and consonant c, and he computed by hand the probabilities in this long poem :

$$P_1(v) = 0.432, P_1(c) = 0.568 \text{ and}$$

$$P_2(v|c) = 0.663, P_2(c|c) = .337, P_2(v|v) = 0.128, P_2(c|v) = 0.872.$$

Moreover, he showed that these numbers varied substantially in other works, hence they captured some aspect of the author's style.

Using these conditional probabilities, we can define a simplified probability model on strings of any length $N > n$ by a sort of sliding window :

$$P_N^{(n)}(a_1 ... a_N) = \begin{cases} P_n(a_1 \cdots a_n) \cdot \prod_{i=n+1}^{N} P_n(a_i | a_{i-n+1} ... a_{i-1}) \\ \qquad \text{if } P_{n-1} \text{ of all length } n-1 \text{ substrings of } a_1 ... a_N \text{ is non-zero} \\ 0 \text{ otherwise} \end{cases} \tag{1}$$

$P^{(n)}$ is called the n**-gram approximation to the full language**.

A good way to see what patterns the stochastic model given by the frequency tables P_n capture is **to sample** from its extension $P_N^{(n)}$ to long strings.

The sampling procedure is simple : we choose $(a_1 ... a_n)$ randomly from the distribution P_n, then a_{n+1} randomly from the distribution $P_n(b | a_2 ... a_n)$, then a_{n+2} from the distribution $P_n(b | a_3 ... a_{n+1})$, etc.

This ‘analysis by synthesis’ was first done by Shannon (1948) :

Samples from the n-gram distributions

- Random characters :
  XFOML RXKHRJFFJUJ ZLPWCFWKCYJ FFJEYVKCQSGXYD
  QPAAMKBZAACIBZLHJQD
- Sample from $P^{(1)}$
  OCRO HLI RGWR NMIELWIS EU LL NBBESEBYA TH EEI ALHENHTTPA OO
  BTTV
- Sample from $P^{(2)}$
  ON IE ANTSOUTINYS ARE T INCTORE ST BE S DEAMY ACHIN D
  ILONASIVE TUCOOWE FUSO
  TIZIN ANDY TOBE SEACE CTISBE
- Sample from $P^{(3)}$
  IN NO IST LAY WHEY CRATICT FROURE BERS GROCID PONDENOME OF
  DEMONSTURES OF THE REPTAGIN IS REGOACTIONA OF CRE
- Sample from $P^{(4)}$
  THE GENERATED JOB PROVIDUAL BETTER TRAND THE DISPLAYED CODE
  ABOVERY UPONDULTS WELL THE CODERST IN THESTICAL IT TO HOCK
  BOTHE

We see how the samples get closer and closer in some intuitive sense to reasonable English text. But we really would like to measure quantitatively how close the n-gram models $P^{(n)}$ are to the “true” probability model of length N strings.

Basics : Entropy and information

Entropy is a positive real number associated to an arbitrary finite probability space. It measures the information carried by a random variable, or equivalently by a probability distribution.

Definition

Let $\Omega = \{a_1, ..., a_N\}$ *and let* $P : \Omega \to \mathbb{R}_+$ *be a function such that* $\sum P(a_i) = 1$*, making* Ω *into a finite probability space. Or equivalently, let* $\mathcal{X}$ *be a random variable with values* $\{a_1, ..., a_N\}$ *and* $P(a_i) = \mathbb{P}(\mathcal{X} = a_i)$*. Then the* **entropy** *of* P *or of* $\mathcal{X}$ *is defined by*

$$H(P) = H(\mathcal{X}) = -\sum_{i=1}^{N} P(a_i) \log_2 P(a_i) = \mathbb{E}(\log_2(1/P)).$$

Notice that $H(P) \geq 0$ because $0 \leq P(a_i) \leq 1$.

By convention, if $P(a_i) = 0$, then we set $P(a_i) \log_2 P(a_i) = 0$.

Entropy and coding

As was shown by Shannon, $H(\mathcal{X})$ can be interpreted in a quite precise way as measuring the average number of *bits* of information contained in a sample of the random variable $\mathcal{X}$.

Suppose we decide to **code** the fact that we have drawn the sample a_i by the fixed bit string (i.e. a sequence of 0's and 1's) σ_i. A sequence of samples $a_{i_1}, \cdots, a_{i_M}$ is then encoded by the concatenation of the bit strings $\sigma_{i_1} \cdots \sigma_{i_M}$.

In order to be able to decode uniquely (without the extra overhead of marking the ends of codewords), we require that for all $j \neq i$, σ_j is not a prefix of σ_i. Such a code is called a **prefix code**.

Shannon's fundamental result is :

Theorem

1. *For all prefix codes σ_i, the expected length of the code for one sample $\sum_i P(a_i)|\sigma_i|$ satisfies :*

$$\sum_i P(a_i)|\sigma_i| \geq \sum_i P(a_i)\log_2(1/P(a_i)) = H(P).$$

2. *We can find a prefix code such that*

$$|\sigma_i| = \lceil \log_2(1/P(a_i)) \rceil,$$

   where for x real, $\lceil x \rceil$ denotes the integer such that $x \leq \lceil x \rceil < x+1$. Hence, with this coding, the expected length $\sum_i P(a_i)|\sigma_i|$ is less than $H(P)+1$.

3. *If we use* block coding, *i.e. we encode sequences of independent messages of the same type by a single code word, then the expected coding length per message tends to $H(P)$.*

First variant : Conditional Entropy

Definition

Let $\mathcal{X}$ be a random variable with values in $\Omega_1 = \{x_1, ..., x_N\}$ and probability measure $P(x_i) = \mathbb{P}(\mathcal{X} = x_i)$. Let $\Omega_2 = \{y_1, ..., y_M\}$ be a finite set and f a function, $f : \Omega_1 \to \Omega_2$. We define the induced probability measure P on the random variable $\mathcal{Y} = f(\mathcal{X})$ by $P(y) = \sum_{f(x)=y} P(x)$ and define the conditional probability measures $P(x|y)$ to be $P(x)/P(y)$ when $y = f(x)$. The ***conditional entropy*** *is then defined by*

$$H(\mathcal{X}|\mathcal{Y}) = -\sum_{x,y=f(x)} P(x)\log_2 P(x|y).$$

If $\mathcal{X}$ and $\mathcal{Y}$ are two random variables for which we have a joint probability distribution, i.e. a probability measure on the space $\Omega_1 \times \Omega_2$, it is customary to extend the idea of conditional entropy, defining $H(\mathcal{X}|\mathcal{Y})$ to be $H((\mathcal{X},\mathcal{Y})|\mathcal{Y})$.

Note that $H(\mathcal{X}|\mathcal{Y}) \geq 0$ and that (taking the general case) :

$$H(\mathcal{X}|\mathcal{Y}) \quad = \quad H(\mathcal{X},\mathcal{Y}) - H(\mathcal{Y}) \quad \text{or } = H(\mathcal{X}) - H(\mathcal{Y}) \text{ when } \mathcal{Y} = f(\mathcal{X}).$$

Rk : When $\mathcal{X}$ and $\mathcal{Y}$ are independent, then $H(\mathcal{X},\mathcal{Y}) = H(\mathcal{X}) + H(\mathcal{Y})$ and thus $H(\mathcal{X}|\mathcal{Y})$ is simply $H(\mathcal{X})$.

Second variant : the Kullback-Leibler distance

Definition

Let Ω be a set, not necessarily finite, with two probability measures P and Q on it. Then the **Kullback-Leibler distance** *from P to Q, also called the divergence of Q from P, is defined by*

$$\begin{aligned} D(P||Q) &= \mathbb{E}_P\left(\log_2 \frac{P}{Q}\right) \\ &= \sum_{a\in\Omega} P(a)\log_2 \frac{P(a)}{Q(a)} \quad \textit{if } \Omega \textit{ is finite.} \end{aligned}$$

Rk : $D(P||Q)$ is *not symmetric* in P and Q, but we do have the following which justifies the use of the term of the term “distance”.

Theorem

$D(P||Q) \geq 0$ and $D(P||Q) = 0$ if and only if $P = Q$.

This gives us many corollaries, such as :

Proposition

If Ω is finite with N elements in it, and U denotes the uniform probability distribution on Ω (for each a, $U(a) = 1/N$), then if P is any probability distribution on Ω,

$$D(P||U) = \log_2 N - H(P).$$

Hence, $0 \leq H(P) \leq \log_2 N$ and $H(P) = \log_2 N$ if and only if $P = U$.

Link between, entropy and conditional entropy via the KL-distance :

Proposition

For any random variables $\mathcal{X}$ and $\mathcal{Y}$, $H(\mathcal{X}|\mathcal{Y}) \leq H(\mathcal{X})$. In fact, if P is the joint distribution of $\mathcal{X}$ and $\mathcal{Y}$ and the product $P_1 P_2$ is the modified probability distribution on $\mathcal{X}$ and $\mathcal{Y}$ making them independent (where P_1 and P_2 are the marginals on $\mathcal{X}$ and $\mathcal{Y}$ given by $P_1(x) = \sum_y P(x,y)$ and $P_2(y) = \sum_x P(x,y)$), then

$$H(\mathcal{X}) = H(\mathcal{X}|\mathcal{Y}) + D(P||P_1 P_2).$$

More generally, the same reasoning proves that for any three random variables $\mathcal{X}, \mathcal{Y}$ and $\mathcal{Z}$:

$$H(\mathcal{X}|\mathcal{Y},\mathcal{Z}) \leq H(\mathcal{X}|\mathcal{Z}), \tag{2}$$

which can be interpreted to mean that adding extra information can only decrease the number of bits needed to describe $\mathcal{X}$.

Proof of Shannon's optimal coding theorem

- We first need the following lemma :

Lemma : *For any prefix code,*

$$\sum_i \frac{1}{2^{|\sigma_i|}} \leq 1.$$

- We can now prove the first part of the theorem. Let α be the real number defined by

$$\frac{1}{\alpha} = \sum_i \frac{1}{2^{|\sigma_i|}} \leq 1.$$

We then define the probability distribution Q by $Q(a_i) = \alpha/2^{|\sigma_i|}$. The Kullback-Leibler distance from P to Q is positive and this proves the first part of the theorem.

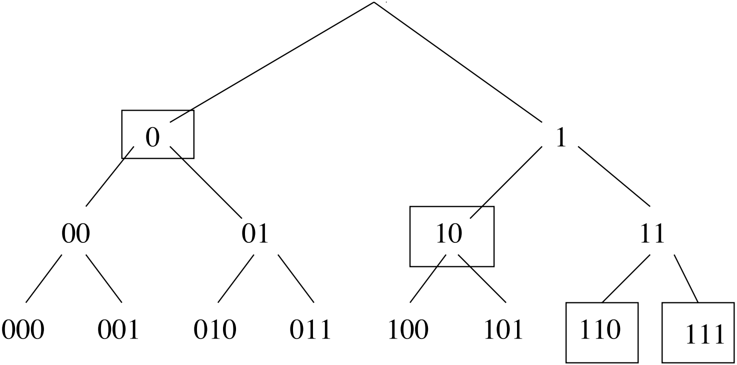
- For the second part of the theorem, let $k_i = \lceil \log_2(1/P(a_i)) \rceil$. Then $k_i \geq \log_2(1/P(a_i))$, so that

$$\sum_i \frac{1}{2^{k_i}} \leq 1.$$

Now, we can construct a prefix code such that for all i, $|\sigma_i| = k_i$.

- For the last part of the theorem, just note that if you encode with one codebook a string of M independent messages $\{a_{i_1}, \cdots, a_{i_M}\}$, then the previous argument shows that the expected length is at most one more than the entropy of the space of M-tuples, which is M times the entropy $H(P)$. Thus the expected length *per message* is at most $H(P) + 1/M$.

Let's give a very simple example : suppose $\Omega = \{a, b, c, d\}$ and $P(a) = 0.5, P(b) = 0.25, P(c) = 0.125$ and $P(d) = 0.125$. We could use the four codes 00, 01, 10, 11 for a, b, c, d in which case 2 would always be the coding length. But we do better if we code a by a 1-bit code, e.g. 0, b by 2 bits, e.g. 10, and c and d by 3 bits, e.g. 110 and 111. The resulting coding scheme has expected coding length 1.75 which saves a quarter of a bit. And it is optimal since it equals $H(P)$.



Rk : The proof of Shannon's theorem also gives us an interpretation of $D(P||Q)$ in terms of information. Suppose $\mathcal{X}$ is a random variable whose true probability distribution is P. Then if we code values of $\mathcal{X}$ using the suboptimal code associated to Q, $D(P||Q)$ is the number of extra bits this entails.

Measuring the n-gram approximation with entropy

As we have seen Ω_N carries not only the 'true' model P_N but the approximations given by the prolongations $P_N^{(1)}, P_N^{(2)}, \cdots, P_N^{(N-1)}$ defined by formula (1). Let's see if we can use the distances $D(P_N || P_N^{(n)})$ to see how good these approximations are.

We use the notation $a_1 \cdots a_n$ to describe *random strings* $\mathcal{S}_n$ of length n.
Let $f_n, g_n : \Omega_n \to \Omega_{n-1}$, be the "initial substring" and "final substring" maps, that is $f_n(\sigma_{n-1} a) = \sigma_{n-1}$ and $g_n(a\sigma_{n-1}) = \sigma_{n-1}$.

The key numbers for describing strings information-theoretically are the conditional entropies F_n :

$$F_n = H_{P_n}(a_n | a_1 \cdots a_{n-1}) = -\sum_{\sigma_n} P_n(\sigma_n) \log_2 P_n(\sigma_n | f_n(\sigma_n)) = H(P_n) - H(P_{n-1}).$$

These numbers measure the predictability of the n^{th} character given the $n-1$ preceding characters, that is the average number of bits needed to encode the n^{th} character in a string of length n given the $n-1$ previous ones.

By formula (2) with $\mathcal{X} = a_n, \mathcal{Y} = a_1$ and $\mathcal{Z} = a_2 \cdots a_{n-1}$, we find

$$H(a_n | a_1 \cdots a_{n-1}) \leq H(a_n | a_2 \cdots a_{n-1}), \text{ that is } F_n \leq F_{n-1}.$$

We let F_1 be simply $H(\mathcal{S}_1) = H(P_1)$.
The entropy of strings of length N which represents the average number of bits needed to encode the whole string of length N :

$$H(P_N) = F_N + F_{N-1} + \cdots + F_1.$$

If, instead of the true probability distribution P_N, we use the approximation $P_N^{(n)}$, then for all $k \geq n$, the conditional entropy of σ_k with respect to $f_k(\sigma_k)$ simplifies :

$$\begin{aligned} H_{P_k^{(n)}}(\mathcal{S}_k | f_k(\mathcal{S}_k)) &= -\sum P_k^{(n)}(\sigma_k) \log_2 P_k^{(n)}(\sigma_k | f_k(\sigma_k)) \\ &= -\sum P_k^{(n)}(\sigma_k) \log_2 P_n(a_k | a_{k-n+1} \cdots a_{k-1}) \\ &= F_n \end{aligned}$$

This gives us by summing over k :

$$H(P_N^{(n)}) = (N-n+1)F_n + F_{n-1} + \cdots + F_1.$$

Finally, we can also compute the Kullback-Leibler distances in terms of the F's and find :

$$\begin{aligned} D(P_N || P_N^{(n)}) &= \sum_{\sigma_N} P_N(\sigma_N) \log_2 \left(\frac{P_N(\sigma_N)}{P_N^{(n)}(\sigma_N)} \right) \\ &= \ldots \\ &= H(P_N^{(n)}) - H(P_N) = \sum_{k=n+1}^{N} (F_n - F_k). \end{aligned}$$

Since the F_n are decreasing, they have a limit as n goes to infinity, which we denote by F. Combining the formulas for $H(P_N)$ and $H(P_N^{(n)})$ in terms of the F's, it follows that :

$$F \leq \frac{H(P_N)}{N} \leq \frac{H(P_N^{(n)})}{N} \leq F_n + \frac{n}{N}(F_1 - F_n).$$

Letting first N go to infinity and then n, it follows that :

$$\lim_{n \to \infty} \frac{H(P_N)}{N} = F.$$

Thus F is the key number which describes the information content per symbol of long signals, called the "entropy of the source". Finally, we get :

$$\frac{D(P_N || P_N^{(n)})}{N} \leq F_n - F,$$

which shows convergence in the Kullback-Leibler sense of the n^{th} order models to the true model.

For some concrete results, we refer to the work of Shannon in "Prediction and Entropy of Printed English" (1951). He worked on strings of English, with alphabet $\{a, b, ..., z, space\}$. From standard tables of frequencies he then computed the conditional entropies :

$$F_0 = \log_2(27) \simeq 4.75; \quad F_1 = 4.03; \quad F_2 = 3.32; \quad F_3 = 3.1; \quad F_4 = 2.8.$$

Markov chains and the n-gram models

We saw that strings of letters can be modeled as sequences of n-grams with some transition probability between the different n-grams. The right mathematical framework for such sequences is the theory of Markov chains.

Definition

Let Ω be a finite set of states. A sequence $\{\mathcal{X}_0, \mathcal{X}_1, \mathcal{X}_2, \cdots\}$ of random variables taking values in Ω is said to be a **Markov chain** *if it satisfies the Markov condition :*

$$\mathbb{P}(\mathcal{X}_n = i_n | \mathcal{X}_0 = i_0, \mathcal{X}_1 = i_1, \ldots, \mathcal{X}_{n-1} = i_{n-1}) = \mathbb{P}(\mathcal{X}_n = i_n | \mathcal{X}_{n-1} = i_{n-1}),$$

for all $n \geq 1$ and all $i_0, i_1, \ldots i_n$ in Ω.
Moreover, the chain is said **homogeneous** *if for all $n \geq 1$ and all i, j in Ω,*

$$\mathbb{P}(\mathcal{X}_n = j | \mathcal{X}_{n-1} = i) = \mathbb{P}(\mathcal{X}_1 = j | \mathcal{X}_0 = i).$$

Markov chains will always be assumed homogeneous unless otherwise specified. In this case, the transition matrix Q of the chain is defined as the $|\Omega| \times |\Omega|$ matrix of all transition probabilities $Q(i,j) = q_{i \to j} = \mathbb{P}(\mathcal{X}_1 = j | \mathcal{X}_0 = i)$.

Then, the matrix Q^n gives the law of $\mathcal{X}_n$ for the chain starting at $\mathcal{X}_0 = i$:

$$\mathbb{P}(\mathcal{X}_n = j | \mathcal{X}_0 = i) = Q^n(i,j).$$

This can be shown by induction on n.

Definition

We say that the Markov chain is **irreducible or primitive** *if for all i, j in Ω, there exists $n \geq 0$ such that $Q^n(i,j) > 0$.*

It means that if the chain starts at $\mathcal{X}_0 = i$, then for any j there exists an integer n such that $\mathbb{P}(\mathcal{X}_n = j | \mathcal{X}_0 = i) > 0$. In other words, all states can be “connected” through the chain.

But it needn’t necessarily happen that there is one n such that $Q^n(i,j) > 0$ for all j. For instance there can be two states a and b such that : $Q(a,a) = Q(b,b) = 0$ and $Q(a,b) = Q(b,a) = 1$. Then the chain just goes back and forth between a and b, and e.g. $Q^n(a,a) = 0$ for all odd n, and $Q^n(a,b) = 0$ for all even n.

This can be measured by defining the **period** of a state i : it is the greatest common divisor of all the n such that $Q^n(i,i) > 0$. Then, it is an easy theorem that for any irreducible Markov chain, all states have the same period d and there is a decomposition $\Omega = S_1 \cup \ldots \cup S_d$ such that the chain Q takes S_k to S_{k+1} with probability 1.

Definition

We say that a Markov chain is **aperiodic** *if for all i in Ω the greatest common divisor of the $n \geq 1$ such that $Q^n(i,i) > 0$ is 1 (all states have period $d = 1$).*

If the Markov chain is irreducible and aperiodic, since Ω is finite, this implies that there exists n_0 such that for all $n \geq n_0$ and for all i,j in Ω, $Q^n(i,j) > 0$.

Definition

A probability distribution Π on Ω is an **equilibrium (or steady-state) probability distribution** *of Q iff*

$$\forall j \in \Omega, \quad \sum_i \Pi(i) Q(i,j) = \Pi(j).$$

We then have the following theorem (recall that Ω is finite) :

Theorem

If Q is irreducible, then there exists a unique equilibrium probability distribution Π for Q. If moreover Q is aperiodic, then

$$\forall i,j \in \Omega, \quad Q^n(i,j) \underset{n \to +\infty}{\longrightarrow} \Pi(j).$$

Notice that in the above result, the limit is independent of the starting state i.

Markov property for the n-gram models

Proposition

Let Ω_N be the space of strings $a_1 \dots a_N$ of length N, the a_i's being taken from a finite alphabet S. Consider a probability distribution $P : \Omega_N \to \mathbb{R}_+$. Fix an integer $n \geq 1$. The following conditions are equivalent :

1. **Conditional factorization** *: there exist P_0 and P_1 such that P has the form*

$$P(a_1 \dots a_N) = P_0(a_1 \dots a_{n-1}) \prod_{k=0}^{N-n} P_1(a_{k+n} | a_{k+1} \dots a_{k+n-1}).$$

2. **Exponential form** *:*

$$P(a_1 \dots a_N) = \frac{1}{\mathcal{Z}} \exp\left(- \sum_{\sigma \in \Omega_n} \lambda_\sigma \cdot \#\mathrm{occ}(\sigma, a_1 \dots a_N) \right),$$

   for suitable λ_σ's, where Ω_n is the set of strings of length n and $\#\mathrm{occ}(\sigma, a_1 \dots a_N)$ denotes the number of occurrences of the string σ in $a_1 \dots a_N$.

3. **Markov property** *: for all $I = (k+1, \dots, k+n-1)$ of length $n-1$, let $a(I) = a_{k+1} \cdots a_{k+n-1}$, then*

$$P(a_1 \dots a_N | a(I)) = P_1(a_1 \dots a_k | a(I)) \cdot P_2(a_{k+n} \dots a_N | a(I)),$$

   which means that $a($before $I)$ and $a($after $I)$ are conditionally independent given $a(I)$.

We now look at the case where the probability P on Ω_N has the above Markov property with substrings of length n.
Let Ω_n^* be the set of strings σ of length n with $P(\sigma) > 0$. Using the Markov property, we can convert the generation of these longer and longer strings into a Markov chain on Ω_n^* in the following way. Given a string $(a_1 ... a_N)$, for $1 \leq k \leq N-n+1$ let $\sigma_k = (a_k ... a_{k+n-1})$ be the substring of length n starting at the k^{th} letter. Then σ_1,..., σ_{N-n+1} is a Markov chain with initial distribution $P_0(\sigma_1)$ and transition probabilities :

$$P(\sigma \to \tau) = P(a_1 ... a_n \to b_1 ... b_n) = \begin{cases} P(b_n | a_1 ... a_n) \;\; \text{if} \;\; b_1 = a_2, b_2 = a_3 \ldots \text{ and } b_{n-1} = a_n, \\ 0 \text{ otherwise .} \end{cases}$$

We now can use the standard theorems on Markov chains to deduce “regularities” or “ergodic conditions” on the n-gram model. In particular, an n-gram string model is irreducible and aperiodic if there exists an integer m such that for all strings σ, τ of length n there exists a string $(\sigma\rho\tau)$ of positive probability with length $|\rho| = m$. Then, for n-gram strings from such a model, there is a unique equilibrium distribution. It is given by

$$\Pi(\tau) = \lim_{m \to \infty} P(a_m ... a_{m+n-1} = \tau | a_1 ... a_n).$$

Notice that it is independent of $(a_1 ... a_n)$.

Given an homogeneous Markov chain $\{\mathcal{X}_0, \mathcal{X}_1, \ldots\}$, the **averaged entropy of the first** N **variables** can be defined by the quantity $\frac{1}{N}H(\mathcal{X}_1, \mathcal{X}_2, \ldots, \mathcal{X}_N)$. In the case of strings, this represents the entropy per symbol of text. Using the chain rule for conditional entropy, we have

$$H(\mathcal{X}_0, \mathcal{X}_1, \ldots, \mathcal{X}_N) = H(\mathcal{X}_0) + \sum_{k=1}^{N} H(\mathcal{X}_k|\mathcal{X}_0, \ldots \mathcal{X}_{k-1}) = H(\mathcal{X}_0) + \sum_{k=1}^{N} H(\mathcal{X}_k|\mathcal{X}_{k-1}).$$

If we assume that the Markov chain is irreducible and aperiodic, and if we denote by Q its transition matrix and by Π its equilibrium probability distribution then

$$\begin{aligned} H(\mathcal{X}_k|\mathcal{X}_{k-1}) &= -\sum_{i,j} P(\mathcal{X}_{k-1} = i, \mathcal{X}_k = j) \log_2 P(\mathcal{X}_k = j|\mathcal{X}_{k-1} = i) \\ &= -\sum_{i,j} Q(i,j) P(\mathcal{X}_{k-1} = i) \log_2 Q(i,j) \\ &\underset{k\to\infty}{\longrightarrow} -\sum_{i,j} Q(i,j)\Pi(i) \log_2 Q(i,j). \end{aligned}$$

Thanks to Cesaro's theorem, this implies that the averaged entropy per symbol converges as N goes to infinity and that its limit is :

$$\lim_{N\to\infty} \frac{H(\mathcal{X}_1, \mathcal{X}_2, \ldots, \mathcal{X}_N)}{N} = -\sum_{i,j} Q(i,j)\Pi(i) \log_2 Q(i,j).$$

This limit, also called the **entropy rate** was introduced by Shannon. We will denote it by $\overline{H}(Q)$.

Let us go back to our n-gram model. If we use the n-gram distribution P_n, as we have already explained it above, we can convert an infinite sequence of letters $a_1, a_2 \dots a_N \dots$ into a Markov chain $\{\sigma_1, \sigma_2, \dots\}$, by setting $\sigma_k = (a_k ... a_{k+n-1})$, and using the transition matrix Q_n given by

$$Q_n(\sigma, \tau) = Q_n(a_1 ... a_n \to b_1 ... b_n) = \begin{cases} \frac{P_n(a_2 \dots a_n b_n)}{P_{n-1}(a_2 \dots a_n)} \text{ if } b_1 = a_2, b_2 = a_3 \dots \text{ and } b_{n-1} = a_n, \\ 0 \text{ otherwise .} \end{cases}$$

Then, it is easy to check that an equilibrium probability distribution of this Markov chain is P_n. Indeed, let $(b_1 ... b_n)$ be a string of length n, we then have

$$\sum_{a_1, \dots, a_n} P_n(a_1 \dots a_n) Q_n(a_1 ... a_n \to b_1 ... b_n) = \dots = P_n(b_1 ... b_n).$$

If the Markov chain is irreducible and aperiodic, then P_n is the unique equilibrium probability distribution. And we find that the entropy rate of this chain is equal to the conditional entropy F_n introduced in the previous section :

$$\begin{aligned} \overline{H}(Q_n) &= -\sum_{a_1 \dots a_n a_{n+1}} P_n(a_1 ... a_n) \frac{P_n(a_2 ... a_n a_{n+1})}{P_{n-1}(a_2 ... a_n)} \log_2 \frac{P_n(a_2 ... a_n a_{n+1})}{P_{n-1}(a_2 ... a_n)} \\ &= H(P_n) - H(P_{n-1}) = F_n. \end{aligned}$$

Markov chains and Mutual information

We can translate the convergence of Markov chains in the language of information theory introducing a new definition – mutual information.

Definition

Given two random variables $\mathcal{X}$ and $\mathcal{Y}$ where the possible values of $\mathcal{X}$ (respectively $\mathcal{Y}$) are $x_1,\dots,x_n$ (respectively $y_1,\dots,y_m$), let $P(x,y)$ be a joint distribution on the two variables $\mathcal{X}$ and $\mathcal{Y}$. Let

$$P_1(x) = \sum_y P(x,y) \text{ and } P_2(y) = \sum_x P(x,y)$$

be its marginals and

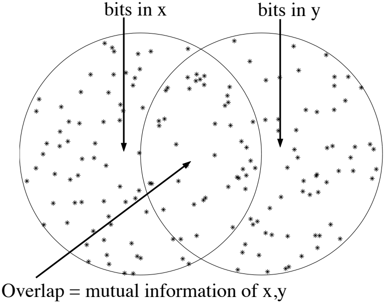
$$Q(x,y) = P_1(x)P_2(y)$$

be the distribution if $\mathcal{X}$ and $\mathcal{Y}$ were independent. Then the **mutual information** *of $(\mathcal{X},\mathcal{Y})$ is*

$$\begin{aligned} MI(\mathcal{X},\mathcal{Y}) &= D(P||Q) \\ &= \sum_{x,y} P(x,y)\log_2 \frac{P(x,y)}{P_1(x)P_2(y)} \\ &= -\sum_{x,y} P(x,y)\log_2 P_1(x) - \sum_{x,y} P(x,y)\log_2 P_2(y) + \sum_{x,y} P(x,y)\log_2 P(x,y) \\ &= H(\mathcal{X}) + H(\mathcal{Y}) - H(\mathcal{X},\mathcal{Y}) = H(\mathcal{X}) - H(\mathcal{X}|\mathcal{Y}). \end{aligned}$$

In particular, note that $0 \le MI(\mathcal{X},\mathcal{Y}) \le \min(H(\mathcal{X}),H(\mathcal{Y}))$ and that $MI(\mathcal{X},\mathcal{Y}) = 0$ when $\mathcal{X}$ and $\mathcal{Y}$ are independent.

A convenient way to ‘visualize’ the meaning of mutual information is by the Venn diagram.
The circle labeled X is to have area $H(\mathcal{X})$ and represents the bits in a sample of $\mathcal{X}$. Likewise the circle labeled Y has area $H(\mathcal{Y})$ and represents the bits in a sample of $\mathcal{Y}$. A joint sample of $\mathcal{X}, \mathcal{Y}$ can be coded by giving $\mathcal{X}$ and $\mathcal{Y}$ as though they had nothing to do with other, i.e. were independent : this would need $H(\mathcal{X}) + H(\mathcal{Y})$ bits. But knowing the value of $\mathcal{X}$ usually tells us something about the value of $\mathcal{Y}$, so the number of extra bits $H(\mathcal{Y}|\mathcal{X})$ is less than $H(\mathcal{Y})$. The diagram shows this by having the circles for $\mathcal{X}$ and $\mathcal{Y}$ overlap in a region of area $MI(\mathcal{X}, \mathcal{Y})$, the number of bits of mutual information. Thus the union of the two circles has area $H(\mathcal{X}, \mathcal{Y})$ are represents correctly the number of bits required to describe a sample of $\mathcal{X}, \mathcal{Y}$ together.



Nice as this Venn diagram way of thinking about information is, it **fails** when you have three random variables $\mathcal{X}, \mathcal{Y}, \mathcal{Z}$! Following the usual inclusion/exclusion idea, one is tempted to define a 3-way mutual information by requiring :

$$H(\mathcal{X}, \mathcal{Y}, \mathcal{Z}) = H(\mathcal{X}) + H(\mathcal{Y}) + H(\mathcal{Z}) - MI(\mathcal{X}, \mathcal{Y}) - MI(\mathcal{X}, \mathcal{Z}) - MI(\mathcal{Y}, \mathcal{Z}) + MI(\mathcal{X}, \mathcal{Y}, \mathcal{Z}).$$

This should represent area of the triple overlap or the number of bits in common to all three variables.
But consider the following simple example : let $\mathcal{X}, \mathcal{Y}$ and $\mathcal{Z}$ be three binary variables with $\mathcal{X}$ and $\mathcal{Y}$ independent and $\mathcal{Z}$ being 1 if $\mathcal{X} = \mathcal{Y}$, 0 if $\mathcal{X} \neq \mathcal{Y}$. Then not only are $\mathcal{X}$ and $\mathcal{Y}$ independent but $\mathcal{X}$ and $\mathcal{Z}$ are independent and $\mathcal{Y}$ and $\mathcal{Z}$ are independent. But clearly, knowing any two of $\mathcal{X}, \mathcal{Y}, \mathcal{Z}$ determines the third. Thus
$H(\mathcal{X}) = H(\mathcal{Y}) = H(\mathcal{Z}) = 1$ and $MI(\mathcal{X}, \mathcal{Y}) = MI(\mathcal{X}, \mathcal{Z}) = MI(\mathcal{Y}, \mathcal{Z}) = 0$ but
$H(\mathcal{X}, \mathcal{Y}, \mathcal{Z}) = 2$. Thus the above definition makes $MI(\mathcal{X}, \mathcal{Y}, \mathcal{Z}) = -1$ so it cannot represent the area of the triple overlap !

Mutual information is one of many ways to measure the convergence of Markov chains :

Proposition

If $\{\mathcal{X}_n\}$ is an irreducible and aperiodic Markov chain on a finite space, then for all $\varepsilon > 0$, there exists an integer m such that

$$\forall k \quad MI(\mathcal{X}_k, \mathcal{X}_{k+m}) < \varepsilon.$$

Chains with this property are called “information regular”.

Proof : We fix k and let n be a positive integer, then

$$\begin{aligned}
MI(\mathcal{X}_k, \mathcal{X}_{k+n}) &= \sum_{x,y} P(\mathcal{X}_k = x, \mathcal{X}_{k+n} = y) \log_2 \frac{P(\mathcal{X}_{k+n} = y | \mathcal{X}_k = x)}{P(\mathcal{X}_{k+n} = y)} \\
&= \sum_{x} P(\mathcal{X}_k = x) \sum_{y} P(\mathcal{X}_{k+n} = y | \mathcal{X}_k = x) \log_2 (P(\mathcal{X}_{k+n} = y | \mathcal{X}_k = x)) \\
&\quad - \sum_{y} (P(\mathcal{X}_{k+n} = y) \log_2 (P(\mathcal{X}_{k+n} = y)) \\
&= H(\mathcal{X}_{k+n}) - \sum_{x} P(\mathcal{X}_k = x) H(\mathcal{X}_{k+n} | \mathcal{X}_k).
\end{aligned}$$

Let Π denote the equilibrium probability distribution of the Markov chain. Then

$$\forall x, y \quad \lim_{n \to \infty} P(\mathcal{X}_{k+n} = y | \mathcal{X}_k = x) = \Pi(y),$$

and

$$\forall y \quad \lim_{n \to \infty} P(\mathcal{X}_{k+n} = y) = \Pi(y).$$

Therefore

$$\lim_{n \to \infty} H(\mathcal{X}_{k+n} | \mathcal{X}_k) = H(\Pi), \text{ and } \lim_{n \to \infty} H(\mathcal{X}_{k+n}) = H(\Pi)$$

hence $\lim_{n \to \infty} MI(\mathcal{X}_k, \mathcal{X}_{k+n}) = 0$, and this, uniformly in k.

Words

We have set up the analysis of English text, following Shannon, by including ‘space’ as a character, so words are obvious : they are the units between the spaces. But this is really just a convenience for readers. In spoken languages, word boundaries are not marked.

But word boundaries are not arbitrary breakpoints in the signal : **they are the natural places to break up the signal if we want to code it most efficiently.** Therefore, we should expect that they can be recovered nearly correctly by using low-level statistics.

Considering strings without markers for word boundaries, the word boundaries are the simplest example of **geometric patterns**. The n-gram statistics are all examples of ‘value statistics’, i.e. frequencies derived from the values of the signal.

There are various approaches to finding the n-gram trace of word breaks and fairly sophisticated analyses seem to be needed to get the most accurate results. But we can do surprisingly well using a very simple method based on the concept of **mutual information** (as suggested by Brent (1999)).
Take a text from which the spaces have been removed or take a phonetic transcription of speech. The idea is to look at each point in the signal and **ask how much the preceding 2 or 3 characters tell us about the next 2 or 3 characters**. If we are in the middle of a word, these small strings constrain each other, i.e. they have high mutual information. If we are between two words, these fragments are much more nearly independent.

To define these statistics, take any n and m and write a random string $\mathcal{S}_{n+m}$ as the concatenation $\mathcal{S}'_n\mathcal{S}''_m$ of its initial substring of length n and its final substring of length m. Then we can consider how much information the first substring $\mathcal{S}'_n$ gives us about its successor $\mathcal{S}''_m$, i.e. $MI(\mathcal{S}'_n, \mathcal{S}''_m)$. This is computable in terms of the F_k's :

$$\begin{aligned}
MI(\mathcal{S}'_n, \mathcal{S}''_m) &= \mathbb{E}\log_2\left(\frac{P_{n+m}(a_1\cdots a_{n+m})}{P_n(a_1\cdots a_n)P_m(a_{n+1}\cdots a_{n+m})}\right) \\
&= \mathbb{E}\log_2(P_{n+m}(a_{n+1}\cdots a_{n+m}|a_1\cdots a_n)) - \mathbb{E}\log_2(P_m(a_1\cdots a_m)) \\
&= \sum_{k=n+1}^{n+m}\mathbb{E}\log_2(P_k(a_k|a_1\cdots a_{k-1})) - \sum_{k=1}^{m}\mathbb{E}\log_2(P_k(a_k|a_1\cdots a_{k-1})) \\
&= -(F_{n+1}+\cdots+F_{n+m}) + (F_1+\cdots+F_m) \\
&= \sum_{k=1}^{m}(F_k - F_{k+n})
\end{aligned}$$

Taking Shannon's estimates, we find that for English text, the mutual information of two adjacent characters is $F_1 - F_2 \approx 0.8$ bits. And the mutual information between the first and last pair in 4 consecutive characters is $F_1 + F_2 - F_3 - F_4 \approx 1.36$ bits.

But we can also consider not merely the expected value but the log of the individual fractions :

$$\beta(\sigma'_n, \sigma''_m) = \log_2\left(\frac{P_{n+m}(a_1 \cdots a_{n+m})}{P_n(a_1 \cdots a_n)P_m(a_{n+1} \cdots a_{n+m})}\right)$$

for specific strings $\sigma_{n+m} = \sigma'_n \sigma''_m$.

This kind of number occurs frequently : it is the log of the **likelihood ratio** of the long string $a_1 \cdots a_{n+m}$ measured with two models, one with dependencies between the two parts considered and one with the initial and final strings independent. We call this the **binding energy** with which σ'_n and σ''_m are coupled. It's also the difference in the optimal coding length of the combined string vs. that for separate coding of the two substrings.

This will vary above and below its expectation $MI(\mathcal{S}'_n, \mathcal{S}''_m)$.

The idea of this word-boundary finding algorithm is to choose some small n (e.g. 2) and calculate at each point in the text the mutual information between the n preceding characters and n succeeding characters $\beta((a_{i-n} \cdots a_{i-1}), (a_i \cdots a_{i+n-1}))$. Then put word boundaries at all i which are local minima of β. We expect the binding energy to oscillate above and below the mutual information, e.g. $\simeq 1.4$ for $n = 2$.

Example

Example from D. Mumford : “we downloaded off the web 8 novels by Mark Twain from which we removed all punctuation, including spaces (and numbers), eliminated capitalization and concatenated the lot into a long string with exactly 2,949,386 alphabetic characters.

We next took statistics of 4-tuples from this corpus and used this to compute the binding energy for each pair of consecutive pairs $(a_{n-2}a_{n-1})$ and $(a_n a_{n+1})$ of characters. We then found all local minima of the binding energy and eliminated local minima whose value was greater than 2.5 bits as not being strong enough. At this point, we find correctly about 60% on the word boundaries.

However, examining the output shows that a large number of the mistakes are due to missing a word boundary by one character. This has a simple explanation : low binding energy indicates that the 4-tuple $(a_{n-2}a_{n-1}a_n a_{n+1})$ is not a good unit and this may be caused by a break between a_{n-2} and a_{n-1} or between a_n and a_{n+1} as well as a break between a_{n-1} and a_n. Thus postprocessing to compare these three possible breaks is called for. In this we simply consider the likelihood ratios :

$$\frac{P(a_{n-3}a_{n-2})P(a_{n-1}a_n a_{n+1})}{P(a_{n-3}a_{n-2}a_{n-1})P(a_n a_{n+1})} \text{ and } \frac{P(a_{n-2}a_{n-1})P(a_n a_{n+1}a_{n+2})}{P(a_{n-2}a_{n-1}a_n)P(a_{n+1}a_{n+2})}$$

to see whether either of the breaks before or after the indicated one is better. After this step, we find 74% of the word breaks correctly”.

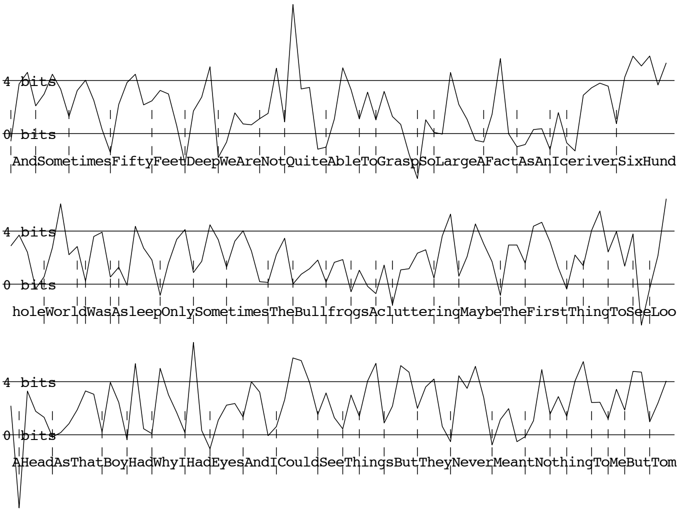


FIGURE: Binding energy of 4-tuples in three selections from Mark Twain. The text is shown without word breaks but with each new word beginning with a capital letter. The graph shows the binding energy and the vertical hatched lines are the word breaks indicated by the binding energy algorithm. Note that some of the incorrect word breaks are syllable breaks where two words might have occurred ('some-times') or where a common suffix is present ('clutter-ing'). Note too how the binding energy peaks at the pair 'qu'. Clearly, finding words needs a lot more than 4-tuple statistics.

Markov chain with words

Assuming we have identified the words properly, we can create a new stochastic language whose ‘characters’ are now these words and play the same game as with the original strings of characters. That is to say, we can compute the frequency of each word and each consecutive pair of words, etc. and we can make Markov models of the language using the word statistics.

Shannon did this for single words and for word pairs and his work has been extended up to triples of consecutive words with some extra assumptions. The problem is that even the largest databases are not big enough to get correct word triple statistics by brute force and more subtle techniques are called for. Shannon’s results are, first for single words with their correct frequencies :

```
REPRESENTING AND SPEEDILY IS AN GOOD APT OR COME CAN DIFFERENT
NATURAL HERE HE THE A IN CAME THE TO OF TO EXPERT GRAY COME
```

and with correct word pair frequencies :

```
THE HEAD AND IN FRONTAL ATTACK ON AN ENGLISH WRITER THAT THE
CHARACTER OF THIS POINT IS THEREFORE ANOTHER METHOD FOR THE
```

One gets a clear sense of convergence of these models to the true stochastic language.

Relationship between the character based models and the word based models

We can formalize the relationship between the character based models and the word based models like this. Let Ω_N be the set of strings of characters of length N without spaces and let Λ be the lexicon, i.e. the set of strings which are single words. Each word has a length, i.e. the number of characters in it. Let Ω_N^Λ be the set of strings of words whose total length is at least N and less than N if the last word is deleted. We get a map :

$$sp : \Omega_N^\Lambda \to \Omega_N$$

gotten by spelling out the string of words as a string of characters and omitting the small spillover for the last word. If $\sigma \in \Omega_N$, then writing $\sigma = sp(\tau)$ for some $\tau \in \Omega_N^\Lambda$ is the same thing is specifying the set of word boundaries in the string σ.

This is a very common situation in Pattern Theory. We have the observed signal – in this case, the string of characters. This signal has extra structure which is not given to us explicitly – in this case, its partition into words. We seek **to infer this hidden structure**, which amounts to lifting the sample from the probability space of observed signals to the larger probability space of signals with extra structure.

A typical method for lifting the signal is **maximum likelihood**, called the ML estimate. We use our best available model for the probability distribution on the larger space and find or approximate the most probable lifting of the signal from the space of observed signals to the larger space. In our case, we may put an n-gram distribution $P_N^{(n)}$ on Ω_N^Λ and seek the $\tau \in \Omega_N^\Lambda$ such that $sp(\tau) = \sigma$ which maximizes $P_N^{(n)}(\tau)$. In general, finding the ML lifting is difficult. But for this case of parsing a signal of characters into words, the method of dynamic programming is available.

Algorithm : Dynamic programming

The dynamic programming algorithm of Bellman is a very efficient algorithm to compute the minimum of a function F of n variables $x_1, \ldots, x_n$, provided this function can be decomposed as the sum of functions $f_i(x_i, x_{i+1})$.

Theorem

If $F(x_1, ..., x_n)$ is a real-valued function of n variables $x_i \in S_i$, S_i being a finite set, of the form

$$F(x_1, ..., x_n) = f_1(x_1, x_2) + f_2(x_2, x_3) + ... + f_{n-1}(x_{n-1}, x_n)$$

then one can compute the global minimum of F in time $O(s^2 n)$ and space $O(sn)$, where $s = \max_i |S_i|$.

The algorithm goes like this :

1. First initialize h_2 and Φ_2 by :

$$\begin{aligned} \forall x_2 \in S_2, \quad h_2(x_2) &= \min_{x_1 \in S_1} f_1(x_1, x_2) \\ \forall x_2 \in S_2, \quad \Phi_2(x_2) &= \operatorname*{argmin}_{x_1 \in S_1} f_1(x_1, x_2) \end{aligned}$$

2. We now loop over the variable k. At each stage, we will have computed :

$$\begin{aligned} \forall x_k \in S_k, \quad h_k(x_k) &= \min_{x_1,..,x_{k-1}} [f_1(x_1, x_2) + ... + f_{k-1}(x_{k-1}, x_k)] \\ \forall x_k \in S_k, \quad \Phi_k(x_k) &= \operatorname*{argmin}_{x_{k-1}}(\min_{x_1,..,x_{k-2}} [f_1(x_1, x_2) + ... + f_{k-1}(x_{k-1}, x_k)]). \end{aligned}$$

   Then we define :

$$\begin{aligned} \forall x_{k+1} \in S_{k+1}, \quad h_{k+1}(x_{k+1}) &= \min_{x_1,..,x_k} [f_1(x_1, x_2) + ... + f_{k-1}(x_{k-1}, x_k) + f_k(x_k, x_{k+1})] \\ &= \min_{x_k}(h_k(x_k) + f_k(x_k, x_{k+1})) \\ \forall x_{k+1} \in S_{k+1}, \quad \Phi_{k+1}(x_{k+1}) &= \operatorname*{argmin}_{x_k}(h_k(x_k) + f_k(x_k, x_{k+1})). \end{aligned}$$

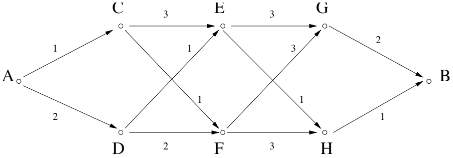
3. At the end, we let $h = \min_{x_n}(h_n(x_n))$ and set :

$$\bar{x}_n = \operatorname*{argmin}_{x_n}(h_n(x_n)), \quad \bar{x}_{n-1} = \Phi_n(\bar{x}_n), \cdots, \bar{x}_1 = \Phi_2(\bar{x}_2).$$

   Then h is the minimum of F and $F(\bar{x}_1, ..., \bar{x}_n) = h$.

If we look at the complexity of the algorithm, we see that at step k, for all x_{k+1} we have to search $\min_{x_k}(h_k(x_k) + f_k(x_k, x_{k+1}))$, and since there are n steps, the complexity is in $O(ns^2)$. And moreover we have to store all the $\Phi_k(x_k)$, which means that the complexity in space is $O(sn)$.

Example : Suppose that we want to find the minimum length path from A to B knowing that we have the following graph :



If D_X denotes the minimal distance from A to a point $\mathcal{X}$, then we can compute these in the order :

$$\begin{aligned}
D_E &= \min(d(A,C) + d(C,E), d(A,D) + d(D,E)) = 3 \\
D_F &= \min(d(A,C) + d(C,F), d(A,D) + d(D,F)) = 2 \\
D_G &= \min[D_E + d(E,G), D_F + d(F,G)] = 5 \\
D_H &= \min[D_E + d(E,H), D_F + d(F,H)] = 4 \\
D_B &= \min[D_G + d(G,B), D_H + d(H,B)] = 5
\end{aligned}$$

And finally, working backwards, we find that the “best” path from A to B is A, D, E, H, B.

Word boundaries revisited

Let's illustrate how the algorithm can be used to find the maximum likelihood partition of a character string $\sigma = (a_1 \cdots a_N)$ into words. We assume the probability model on words is the 1st order Markov model, i.e. we assume the words λ are independent and distributed with probabilities $P(\lambda)$. Thus if $\tau = (\lambda_1 \cdots \lambda_M)$, we will want to minimize :

$$\log(1/P(\tau)) = -\sum_{1}^{M} \log(P(\lambda_l))$$

over all sequences τ of words which expand to the string σ.

When we parse the string σ into words, each letter a_i will be assigned to a specific letter in a specific word, that is to an element of the set :

$$S_i = \{(\lambda, k) | \lambda \in \Lambda, 1 \leq k \leq |\lambda|, a_i = k^{th} \text{ letter in } \lambda\}.$$

We define the cost function on $S_i \times S_{i+1}$ by :

$$f_i((\lambda_i, k_i), (\lambda_{i+1}, k_{i+1})) = \begin{cases} -\log(P(\lambda_{i+1})) \ \text{ if } \ k_{i+1} = 1, k_i = |\lambda_i| \\ 0 \ \text{ if } \ \lambda_{i+1} = \lambda_i, k_{i+1} = k_i + 1 \\ \infty \ \ \text{otherwise} \end{cases}$$

When we minimize the sum of the f's, we must avoid the infinite values, so the values of the x_i simply give us a partition of σ into words λ_l and the penalty is just the sum of the corresponding $-\log(P(\lambda_l))$'s, which is the negative log probability of the parse.