Probability and expected document frequency of discontinued word sequences

An efficient method for their exact computation

Antoine Doucet — Helena Ahonen-Myka

Department of Computer Science
P.O. Box 68 (Gustaf Hällströmin katu 2b)
FI-00014 University of Helsinki, Finland
{Antoine.Doucet, Helena.Ahonen-Myka}@cs.Helsinki.fi

ABSTRACT. *We present an efficient technique for calculating the probability of occurrence of a discontinued sequence of words, i.e., the probability that those words occur, and that they occur in a given order, regardless of which and how many other words may occur between them. The procedure we introduce for words and documents may be generalized to any type of sequential data, e.g., item sequences and transactions. Our method relies on the formalization into a particular Markov chain model, whose specificities are combined with techniques of probability and linear algebra to offer competitive computational complexity. This work is further extended to permit the efficient calculation of the expected document frequency of a sequence. We finally present an application, a fast, automatic, and direct method to evaluate the interestingness of word sequences, by comparing their expected and observed frequencies.*

RÉSUMÉ. *Nous présentons une technique efficace pour calculer la probabilité d'une séquence de mots éventuellement discontigus, c'est-à-dire la probabilité que ces mots apparaissent dans un ordre donné, quel que soit le nombre d'autres mots pouvant apparaître entre eux. Notons qu'en lieu et place de mots et de documents, nous pouvons utiliser tout type de données séquentielles. Notre approche est basée sur une formalisation du problème en une chaîne de Markov particulière, dont nous présentons et exploitons les spécificités afin d'obtenir une complexité compétitive. Nous développons notre approche plus avant afin de calculer la fréquence documentaire attendue d'une séquence donnée. Cet article présente finalement une application de ces travaux : une méthode automatique pour l'évaluation directe de l'intérêt d'une séquence de mots, par le biais de comparaisons statistiques entre leurs fréquences attendues et observées.*

KEYWORDS: *word sequences, n-grams, lexical cohesion, information retrieval.*

MOTS-CLÉS : *séquences de mots, n-grams, cohésion lexicale, recherche d'information.*

TAL. Volume 46 – n°2/2005, pages 13 to 37

1. Introduction

Due to the higher information content and specificity of phrases versus words, information retrieval researchers have always been interested in multi-word units. However, the definition of what makes a few words form a unit has varied with time, and notably through the evolution of computational capacities.

The first models, introduced until the late 1980's, came with numerous restrictions. Mitra et al. (Mitra *et al.,* 1987), for example, defined phrases as adjacent pairs of words occurring in at least 25 documents of the TREC-1 collection. Choueka et al. (Choueka *et al.,* 1983) extracted adjacent word sequences of length up to 6. The extraction of sequences of longer size was then intractable. The adjacency constraint is regrettable, as natural language often permits to express similar concepts by introducing one or more words between two others. For example, the phrases "President John Kennedy" and "President Kennedy" are likely to refer to the same person. Church and Hanks (Church *et al.,* 1990) proposed a technique based on the notion of mutual information, which permits to produce word pairs occurring in the same window, regardless of their relative positions.

A new trend started in the 1990's, as linguistic information started to be used to filter out "undesirable" patterns. The idea consists in using parts-of-speech (POS) analysis to automatically select (or skip) the phrases matching a given set of linguistic patterns. Most recent extraction techniques still rely on a combination of statistical and syntactical methods (Smadja, 1993, Frantzi *et al.,* 1998).

However, at a time when multilingual information retrieval is in full expansion, we think it is of crucial importance to propose language-independent techniques. There is very few research in this direction, as was suggested by a recent workshop on multiword expressions (Tanaka *et al.,* 2004) where most of the 11 accepted papers presented monolingual techniques, in a total of 6 distinct languages.

Dias (Dias *et al.,* 2000) introduced an elegant generalization of conditional probabilities to n-grams extraction. The normalized expectation of an n-words sequence is the average expectation to see one of the words occur in a position, given the position of occurrence of all the others. Their main metric, the mutual expectation, is a variation of the normalized expectation that rewards n-grams occurring more frequently. While the method is language-independent and does not require word adjacency, it still recognizes phrases as a very rigid concept. The relative word positions are fixed, and to recall our previous example, no relationship is taken into account between "President John Kennedy" and "President Kennedy".

We present a technique that permits to efficiently calculate the exact probability (respectively, the expected document frequency) of a given sequence of n words to occur in a document of size l, (respectively, in a document collection D) with an unlimited number of other words eventually occurring between them. We assume that words occur independently, i.e., the probability of occurrence of a word in a given position does not depend on its context.

The main challenges we had to handle in this work were to avoid the computational issue of using a potentially unlimited distance between each two words, while not making those distances rigid (we do see an occurrence of "President Kennedy" in the text fragment "President John Kennedy"). Achieving language-independence (neither stoplists nor POS analysis are used) and dealing with document frequencies rather than term frequencies are further specificics of this work.

An application of this result is a *fast and automatic technique to directly evaluate the interestingness* of word sequences. Phrase extraction techniques often output a number of uninteresting sequences and it is desirable to have means to sort them by their level of interestingness. One main advantage of a ranked list over a set of phrasal descriptors is that it permits to the end-user to save time by reading through the most important findings first. This is especially important in real-life applications, where time is a limited resource. To rank a list of phrasal descriptors is not trivial, especially when it comes to comparing phrases of different lengths.

By exploiting statistical techniques, of *hypothesis testing*, our method provides the ability to do exactly that. The main idea is to account for the fact that word sequences are bound to happen by chance, and to compare how often a given word sequence *should* occur to how often it truly does. That is, the more the actual number of occurrences of a phrase is higher than its expected frequency, the stronger the lexical cohesion of that phrase. This evaluation technique is entirely language-independent, as well as domain- and application-independent. It permits to efficiently rank a set of candidate multi-word units, based on statistical evidence, without requiring manual assessment of a human expert.

In the next section, we will introduce the problem, present an approximation of the probability of an n-words sequence to occur in a document, and present a technique to reach for the exact results. We will then introduce our technique in full detail, including a complexity analysis that shows how it outperforms naive approaches. In Section 3, we will show how the probability of occurrence of an n-words sequence in a document can be generalized to compute its expected document frequency in a document collection, with a very reasonable computational complexity. Section 4 explains and experiments the use of statistical testing as an automatic way to rank general-purpose non-contiguous lexical cohesive relations. Section 5 concludes this paper.

2. The probability of discontinued occurrence of an n-words sequence

2.1. *Problem Definition*

Let $A_1, A_2, \ldots, A_n$ be n words, and d a document of length l (i.e., d contains l word occurrences). Each word A_i is assumed to occur independently with probability p_{A_i}.

Problem: In d, we want to calculate the probability $P(A_1 \rightarrow A_2 \rightarrow \cdots \rightarrow A_n, l)$ of the words $A_1, A_2, \ldots, A_n$ to occur at least once in this order, an unlimited number of interruptions of any size being permitted between each A_i and A_{i+1}, $1 \leq i \leq (n+1)$.

More definitions. Let D be the document collection, and W the set of all distinct words occurring in D. The calculation of the probability p_w of occurrence of a word w is a vast problem. In this work, we assume these probabilities to be given. In our experiments and along the examples, we use the term frequency of w in the whole document collection, divided by the total number of word occurrences in the collection. One reason to choose this approach is that the set of all word probabilities $\{p_w \mid \forall w \in W\}$ is then a (finite) probability space. Indeed, we have:

$$\sum_{w \in W} p_w = 1, \text{ and } p_w \geq 0, \forall w \in W.$$

For convenience, we will also simplify the notation of p_{A_i} to p_i, and define $q_i = 1 - p_i$, the probability of non-occurrence of the word A_i.

A running example. Let there be a hypothetic document collection containing only three different words A, B, and C, each occurring with equal frequency. We want to find the probability that the bigram $A \rightarrow B$ occurs in a document of length 3.

For such a simple example, we can afford an exhaustive manual enumeration. There exist $3^3 = 27$ distinct documents of size 3, each occurring with equal probability $\frac{1}{27}$. These documents are:
$\{AAA, \boxed{\text{AAB}}, AAC, \boxed{\text{ABA}}, \boxed{\text{ABB}}, \boxed{\text{ABC}}, ACA, \boxed{\text{ACB}}, ACC,$
$BAA, \boxed{\text{BAB}}, BAC, BBA, BBB, BBC, BCA, BCB, BCC,$
$CAA, \boxed{\text{CAB}}, CAC, CBA, CBB, CBC, CCA, CCB, CCC\}$
The 7 framed documents contain the n-gram AB. Thus, we have $p(A \rightarrow B, 3) = \frac{7}{27}$.

2.2. *A Decent Over-Estimation in the General Case*

We can attempt to enumerate the number of occurrences of $A_1 \rightarrow \cdots \rightarrow A_n$ in a document of size l, by separately counting the number of ways to form the $(n-1)$-gram $A_2 \rightarrow \cdots \rightarrow A_n$, given the l possible positions of A_1. For each of these possibilities, we can then separately count the number of ways to form the $(n-2)$-gram $A_3 \rightarrow \cdots \rightarrow A_n$, given the various possible positions of A_2 following that of A_1. And so on until we need to find the number of ways to form the 1-gram A_n, given the various possibilities left for placing A_{n-1}. This enumeration leads to n nested sums of binomial coefficients:

$$\sum_{pos_{A_1}=1}^{l-n+1} \left(\sum_{pos_{A_2}=pos_{A_1}+1}^{l-n+2} \left(\cdots \sum_{pos_{A_n}=pos_{A_{n-1}}+1}^{l} \binom{l - pos_{A_n}}{0} \right) \right), \qquad [1]$$

where each pos_{A_i}, $1 \leq i \leq n$, denotes the position of occurrence of A_i.

The following can be proved easily by induction:

$$\sum_{i=k}^{n} \binom{i}{k} = \binom{n+1}{k+1},$$

and we can use it to simplify Formula [1] by observing that:

$$\begin{aligned} \sum_{pos_{A_i}=pos_{A_{i-1}}+1}^{l-n+i} \binom{l-pos_{A_i}}{n-i} &= \sum_{pos_{A_i}=n-i}^{l-pos_{A_{i-1}}-1} \binom{pos_{A_i}}{n-i} \\ &= \binom{l-pos_{A_{i-1}}}{n-i+1}. \end{aligned}$$

Therefore, leaving further technical details to the reader, the previous nested summation [1] interestingly simplifies to $\binom{l}{n}$, which permits to obtain the following result:

$$enum\_overestimate(A_1 \rightarrow \cdots \rightarrow A_n, l) = \binom{l}{n} \cdot \prod_{i=1}^{n} p_i,$$

where $\binom{l}{n}$ is the number of ways to form the n-gram, and $\prod_{i=1}^{n} p_i$ the probability of conjoint occurrence of the words $A_1, \ldots, A_n$ (since we assumed that the probability of occurrence of a word in one position is independent of which words occur in other positions).

The big flaw of this result, and the reason why it is an approximation only, is that some of the ways to form the n-gram are obviously overlapping. Whenever we separate the alternative ways to form the n-gram, knowing that A_i occurs in position pos_{A_i}, with $1 \leq i \leq n$ and $(pos_{A_{i-1}}+1) \leq pos_{A_i} \leq (l-n+i)$, we do ignore the fact that A_i may also occur before position pos_{A_i}. In this case, we find and add different ways to form the same occurrence of the n-gram. We do enumerate each possible case of occurrence of the n-gram, but we count some of them more than once, since it is actually *the ways* to form the n-gram that are counted.

Running Example. This is better seen by returning to the running example presented in Section 2.1. As described above, the upper-estimate of the probability of the bigram $A \rightarrow B$, based on the enumeration of the ways to form it in a document of size 3 is: $(\frac{1}{3})^2\binom{3}{2} = \frac{9}{27}$, while the actual probability of $A \rightarrow B$ is $\frac{7}{27}$. This stems from the fact that in the document AAB (respectively ABB), there exist two ways to form the bigram $A \rightarrow B$, using the two occurrences of A (respectively B). Hence, out of the 27 possible equiprobable documents, 9 ways to form the bigram $A \rightarrow B$ are found in the 7 documents that contain it.

With longer documents, the loss of precision due to those cases can be considerable. Still assuming we are interested in the bigram $A \rightarrow B$, we will count one extra occurrence for every document that matches *A*B*B*, where * is used as a wildcard. Similarly, 8 ways to form $A \rightarrow B$ are found in each document matching *A*A*B*B*B*B*.

2.3. *Exact Probability of a Discontiguous Word Sequence*

With a slightly different approach, we can actually reach the exact result. The previous technique exposed *overlapping* ways to form a word sequence. That is why the result was only an overestimate of the desired probability.

In this section, we will present a way to categorize the different sets of documents of size l in which the n-gram $A_1 \rightarrow \cdots \rightarrow A_n$ occurs, with the property that all the sets are disjoint and that no case of occurrence of the n-gram is forgotten. This ensures that we can calculate $p(A_1 \rightarrow \cdots \rightarrow A_n, l)$ by summing up the probabilities of each set of documents where $A_1 \rightarrow \cdots \rightarrow A_n$ occurs.

2.3.1. *A Disjoint Categorization of Successful Documents*

We can split the *successful* documents (those in which the n-gram occurs) of size l, depending on the position from which a successful outcome is guaranteed. For example, and for $l \geq n$, the documents of size l for which success is guaranteed as soon as from position n onwards can be represented by the set of documents E_0:

$$E_0 = \{A_1 A_2 \dots A_n W^{l-n}\},$$

where as defined earlier W is the set of all words in the document collection, and using regular expression notation, W^{l-n} stands for a concatenation of any $(l-n)$ words of W. Because each word is assumed to occur independently of the others, the probability of a given document is a conjunction of independent events, and therefore it equals the multiplication of the probability of all the words in the document. The probability of the set of documents E_0 is the probability of occurrence of A_1, $A_2 \dots$, and A_n once, plus $(l-n)$ times any word of W (with probability 1). Therefore,

$$p(E_0) = p_1 \cdot p_2 \dots p_n \cdot 1^{l-n} = \prod_{i=1}^{n} p_i.$$

Similarly, the documents in which the occurrence of the n-gram is guaranteed as soon as the $(n+1)$-th word can be represented by the set E_1 where, for $1 \leq k \leq n$, $i_k \geq 0$:

$$E_1 = \{\bar{A_1}^{i_1} A_1 \bar{A_2}^{i_2} A_2 \dots \bar{A_n}^{i_n} A_n W^{l-n-1} \mid \sum_{k=1}^{n} i_k = 1\},$$

where $\bar{A_k}$, $1 \leq k \leq n$, represents any word but A_k. In other words, E_1 is the set of all documents where a total number of 1 word is inserted before each word of the n-gram. The probability of this set of documents is:

$$\begin{aligned} p(E_1) &= (q_1 p_1 p_2 \dots p_n 1^{l-n-1}) + (p_1 q_2 p_2 \dots p_n 1^{l-n-1}) + \\ &\quad \cdots + (p_1 p_2 \dots p_{n-1} q_n p_n 1^{l-n-1}) \\ &= \prod_{i=1}^{n} p_i \sum_{k=1}^{n} q_k. \end{aligned}$$

We can proceed similarly for the following positions after which a successful outcome is guaranteed. Finally, the same idea provides an expression for the set of documents for which the occurrence of the n-gram was not complete before the word in position l (and therefore the last word of the document is A_n):

$$E_{l-n} = \{\bar{A_1}^{\,i_1} A_1 \bar{A_2}^{\,i_2} A_2 \dots \bar{A_n}^{\,i_n} A_n \mid \sum_{k=1}^{n} i_k = (l-n)\}.$$

The set E_{l-n} contains all the possibilities to disseminate exactly $(l-n)$ other words before the words of the n-gram. Its probability of occurrence is:

$$\begin{aligned}
p(E_{l-n}) &= p\left(\left\{\bar{A_1}^{\,i_1} A_1 \dots \bar{A_n}^{\,i_n} A_n \mid \sum_{k=1}^{n} i_k = (l-n)\right\}\right) \\
&= p_n \sum_{i_n=0}^{l-n} q_n^{i_n} p\left(\left\{\bar{A_1}^{\,i_1} A_1 \dots \bar{A_{n-1}}^{\,i_{n-1}} A_{n-1} \mid \sum_{k=1}^{n-1} i_k = (l-n-i_n)\right\}\right) \\
&= p_n p_{n-1} \sum_{i_n=0}^{l-n} \sum_{i_{n-1}=0}^{l-n-i_n} q_n^{i_n} q_{n-1}^{i_{n-1}} \\
&\qquad p\left(\left\{\bar{A_1}^{\,i_1} A_1 \dots \bar{A_{n-2}}^{\,i_{n-2}} A_{n-2} \mid \sum_{k=1}^{n-2} i_k = (l-n-i_n-i_{n-1})\right\}\right) \\
&= \dots \\
&= \prod_{i=1}^{n} p_i \sum_{i_n=0}^{l-n} \dots \sum_{i_2=0}^{l-n-(i_n+\dots+i_3)} q_n^{i_n} \dots q_2^{i_2} q_1^{l-n-(i_n+\dots+i_2)}.
\end{aligned}$$

In general, for $0 \le k \le l-n$, we can write:

$$p(E_k) \;=\; \prod_{i=1}^{n} p_i \sum_{i_n=0}^{k} \dots \sum_{i_2=0}^{k-(i_n+\dots+i_3)} q_1^{\,k-\sum_{j=2}^{n} i_j} q_2^{i_2} \dots q_n^{i_n}.$$

2.3.2. *The precise formula*

It is clear that the sets E_k, for $0 \le k \le (l-n)$, are all disjoint, because in any document, the presence of the n-gram is ensured from only one position onwards. It is also evident that in any document of size l containing the n-gram, its occurrence will be ensured between the n-th and l-th position. Therefore the sets E_k are mutually exclusive, for $0 \le k \le (l-n)$, and their union contains all the documents of size l where $A_1 \to \dots \to A_n$ occurs. Consequently,

$$p(A_1 \to \dots \to A_n, l) \;=\; \sum_{k=0}^{l-n} p(E_k)$$

Finally, the formula of the probability of occurrence of a discontiguous sequence of length n in a document of length l is:

$$p(A_1 \to \cdots \to A_n, l) = \prod_{i=1}^{n} p_i \sum_{i_n=0}^{l-n} \cdots \sum_{i_1=0}^{l-n-(i_n+\cdots+i_2)} q_1^{i_1} q_2^{i_2} \ldots q_n^{i_n}. \qquad [2]$$

Running Example. For better comprehension, let us return to the running example:

$$\begin{aligned} p(A \to B, 3) &= p_a p_b \sum_{i_b=0}^{1} \sum_{i_a=0}^{1-i_b} q_a^{i_a} q_b^{i_b} \\ &= p_a p_b \left(1 + q_a + q_b\right). \end{aligned}$$

And we find the exact result of $\frac{7}{27}$. But we will now see that the direct calculation of Formula [2] is is not satisfying in practice because of an exponential computational complexity.

2.3.3. *Computational Complexity*

Let us observe the steps involved in the computation of $p(E_k)$, $0 \leq k \leq l - n$. To calculate this probability consists in multiplying n values (the p_i's) by a summation of summations. The total number of terms resulting from these nested summations equals the total number of ways to insert k terms in n different positions: n^k. Thus, $p(E_k)$ is the result of multiplying n values by a summation of n^k distinct terms, each individually calculated by k multiplications. Hence, the gross number of operations to calculate $p(A_1 \to \cdots \to A_n, l)$ with Formula [2] is:

$$n \sum_{k=0}^{l-n} k n^k.$$

Therefore, the order of complexity of the direct computation of Formula [2] is $O(l n^{l-n})$. Consequently, this formula is hardly usable at all, except for extremely short documents and length-restricted n-grams.

2.4. *Efficient Computation through a Markov Chain Formalization*

We found a way to calculate the probability of discontiguous occurrence of an n-words sequence in a document of size l. However, its computational complexity cuts clear any hope to use the result in practice. The following approach permits to reach the exact result with a far better complexity.

2.4.1. *An Absorbing Markov Chain*

Another interesting way to formalize the problem is to consider it as a sequence of l trials whose outcomes are $X_1, X_2, \ldots, X_l$. Let each of these outcomes belong to the set $\{0, 1, \ldots, n\}$, where the outcome i signifies that the i-gram $A_1 \to A_2 \to \cdots \to A_i$ has already occurred. This sequence of trials verifies the following two properties:

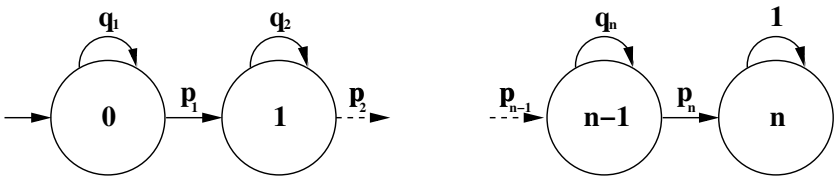

Figure 1. *The state-transition diagram of the Markov Chain M.*

(i) All the outcomes $X_1, X_2, \ldots, X_l$ belong to a finite set of outcomes $\{0, 1, \ldots, n\}$ called the *state space* of the system. If i is the outcome of the m-th trial ($X_m = i$), then we say that the system is in state i at the m-th step. In other words, the i-gram $A_1 \rightarrow A_2 \rightarrow \cdots \rightarrow A_i$ has been observed after the m-th word of the document.

(ii) The second property is called the *Markov property*: the outcome of each trial depends at most upon the outcome of the immediately preceding trial, and not upon any other previous outcome. In other words, *the future is independent of the past, given the present*. This is verified indeed; if we know that we have seen $A_1 \rightarrow A_2 \rightarrow \cdots \rightarrow A_i$, we only need the probability of A_{i+1} to determine the probability that we will see more of the desired n-gram during the next trial.

These two properties are sufficient to call the defined stochastic process a (finite) *Markov chain*. The problem can thus be represented by an $(n + 1)$-states Markov chain M (see Figure 1). The state space of the system is $\{0, 1, \ldots, n\}$ where each state, numbered from 0 to n tells how much of the n-gram has already been observed. Presence in state i means that the sequence $A_1 \rightarrow A_2 \rightarrow \cdots \rightarrow A_i$ has been observed.Therefore, $A_{i+1} \rightarrow \cdots \rightarrow A_n$ remains to be seen, and the following expected word is A_{i+1}. It will be the next word with probability p_{i+1}, in which case a state transition will occur from i to $(i + 1)$. A_{i+1} will not be the following word with probability q_{i+1}, in which case we will remain in state i. Whenever we reach state n, we can denote the experience a success: the whole n-gram has been observed. The only outgoing transition from state n leads to itself with associated probability 1 (such a state is said to be *absorbing*).

2.4.2. *Stochastic Transition Matrix (in general)*

Another way to represent this Markov chain is to write its transition matrix. For a general finite Markov chain, let $p_{i,j}$ denote the transition probability from state i to state j for $1 \leq i, j \leq n$. The (one-step) stochastic transition matrix is:

$$P = \begin{pmatrix} p_{1,1} & p_{1,2} & \cdots & p_{1,n} \\ p_{2,1} & p_{2,2} & \cdots & p_{2,n} \\ & & \cdots & \\ p_{n,1} & p_{n,2} & \cdots & p_{n,n} \end{pmatrix}.$$

Theorem 2.1 *(Feller, 1968) Let P be the transition matrix of a Markov chain process. Then the m-step transition matrix is equal to the m-th power of P. Furthermore, the entry $p_{i,j}(m)$ in P^m is the probability of stepping from state i to state j in exactly m transitions.*

2.4.3. *Our stochastic transition matrix of interest*

For the Markov chain M defined above, the corresponding stochastic transition matrix is the following $(n+1) \times (n+1)$ square matrix:

$$M = \begin{array}{c} \text{states} \\ 0 \\ 1 \\ \\ \\ n \end{array} \begin{array}{c} \begin{array}{ccccc} 0 & 1 & \dots & n-1 & n \end{array} \\ \begin{pmatrix} q_1 & p_1 & \dots & \dots & 0 \\ 0 & q_2 & \ddots & & \vdots \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ \vdots & & \ddots & q_n & p_n \\ 0 & \dots & \dots & 0 & 1 \end{pmatrix} \end{array}.$$

Therefore, the probability of the n-gram $A_1 \rightarrow A_2 \rightarrow \cdots \rightarrow A_n$ to occur in a document of size l is the probability of stepping from state 0 to state n in exactly l transitions. Following Theorem 2.1, this value resides at the intersection of the first row and the last column of the matrix M^l:

$$M^l = \begin{pmatrix} m_{1,1}(l) & m_{1,2}(l) & \dots & \boxed{m_{1,n+1}(l)} \\ m_{2,1}(l) & m_{2,2}(l) & \dots & m_{2,n+1}(l) \\ & & \dots & \\ m_{n+1,1}(l) & m_{n+1,2}(l) & \dots & m_{n+1,n+1}(l) \end{pmatrix}.$$

Thus, the result we are aiming at can simply be obtained by calculating M^l, and looking at the value in the upper-right corner. In terms of computational complexity, however, one must note that to multiply two $(n+1)\times(n+1)$ square matrices, we need to compute $(n+1)$ multiplications and n additions to calculate each of the $(n+1)^2$ values composing the resulting matrix. To raise a matrix to the power l means to repeat this operation $l-1$ times. The resulting time complexity is then $O(ln^3)$.

One may object that there exist more time-efficient algorithms for matrix multiplication. The lowest exponent currently known is $O(n^{2.376})$ (Coppersmith *et al.,* 1987). These results are achieved by studying how matrix multiplication depends on bilinear and trilinear combinations of factors. The strong drawback of such techniques is the presence of a constant so large that it removes the benefits of the lower exponent for all practical sizes of matrices (Horn *et al.,* 1994). For our purpose, the use of such an algorithm is typically more costly than to use the naive $O(n^3)$ matrix multiplication.

Linear algebra techniques, and a careful exploitation of the specificities of the stochastic matrix M will, however, permit to perform a few transformations that will drastically reduce the computational complexity of M^l over the use of any matrix multiplication algorithm (it has been proved that the complexity of matrix multiplication cannot possibly be lower than $O(n^2)$).

2.4.4. *The Jordan normal form*

Definition: A *Jordan block* J_λ is a square matrix whose elements are zero except for those on the principal diagonal, which are equal to λ, and those on the first superdiagonal, which are equal to unity. Thus:

$$J_\lambda = \begin{pmatrix} \lambda & 1 & & 0 \\ & \lambda & \ddots & \\ & & \ddots & 1 \\ 0 & & & \lambda \end{pmatrix}.$$

Theorem 2.2 *(Jordan normal form) (Noble* et al., *1977) If A is a general square matrix, then there exists an invertible matrix S such that*

$$J = S^{-1}AS = \begin{pmatrix} J_1 & & 0 \\ & \ddots & \\ 0 & & J_k \end{pmatrix},$$

where the J_i are $n_i \times n_i$ Jordan blocks. The same eigenvalues may occur in different blocks, but the number of distinct blocks corresponding to a given eigenvalue is equal to the number of eigenvectors corresponding to that eigenvalue and forming an independent set. The number k and the set of numbers $n_1, \ldots, n_k$ are uniquely determined by A.

In the following subsection we will show that M is such that there exists only one block for each eigenvalue.

2.4.5. *Uniqueness of the Jordan block corresponding to any given eigenvalue of M*

Theorem 2.3 *For the matrix M, no two eigenvectors corresponding to the same eigenvalue can be linearly independent. Following theorem 2.2, this implies that there exists a one-one mapping between Jordan blocks and eigenvalues.*

Proof. Because M is triangular, its characteristic polynomial is the product of the diagonals of $(\lambda I_{n+1} - M)$: $f(\lambda) = (\lambda - q_1)(\lambda - q_2)\ldots(\lambda - q_n)(\lambda - 1)$. The eigenvalues of M are the solutions of the equation $f(\lambda) = 0$. Therefore, they are the distinct q_i's, and 1.

Now let us show that whatever the order of multiplicity of such an eigenvalue (how many times it occurs in the set $\{q_1, \ldots, q_n, 1\}$), it has only one associated eigenvector. The eigenvectors associated to a given eigenvalue e are defined as the non null solutions of the equation $M \cdot V = e \cdot V$. If we write the coordinates of V as $[v_1, v_2, \ldots, v_{n+1}]$, we can observe that $M \cdot V = e \cdot V$ results in a system of $(n+1)$ equations, where, for $1 \leq j \leq n$, the j-th equation permits to express v_{j+1} in terms of v_j, and therefore in terms of v_1. That is,

$$\text{for } 1 \leq j \leq n: \; v_{j+1} = \frac{e - q_j}{p_j} v_j = \frac{(e - q_j)\ldots(e - q_1)}{p_j \ldots p_1} v_1.$$

In general (for all the q_i's), v_1 can be chosen freely to have any non-null value. This choice will uniquely determine all the values of V.

Since the general form of the eigenvectors corresponding to any eigenvalue of M is $V = [v_1, v_2, \ldots, v_{n+1}]$, where all the values can be determined uniquely by the free choice of v_1, it is clear that no two such eigenvectors can be linearly independent. Hence, one and only one eigenvector corresponds to each eigenvalue of M. □

Following theorem 2.2, this means that there is a single Jordan block for each eigenvalue of M, whose size equals the order of algebraic multiplicity of the eigenvalue, that is, its number of occurrences in the principal diagonal of M. In other words, there is a distinct Jordan block for every distinct q_i (and its size equals the number of occurrences of q_i in the main diagonal of M), plus a block of size 1 for the eigenvalue 1. Therefore we can write:

$$J = S^{-1}MS = \begin{pmatrix} \boxed{J_{e_1}} & & 0 \\ & \ddots & \\ 0 & & \boxed{J_{e_q}} \end{pmatrix},$$

where the J_{e_i} are $n_i \times n_i$ Jordan blocks, corresponding to the distinct eigenvalues of M. Following the general properties of the Jordan normal form, we have $M^l = SJ^lS^{-1}$ and:

$$J^l = \begin{pmatrix} \boxed{J_{e_1}^l} & & 0 \\ & \ddots & \\ 0 & & \boxed{J_{e_q}^l} \end{pmatrix}.$$

Therefore, by multiplying the first row of S by J^l, and multiplying the resulting vector by the last column of S^{-1}, we do obtain the upper right value of M^l, that is, the probability of the n-gram $(A_1 \to \cdots \to A_n)$ to appear in a document of size l.

2.4.6. *Calculating powers of a Jordan block*

As mentioned above, to raise J to the power l, we can simply write a direct sum of the Jordan blocks raised to the power l. In this section, we will show how to compute $J_{e_i}^l$ for a Jordan block J_{e_i}.

Let us define D_{e_i} and N_{e_i} such that $J_{e_i} = D_{e_i} + N_{e_i}$, where D_{e_i} contains only the principal diagonal of J_{e_i}, and N_{e_i} only its first superdiagonal. That is,

$$D_{e_i} = \begin{pmatrix} e_i & & & 0 \\ & e_i & & \\ & & \ddots & \\ 0 & & & e_i \end{pmatrix} \text{ and } N_{e_i} = \begin{pmatrix} 0 & 1 & & 0 \\ & & \ddots & \\ & & & 1 \\ 0 & & & 0 \end{pmatrix}.$$

Observing that $N_{e_i} D_{e_i} = D_{e_i} N_{e_i}$, we can exploit the binomial theorem and shorten the resulting summation by observing that N_{e_i} is nilpotent ($N_{e_i}^k = 0, \forall k \geq n_i$):

$$J_{e_i}^l = (D_{e_i} + N_{e_i})^l = \sum_{k=0}^{l} \binom{l}{k} N_{e_i}^k D_{e_i}^{l-k} = \sum_{k=0}^{n_i-1} \binom{l}{k} N_{e_i}^k D_{e_i}^{l-k}$$

Hence, to calculate $J_{e_i}^l$, one can compute the powers of D_{e_i} and N_{e_i} from 0 to l, which is a fairly simple task. The power of a diagonal matrix is easy to compute, as it is another diagonal matrix where each term of the original matrix is raised to the same power as the matrix. $D_{e_i}^j$ is thus identical to D_{e_i}, except that the main diagonal is filled with the value e_i^j instead of e_i.

To compute $N_{e_i}^k$ is even simpler. Each multiplication of a power of N_{e_i} by N_{e_i} results in shifting the non-null diagonal one row upwards (the values on the first row are lost, and those on the last row are 0's).

The result of $N_{e_i}^k D_{e_i}^j$ resembles $N_{e_i}^j$, except that the ones on the only non-null diagonal (the j-th superdiagonal) are replaced by the value of the main diagonal of $D_{e_i}^j$, that is, e_i^j. Therefore, we have:

$$N_{e_i}^k D_{e_i}^{l-k} = \begin{pmatrix} 0 & e_i^{l-k} & & 0 \\ & & \ddots & \\ & & & e_i^{l-k} \\ 0 & & & 0 \end{pmatrix}.$$

Since each value of k corresponds to a distinct diagonal, the summation $\sum_{k=0}^{l} \binom{l}{k} N_{e_i}^k D_{e_i}^{l-k}$ is easily written as:

$$J_{e_i}^l = \sum_{k=0}^{l} \binom{l}{k} N_{e_i}^k D_{e_i}^{l-k} = \begin{pmatrix} \binom{l}{0} \cdot e_i^l & \dots & \binom{l}{k} \cdot e_i^{l-k} & \dots & \binom{l}{n_i-1} \cdot e_i^{l-n_i+1} \\ & \ddots & & \ddots & \vdots \\ & & \binom{l}{0} \cdot e_i^l & & \binom{l}{k} \cdot e_i^{l-k} \\ & & & \ddots & \vdots \\ 0 & & & & \binom{l}{0} \cdot e_i^l \end{pmatrix}.$$

2.4.7. *Conclusion*

The probability of the n-gram $A_1 \rightarrow \cdots \rightarrow A_n$ in a document of size l can be obtained as the upper-right value in the matrix M^l such that:

$$M^l = S J^l S^{-1} = S \begin{pmatrix} \boxed{J_{e_1}^l} & & 0 \\ & \ddots & \\ 0 & & \boxed{J_{e_q}^l} \end{pmatrix} S^{-1},$$

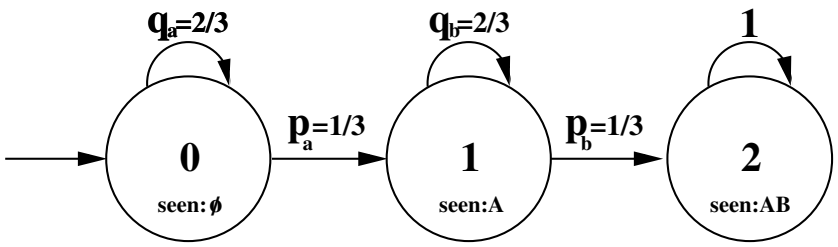

Figure 2. *The state-transition diagram of the Markov Chain corresponding to our running example.*

where the $J^l_{e_i}$ blocks are as described above, while S and S^{-1} are obtained through the Jordan Normal Form theorem (Theorem 2.2). We actually only need the first row of S and the last column of S^{-1}, as we are not interested in the whole matrix M^l but only in its upper-right value.

In the next subsection we will calculate the worst case time complexity of the technique that we just presented. Before that, let us return to the running example presented in subsection 2.1.

2.4.8. *Running Example*

The state-transition diagram of the Markov Chain corresponding to the bigram $A \rightarrow B$ has only three states (see Figure 2). The corresponding transition matrix is:

$$M_{re} = \begin{pmatrix} \frac{2}{3} & \frac{1}{3} & 0 \\ 0 & \frac{2}{3} & \frac{1}{3} \\ 0 & 0 & 1 \end{pmatrix}.$$

Following Theorem 2.2 on the Jordan normal form, there exists an invertible matrix S_{re} such that

$$J_{re} = S_{re}^{-1} M_{re} S_{re} = \begin{pmatrix} \boxed{J_{\frac{2}{3}}} & 0 \\ 0 & \boxed{J_1} \end{pmatrix},$$

where J_1 is a block of size 1, and $J_{\frac{2}{3}}$ a block of size 2 since $q_a = q_b = \frac{2}{3}$. We can actually write J_{re} as:

$$J_{re} = \begin{pmatrix} \frac{2}{3} & 1 & 0 \\ 0 & \frac{2}{3} & 0 \\ 0 & 0 & 1 \end{pmatrix}.$$

Since we seek the probability of the bigram $A \rightarrow B$ in a document of size 3, we need to calculate J^3_{re}:

$$J^3_{re} = \begin{pmatrix} \binom{3}{0}(\frac{2}{3})^3 & \binom{3}{1}(\frac{2}{3})^2 & 0 \\ 0 & \binom{3}{0}(\frac{2}{3})^3 & 0 \\ 0 & 0 & 1 \end{pmatrix} = \begin{pmatrix} \frac{8}{27} & \frac{4}{3} & 0 \\ 0 & \frac{8}{27} & 0 \\ 0 & 0 & 1 \end{pmatrix}.$$

In the next subsection, we will give further details as to the practical computation of S_{re} and the last column of its inverse S_{re}^{-1}. For now, let us simply assume they were calculated, and thus:

$$P(A \to B, 3) = \overbrace{(1\ 0\ 1)}^{\text{first row of } S} \begin{pmatrix} \frac{8}{27} & \frac{4}{3} & 0 \\ 0 & \frac{8}{27} & 0 \\ 0 & 0 & 1 \end{pmatrix} \overbrace{\begin{pmatrix} -1 \\ -\frac{1}{3} \\ 1 \end{pmatrix}}^{\text{last column of } S^{-1}} = \frac{7}{27}.$$

Our technique indeed obtains the right result. But how efficiently is it obtained? The purpose of the following subsection is to answer this question.

2.5. *Algorithmic Complexity*

The process of calculating the probability of occurrence of an n-gram in a document of size l consists of two main phases: calculating J^l, and computing the transformation matrix S and its inverse S^{-1}.

The following complexity analysis might be easier to follow, if studied together with the general formulas of M^l and the Jordan blocks as presented in the conclusion of Section 2.4.7.

2.5.1. *Time complexity of the J^l calculation*

Observing that each block J_i^l contains exactly n_i distinct values, we can see that J^l contains $\sum_{1 \le k \le q} n_k = n + 1$ distinct values. Those $(n + 1)$ values are $(n + 1)$ multiplications of a binomial coefficient by the power of an eigenvalue.

The computation of the powers between 0 and l of each eigenvalue is evidently achieved in $O(lq)$, because each of the q distinct eigenvalues needs to be multiplied by itself l times.

For every Jordan block J_i^l, the binomial coefficients to be computed are: $\binom{l}{0}, \binom{l}{1}, \ldots, \binom{l}{n_i - 1}$. For the whole matrix J^l, we thus need to calculate $\binom{l}{k}$ where $0 \le k \le max_{block}$ and $\max_{block} = \max_{i=1}^{q} n_i$. Observing that $\binom{l}{j+1} = \binom{l}{j} \frac{l-j}{j+1}$, and thus, that $\binom{l}{j+1}$ can be computed from $\binom{l}{j}$ in a constant number of operations, we see that the set $\{\binom{l}{k} \mid 1 \le k \le \max_{block}\}$ can be computed in $O(\max_{block})$.

Finally, all the terms of J^l are obtained by $(n + 1)$ multiplications of powers of eigenvalues (computed in $O(lq)$) and combinatorial coefficients (computed in $O(\max_{block})$). Note that if $l < n$, the probability of occurrence of the n-gram in l is immediately 0, since the n-gram is longer than the document. Therefore, the current algorithm is only used when $l \ge n \ge \max_{block}$. We can therefore conclude that **the time complexity of the computation of J^l is $O(lq)$**.

2.5.2. *Calculating the transformation matrix S*

Following general results of linear algebra (Noble *et al.,* 1977), the $(n+1) \times (n+1)$ transformation matrix S can be written as: $S = \left[S_1 S_2 \dots S_q\right]$, where each S_i is an $n_i \times (n+1)$ matrix corresponding to the eigenvalue e_i, and such that $S_i = \left[v_{i,1} v_{i,2} \dots v_{i,n_i}\right]$,where:

– $v_{i,1}$ is an eigenvector associated with e_i, thus such that $M v_{i,1} = e_i v_{i,1}$, and

– $v_{i,j}$, for all $j = 2 \dots n_i$, is a solution of the equation $M v_{i,j} = e_i v_{i,j} + v_{i,j-1}$.

The vectors $v_{i,1} v_{i,2} \dots v_{i,n_i}$ are sometimes called *generalized eigenvectors* of e_i. We have already seen in Section 2.4.4 that the first coordinate of each eigenvector can be assigned freely, and that every other coordinate can be expressed in function of its immediately preceding coordinate. Setting the first coordinate a_1 to 1, we can write:

$$v_{i,1} = \begin{pmatrix} a_1 \\ a_2 \\ \vdots \\ a_i \\ a_{i+1} \\ \vdots \\ a_{n+1} \end{pmatrix} = \begin{pmatrix} 1 \\ \frac{e_i - q_1}{p_1} \\ \vdots \\ \frac{(e_i - q_1) \dots (e_i - q_{i-1})}{p_1 \dots p_{i-1}} \\ 0 \\ \vdots \\ 0 \end{pmatrix}$$

The resolution of the system of $(n+1)$ linear equations following $M v_{i,2} = e_i v_{i,2} + v_{i,1}$ permits to write:

$$v_{i,2} = \begin{pmatrix} b_1 \\ b_2 \\ \vdots \\ b_i \\ b_{i+1} \\ b_{i+2} \\ \vdots \\ b_{i+k} \\ b_{i+k+1} \\ \vdots \\ b_{n+1} \end{pmatrix} = \begin{pmatrix} b_1 \\ \frac{a_1}{p_1} + \frac{e_i - q_1}{p_1} b_1 \\ \vdots \\ \frac{a_{i-1}}{p_{i-1}} + \frac{e_i - q_{i-1}}{p_{i-1}} b_{i-1} \\ \frac{a_i}{p_i} \\ \frac{e_i - q_{i+1}}{p_{i+1}} b_{i+1} \\ \vdots \\ \frac{e_i - q_{i+k-1}}{p_{i+k-1}} b_{i+k-1} \\ 0 \\ \vdots \\ 0 \end{pmatrix},$$

where k is such that $(i+k)$ is the position of second occurrence of e_i on the principal diagonal of M (that is, $q_{i+k} = q_i = e_i$), the position of first occurrence being i.

We can similarly write the other column vectors $v_{i,j}$, where $j = 3 \dots n_i$. Hence, it is clear that each coordinate of those vectors can be calculated in a constant number of

operations. Therefore, we can compute each column in $O(n)$, and **the whole matrix S in $O(n^2)$**.

Observe that the value of b_1 is free and that it can be set to 0, without loss of generality. The same is true for the value in the first row of each column vector $v_{i,j}$, where $j = 2 \dots n_i$. In our implementation (notably when applying this technique to the running example in Section 2.4.8), we made the choice to assign those first row values to 0. This means that the only non-null values on the first row of S are unity, and that they occur on the q eigenvector columns. This will prove helpful when calculating the expected frequency of occurrence of an n-gram by lowering the complexity of repeated multiplications of the first row of S by various powers of the matrix J.

2.5.2.1. The inversion of S.

The general inversion of an $(n+1) \times (n+1)$ matrix can be done in $O(n^3)$ through Gaussian elimination. To calculate only the last column of S^{-1} does not help, since the resulting system of $(n+1)$ equations still requires $O(n^3)$ operations to be solved by Gaussian elimination.

However, some specificities of our problem will again permit an improvement over this general complexity. When describing the calculation of the similarity matrix S, it became clear that the n_i occurrences of e_i on the main diagonal of the matrix M are matched by n_i column vectors whose last non-null values exactly correspond to the n_i positions of occurrence of e_i on the main diagonal of M.

This is equivalent to saying that S **is a column permutation of an upper-triangular matrix**. Let T be the upper-triangular matrix corresponding to the column permutation of S. We can calculate the vector x, equal to the same permutation of the last column of S^{-1}, by solving the triangular system of linear equations that follows $TT^{-1} = I_{n+1}$:

$$\begin{pmatrix} T_{1,1} & T_{1,2} & \dots & T_{1,n+1} \\ 0 & T_{2,2} & & T_{2,n+1} \\ \vdots & \ddots & \ddots & \vdots \\ 0 & \dots & 0 & T_{n+1,n+1} \end{pmatrix} \quad \overbrace{\begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_{n+1} \end{pmatrix}}^{x = \text{last column of } T^{-1}} = \overbrace{\begin{pmatrix} 0 \\ \vdots \\ 0 \\ 1 \end{pmatrix}}^{\text{last column of } I_{n+1}} .$$

The solution x is calculated by backward substitution:

$$\begin{cases} x_{n+1} &= \frac{1}{T_{n+1,n+1}} \\ x_n &= -\frac{1}{T_{n,n}}(T_{n,n+1}x_{n+1}) \\ \vdots & \\ x_1 &= -\frac{1}{T_{1,1}}(T_{1,2}x_2 + T_{1,3}x_3 + \dots + T_{1,n+1}x_{n+1}) \end{cases}$$

This way, the set of $(n+1)$ triangular linear equations can be solved in $O(n^2)$. It only remains to apply the reverse permutation to x to obtain the last column of S^{-1}.

Finding the permutation and its inverse are simple sorting operations. Thus, **the whole process of computing the last column of the matrix S^{-1} is** $O(n^2)$.

2.5.3. *Conclusion*

To obtain the final result, the probability of occurrence of the n-gram in a document of size l, it only remains to multiply the first row of S by J^l, and the resulting vector by the last column of S^{-1}. The second operation takes $(n+1)$ multiplications and n additions. It is thus $O(n)$.

The general multiplication of a vector of size $(n+1)$ by an $(n+1) \times (n+1)$ square matrix takes $(n+1)$ multiplications and n additions for each of the $(n+1)$ values of the resulting vector. This is thus $O(n^2)$. However, we can use yet another trick to improve this complexity. When we calculated the matrix S, we could assign the first row values of each column vector freely. We did it in such a way that the only non-null values on the first row of S are unity, and that they occur on the q eigenvector columns. Therefore, to multiply the first row of S by a column vector simply consists in the addition of the q terms of index equal to the index of the eigenvectors in S. That operation of order $O(q)$ needs to be repeated for each column of J^l. The multiplication of the first row of S by J^l is thus $O(nq)$.

The worst-case time complexity of the computation of the probability of occurrence of an n-gram in a document of size l is finally $\max\{O(lq), O(n^2)\}$. Clearly, if $l < n$, the document is smaller than the n-gram, and thus the probability of occurrence of the n-gram therein can immediately be said to be null. Our problem of interest is hence limited to $l \geq n$.

Therefore, following our technique, **an upper bound** of the complexity for computing **the probability of occurrence of an n-gram in a document of size l is** $O(ln)$. This is clearly better than directly raising M to the power of l, which is $O(ln^3)$, not to mention the computation of the exact mathematical Formula [2], which is only achieved in $O(ln^{l-n})$.

3. The Expected Frequency of an n-Words Sequence

Now that we have defined a formula to calculate the probability of occurrence of an n-gram in a document of size l, we can use it to calculate the expected document frequency of the n-gram in the whole document collection D. Assuming the documents are mutually independent, the expected frequency in the document collection is the sum of the probabilities of occurrence in each document:

$$Exp\_df(A_1 \rightarrow \cdots \rightarrow A_n, D) = \sum_{d \in D} p(A_1 \rightarrow \cdots \rightarrow A_n, |d|),$$

where $|d|$ stands for the number of word occurrences in the document d.

Naive Computational Complexity
We can compute the probability of an n-gram to occur in a document d in $O(|d|n)$. A separate computation and summation of the values for each document can thus be computed in $O(\sum_{d \in D} |d|n)$.

Better Computational Complexity
We can achieve better complexity by summarizing everything we need to calculate and organizing the computation in a sensible way. Let $L = \max_{d \in D} |d|$ be the size of the longest document in the collection and $|D|$, the number of documents in D. We first need to raise the Jordan matrix J to the power of every distinct document length, and then to multiply the (at worst) $|D|$ distinct matrices by the first row of S and the resulting vectors by the last column of its inverse S^{-1}.

The matrix S and the last column of S^{-1} need to be computed only once, and as we have seen previously, this is achieved in $O(n^2)$, whereas the $|D|$ multiplications by the first row of S are done in $O(|D|nq)$. It now remains to find the computational complexity of the various powers of J.

We must first raise each eigenvalue e_i to the power of L, which is an $O(Lq)$ process. For each document $d \in D$, we obtain all the terms of $J^{|d|}$ by $(n+1)$ multiplications of powers of eigenvalues by a set of combinatorial coefficients computed in $O(\max_{block})$. The total number of such multiplications is thus $O(|D|n)$, an upper bound for the computation of all combinatorial coefficients. The worst case time complexity for computing the set $\{\ J^{|d|} \mid d \in D\}$, is then $\max\{O(|D|n), O(Lq)\}$.

Finally, **the computational complexity for calculating the expected frequency of an n-gram in a document collection D is** $\max\{O(|D|nq), O(Lq)\}$, where q is the number of words in the n-gram having a distinct probability of occurrence, and L is the size of the longest document in the collection. The improvement is considerable, compared to the computational complexities of the more naive techniques, in $O(\sum_{d \in D} |d|n^{l-n})$ and $O(\sum_{d \in D} |d|n^3)$.

4. Direct evaluation of lexical cohesive relations

In this section, we will introduce an application of the expected document frequency that fills a gap in information retrieval. We propose a direct technique, language and domain-independent, to rank a set of phrasal descriptors by their interestingness, *regardless of their intended use*.

The evaluation of lexical cohesion is a difficult problem. Attempts at direct evaluation are rare, simply due to the subjectivity of any human assessment, and to the wide acceptance that we first need to know what we want to do with a lexical unit before being able to decide whether or not it is relevant for that purpose. A common application of research in lexical cohesion is lexicography, where the evaluation is carried out by human experts who simply look at phrases to assess them as good or bad.

This process permits scoring the extraction process with highly subjective measures of precision and recall. However, a linguist interested in the different forms and uses of the auxiliary "to be" will have a different view of what is an interesting phrase than a lexicographer. What a human expert judges as uninteresting may be highly relevant to another.

Hence, most evaluation has been indirect, through question-answering, topic segmentation, text summarization, and passage or document retrieval (Vechtomova, 2005). To pick the last case, such an evaluation consists in trying to figure out which are the phrases that permit to improve the relevance of the list of documents returned. A weakness of indirect evaluation is that it hardly shows whether an improvement is due to the quality of the phrases, or to the quality of the technique used to exploit them. Moreover, text retrieval collections often have a relatively small number of queries, which means that only a small proportion of the phrasal terms will be used at all. This is a strong argument against the use of text retrieval as an indirect way to evaluate the quality of a phrasal index, initially pointed out by Fox (Fox, 1983).

There is a need to fill the lack of a general purpose direct evaluation technique, one where no subjectivity or knowledge of the domain of application will interfere. Our technique permits exactly that, and we will now explain how.

4.1. *Hypothesis testing*

A general approach to estimate the interestingness of a set of events is to measure their statistical significance. In other words, by evaluating the validity of the assumption that an event occurs only by chance (the *null hypothesis*), we can decide whether the occurrence of that event is interesting or not. If a frequent occurrence of a multi-word unit was to be expected, it is less interesting than if it comes as a surprise.

To estimate the quality of the assumption that an n-gram occurs by chance, we need to compare its (by chance) expected frequency and its observed frequency. There exists a number of tests, extensively described in statistics textbooks, even so in the specific context of natural language processing (Manning *et al.,* 1999). In this paper, we will base our experiments on the *t-test*:

$$t = \frac{Obs\_df(A_1 \rightarrow \cdots \rightarrow A_n, D) - Exp\_df(A_1 \rightarrow \cdots \rightarrow A_n, D)}{\sqrt{|D| Obs\_DF(A_1 \rightarrow \cdots \rightarrow A_n)}}$$

4.2. *Example of non-contiguous lexical units: MFS*

Maximal Frequent Sequences (MFS) are word sequences built with an unlimited gap, no stoplist, no POS analysis and no linguistic filtering. They are defined by two characteristics; A sequence is said to be *frequent* if its document frequency is higher than a given threshold. It is *maximal*, if there exists no other frequent sequence that contains it. $MineMFS$ (Ahonen-Myka *et al.,* 2005) is a method that combines

breadth-first and depth-first search so as to extract all the MFSs of a document collection.

The nature of MFSs makes them correspond very well to our technique, since the extraction algorithm provides each extracted MFS with its document frequency. To compare the observed frequency of MFSs to their expected frequency is thus especially meaningful, and it will permit to order a set of MFSs based on their statistical significance.

4.3. *Experiments*

4.3.1. *Corpus*

For experiments we used the publicly available Reuters-21578 newswire collection (Reuters-21578, 1987), which originally contains about $19,000$ non-empty documents. We split the data into $106,325$ sentences. The average size of a sentence is 26 word occurrences, while the longest sentence contains 260.

Using $MineMFS$ with a minimum frequency threshold of 10, we obtained $4,855$ MFSs, distributed in $4,038$ 2-grams, 604 3-grams, 141 4-grams, and so on. The longest sequences had 10 words.

The expected document frequency and the t-test of all the MFSs were computed in 31.425 seconds on a laptop with a 1.40 Ghz processor and 512Mb of RAM. We used an implementation of a simplified version of the algorithm that does not make use of all the improvements presented in this paper.

4.3.2. *Results*

Table 1 shows the overall best-ranked MFSs. The number in parenthesis after each word is its frequency. With Table 2, we can compare the best-ranked bigrams of frequency 10 to their worst-ranked counterparts (which are also the worst-ranked n-grams overall), noticing a difference in quality that the observed frequency alone does not reveal.

It is important to note that our technique permits to rank longer n-grams amongst pairs. For example, the best-ranked n-gram of a size higher than 2 lies in the 10^{th} position: *"chancellor exchequer nigel lawson"* with t-test value 0.02315, observed frequency 57, and expected frequency $0.2052e-07$.

In contrast to this high-ranked 4-gram, the last-ranked n-gram of size 4 occupies the $3,508^{th}$ position: *"issuing indicated par europe"* with t-test value 0.009698, observed frequency 10, and expected frequency $22.25e{-}07$. Now, let us attempt to compare our ranking, based on the expected document frequency of discontiguous word sequences to a ranking obtained through a well-known technique. We must first underline that such a "ranking comparison" can only be empirical, since our standpoint

t-test	n-gram	expected	observed
.0311	los(127) angeles(109)	.0809	103
.0282	kiichi(88) miyazawa(184)	.0946	85
.0274	kidder(91) peabody(94)	.0500	80
.0267	morgan(382) guaranty(93)	.2073	76
.0249	latin(246) america(458)	.6567	67
.0243	orders(516) orders(516)	1.550	66
.0243	leveraged(85) buyout(145)	.0720	63
.0240	excludes(350) extraordinary(392)	.7995	63
.0239	crop(535) crop(535)	1.666	64
.0232	chancellor(120) exchequer(100) nigel(72) lawson(227)	2.052e-8	57

Table 1. *Overall 10 best-ranked MFSs*

t-test	n-gram	expected	observed
9.6973-3	het(11) comite(10)	.6430-3	10
9.6972-3	piper(14) jaffray(10)	.8184-3	10
9.6969-3	wildlife(18) refuge(10)	.0522-3	10
9.6968-3	tate(14) lyle(14)	.1458-3	10
9.6968-3	g.d(10) searle(20)	.1691-3	10
8.2981-3	pacific(502) security(494)	1.4434	10
8.2896-3	present(496) intervention(503)	1.4521	10
8.2868-3	go(500) go(500)	1.4551	10
8.2585-3	bills(505) holdings(505)	1.4843	10
8.2105-3	cents(599) barrel(440)	1.5337	10

Table 2. *The 5 best- and worst-ranked bigrams of frequency 10*

is to focus on general-purpose descriptors. It is therefore, by definition, impossible to assess descriptors individually as interesting and not.

Evaluation through Pointwise Mutual Information

The choice of an evaluation technique to oppose is rather restricted. A computational advantage of our technique is that it does not use distance windows or the distances between words. As a consequence, no evaluation technique based on the mean and variance of the distance between words can be considered. Smadja's z-test (Smadja, 1993) is then out of reach. Another option is to apply a statistical test, using a different technique to calculate the expected frequency of the word sequences. We decided to opt for pointwise mutual information, as applied to collocation discovery by Church and Hanks (Church *et al.,* 1990), an approach we already mentioned in the introduction of this paper.

Bigram	Frequency	Mutual Information
het(11) comite(10)	10	17.872
corpus(12) christi(12)	12	17.747
kuala(14) lumpur(13)	13	17.524
piper(14) jaffray(10)	10	17.524
cavaco(15) silva(11)	11	17.425
lazard(16) freres(16)	16	17.332
macmillan(16) bloedel(13)	13	17.332
tadashi(11) kuranari(16)	11	17.332
hoare(15) govett(14)	13	17.318
ortiz(16) mena(14)	13	17.225

Table 3. *Mutual Information: the 10 best bigrams.*

The rank of all word pairs is obtained by comparing the frequency of each pair to the probability that both words occur together by chance. Given the independence assumption, the probability that two words occur together by chance is the multiplication of the probability of occurrence of each word. And pointwise mutual information is thus calculated as follows:

$$I(w_1, w_2) = log_2 \frac{P(w_1, w_2)}{P(w_1)P(w_2)}.$$

If $I(w_1, w_2)$ is positive, and thus $P(w_1, w_2)$ is greater than $P(w_1)P(w_2)$, it means than the words w_1 and w_2 occur together more frequently than chance. In practice, the mutual information of all the pairs is greater than zero, due to the fact that the maximal frequent sequences that we want to evaluate are already a selection of statistically remarkable phrases.

As stated by Fano (Fano, 1961), the intrinsic definition of mutual information is only valid for bigrams. Table 3 presents the best 10 bigrams, ranked by decreasing mutual information. Table 4 shows the 5 best- and worst-ranked bigrams of frequency 10 (again, the worst ranked bigrams of frequency 10 are also the worst ranked overall). We can observe that, for the same frequency, the rankings are very comparable. Where our technique outperforms mutual information is in ranking together bigrams of different frequencies. It is actually a common criticism against mutual information, to point out that the score of the lowest frequency pair is always higher, with other things equal (Manning *et al.,* 1999). For example, the three best-ranked MFS in our evaluation, *"Los Angeles", "Kiichi Miyazawa"* and *"Kidder Peabody"*, which are among the most frequent pairs, rank only 191^{st}, 261^{st} and 142^{nd} with mutual information (out of $4,038$ pairs).

Mutual information is not defined for n-grams of a size longer than two. Other techniques are defined, but they usually give much higher scores to longer n-grams, and in practice, rankings are successions of decreasing size-wise sub-rankings. A noticeable exception is the measure of mutual expectation (Dias *et al.,* 2000).

Bigram	Frequency	Mutual Information
het(11) comite(10)	10	17.872
piper(14) jaffray(10)	10	17.524
wildlife(18) refuge(10)	10	17.162
tate(14) lyle(14)	10	17.039
g.d(10) searle(20)	10	17.010
pacific(502) security(494)	10	6.734
present(496) intervention(503)	10	6.725
go(500) go(500)	10	6.722
bills(505) holdings(505)	10	6.693
cents(599) barrel(440)	10	6.646

Table 4. *Mutual Information: the 5 best and worst bigrams of frequency 10.*

Compared to the state of the art, the ability to evaluate n-grams of different sizes on the same scale is one of the major strengths of our technique. Word sequences of different size are ranked together, and furthermore, the variance in their rankings is wide. While most of the descriptors are bigrams ($4,038$ out of $4,855$), the 604 trigrams are ranked between the 38^{th} and $3,721^{st}$ overall positions. For the 141 4-grams, the position range is $10-3,508$.

5. Conclusion

The main contribution of this article is a novel technique for calculating the probability of occurrence of any given non-contiguous lexical cohesive relation (actually, of any n sequential items of any data type), that is, the probability that those words occur, and that they occur in a given order, regardless of which and how many other words may occur between them. We found a Markov representation of the problem and exploited the specificities of that representation to reach linear computational complexity. The initial order of complexity of $O(ln^{l-n})$ was brought down to $O(ln)$. The technique is mathematically sound, computationally efficient, and it is fully language- and application-independent.

We further described a method that compares observed and expected document frequencies through a statistical test as a way to give a direct numerical evaluation of the intrinsic quality of a multi-word unit (or of a set of multi-word units). This technique does not require the work of a human expert, and it is fully language- and application-independent. It permits to efficiently compare n-grams of different length on the same scale.

A weakness that our approach shares with most language models is the assumption that terms occur independently from each other. In the future, we hope to present more advanced Markov representations that will permit to account for term dependency.

6. References

Ahonen-Myka H., Doucet A., « Data Mining Meets Collocations Discovery », *Inquiries into Words, Constraints and Contexts, Festschrift in the Honour of Kimmo Koskenniemi*, CSLI Publications, Center for the Study of Language and Information, University of Stanford, p. 194-203, 2005.

Choueka Y., Klein S. T., Neuwitz E., « Automatic Retrieval of frequent idiomatic and collocational expressions in a large corpus », *Journal for Literary and Linguistic computing*, vol. 4, p. 34-38, 1983.

Church K. W., Hanks P., « Word association norms, mutual information, and lexicography », *Computational Linguistics*, vol. 16, n° 1, p. 22-29, 1990.

Coppersmith D., Winograd S., « Matrix multiplication via arithmetic progressions », *STOC'87: Proceedings of the 19th annual ACM conference on Theory of computing*, p. 1-6, 1987.

Dias G., Guilloré S., Bassano J.-C., Pereira Lopes J. G., « Extraction Automatique d'Unités Complexes: Un Enjeu Fondamental pour la Recherche Documentaire », *Traitement Automatique des Langues*, vol. 41, n° 2, p. 447-472, 2000.

Fano R. M., *Transmission of Information: A Statistical Theory of Information*, MIT Press, Cambridge MA, 1961.

Feller W., *An Introduction to Probability Theory and Its Applications*, vol. 1, third edn, Wiley Publications, 1968.

Fox E. A., Some Considerations for Implementing the SMART Information Retrieval System under UNIX, Technical Report n° TR 83-560, Department of Computer Science, Cornell University, Ithaca, NY, September, 1983.

Frantzi K., Ananiadou S., Tsujii J.-i., « The C-value/NC-value Method of Automatic Recognition for Multi-Word Terms », *ECDL '98: Proceedings of the Second European Conference on Research and Advanced Technology for Digital Libraries*, Springer-Verlag, p. 585-604, 1998.

Horn R. A., Johnson C. R., *Topics in matrix analysis*, Cambridge University Press, New York, NY, USA, 1994.

Manning C. D., Schütze H., *Foundations of Statistical Natural Language Processing*, second edn, MIT Press, Cambridge MA, 1999.

Mitra M., Buckley C., Singhal A., Cardie C., « An analysis of statistical and syntactic phrases », *Proceedings of RIAO97, Computer-Assisted Information Searching on the Internet*, p. 200-214, 1987.

Noble B., Daniel J. W., *Applied Linear Algebra*, second edn, Prentice Hall, p. 361-367, 1977.

Reuters-21578, « Text Categorization Test Collection, Distribution 1.0 », 1987. http://www.daviddlewis.com/resources/testcollections/reuters21578.

Smadja F., « Retrieving Collocations from Text: Xtract », *Journal of Computational Linguistics*, vol. 19, p. 143-177, 1993.

Tanaka T., Villavicencio A., Bond F., Korhonen A., (eds). *Second ACL Workshop on Multiword Expressions: Integrating Processing*, 2004.

Vechtomova O., « The Role of Multi-Word Units in Interactive Information Retrieval », *Proceedings of the 27th European Conference on Information Retrieval, Santiago de Compostela, Spain*, p. 403-420, 2005.