Fast extraction of discontiguous sequences in text: a new approach based on maximal frequent sequences.

Antoine Doucet, Helena Ahonen-Myka

Department of Computer Science
PO Box 68 (Gustaf Hällströminkatu 2B)
FI-00014 University of Helsinki

Abstract

In this paper, we present a new technique for the extraction of discontiguous sequential descriptors from text. We are able to form word sequences without any restriction on their size or on the distance between their components. Based on the concept of a maximal frequent sequence (MFS), our approach allows for the extraction of compact text descriptors of quality in a more efficient manner than other previously known techniques. It further scales up to document collections of virtually any size, when other approaches normally fail for collections large enough. After a review of the related work and the presentation of our approach, $MFS\_MineSweep$, we introduce measures of the quality and quantity of information in a set of sequential descriptors representing a document collection. We finally present experiments whose results demonstrate the real-life applicability and superiority of the proposed method.

1. Introduction

Most document models rely on single word terms. A trend to improve this fact is to extract and use multi-word units or phrases. The problem of detecting such cohesive lexical units is a very difficult one as the number of possible word compounds in text is enormous and a vast majority of them do not constitute true multi-word units.

Exhaustive approaches are clearly exponential and researchers have therefore always had to place several restrictions on the search space, such as a maximal phrase length (Church and Hanks, 1990), fixed relative positions (Dias, 2003), or linguistic filtering (Mitra et al., 1987). Maximal frequent sequences (MFS) (Ahonen-Myka and Doucet, 2005) are a type of phrases that presents the advantage to remove most of these constraints. They can be formed of words separated by any distance and they are not restricted in length. MFSs as content descriptors present two major strengths. Firstly, they offer a very compact description: with a maximal phrase length of 8 words, it takes thousands of 8-sequences to replace a single phrase of length 20 (precisely, $\binom{20}{8} = 125\,970$ such sequences). Secondly, they do not require any knowledge about the data at hand. They can therefore be applied to documents in any domain and written in any language. We believe this is a very strong point when billions of heterogeneous documents coexist in real-world document collections such as the World Wide Web.

This paper presents two contributions. The first one is $MFS\_MineSweep$, a technique relying on MFSs to extend the extraction of compact sequential descriptors to very large document collections. The resulting phrasal document descriptions are far more exhaustive and have a higher discriminative power. The second contribution is the introduction of metrics to measure the quality and density of a sequence-based representation of a document collection.

In the next section, we will formally define the concept of a Maximal Frequent Sequence (MFS) and review the current state of the art of work addressing the problem of their extraction We will then present $MineMFS$, the current best-performing algorithm for the extraction of MFSs in text and expose some of its limitations. In Section 3, we will introduce our contribution, $MFS\_MineSweep$, a technique that relies upon $MineMFS$ to extract relevant document descriptors from document collections of virtually any size. We will then introduce a set of metrics for the evaluation of a sequence-based document description (Section 4). Before concluding the paper, we will present and discuss our experiments in Section 5.

2. Maximal Frequent Sequence (MFS)

In this section, we will introduce the concept of a Maximal Frequent Sequence in further detail (Ahonen-Myka and Doucet, 2005). We will then overview the data mining techniques that aim at the extraction of sequential patterns, and particularly those that permit to extract MFSs.

2.1. Definitions

Definition 1 *A sequence $p = a_1 \cdots a_k$ is a* subsequence *of a sequence q if all the items $a_i, 1 \leq i \leq k$, occur in q and they occur in the same order as in p. If p is a subsequence of q, we also say that p* occurs *in q and that q is a* supersequence *of p.*

For instance, the sequence *"unfair practices"* can be found in all of the three sentences in Figure 1.

Definition 2 *A sequence p is* frequent *in a set of fragments S if p is a subsequence of at least σ fragments of S, where σ is a given frequency threshold.*

If we assume that the frequency threshold is 2, we can find the following frequent sequences in our sample set of sentences: *"congress retaliation against foreign unfair trade practices"* and *"unfair practices"* (Fig. 1).

Definition 3 *A sequence p is a* maximal frequent (sub)sequence *in a set of fragments S if there does not exist any sequence p' in S such that p is a subsequence of p' and p' is frequent in S.*

1. The **Congress** subcommittee backed away from mandating specific **retaliation against foreign** countries for **unfair** foreign **trade practices**.

2. He urged **Congress** to reject provisions that would mandate U.S. **retaliation against foreign unfair trade practices**.

3. Washington charged France, West Germany, the U.K., Spain and the EC Commission with **unfair practices** on behalf of Airbus.

Figure 1: A set of sentences from the Reuters-21578 collection (1987).

In our example, the sequence *"unfair practices"* is not maximal, since it is a subsequence of the frequent sequence *"congress retaliation against foreign unfair trade practices"*. This latter sequence is maximal.

With this simple example, we already get a glimpse of the compact descriptive power of MFSs. Should we be restricted to word pairs, the 7-gram *"congress retaliation against foreign unfair trade practices"* would need to be replaced by 21 bigrams. With MFSs, we can obtain a very compact representation of the regularities of text. The rest of this section will focus on the problem of their efficient extraction in a document collection.

2.2. Related Work

Given a document collection and a minimal frequency threshold, the naïve approach is to go through the document collection, collect each frequent word, and use the set of all frequent words to produce candidate word pairs (bigrams) and retain only the frequent ones. The process of forming and counting the frequency of (n+1)-gram candidates from the set of all frequent n-grams can be repeated iteratively as long as frequent (n+1)-grams are found. To obtain the set of all MFSs, it remains to remove every frequent sequence that is a subsequence of another frequent sequence. But this approach is clearly computationally inefficient.

2.2.1. Sequential Pattern Mining

Agrawal and Srikant (1995) introduced the problem of *mining sequential patterns* as an advanced subtask of data mining, where typical data consists of customer transactions, that is, database entries keyed on a *transaction id* and each consisting of a *customer id* associated to the list of items that she bought in this very transaction. The problem of mining sequential patterns is an advanced version of that of the extraction of interesting *item sets*. But in sequential pattern mining, we also aim to exploit the fact that the transaction entries of the databases include a time field that permits to sort the transactions in chronological order and even know the time interval (or distance) that separates them. A motivating example of a sequential pattern, from (Agrawal and Srikant, 1995), would be that customers typically rent the movie "Star Wars", then "The Empire Strikes Back", and finally "The Return of the Jedi".

Agrawal and Srikant (1995) presented an improvement of the naïve approach that benefits of an intermediary pruning step to remove all (n+1)-gram candidates that contain at least one non-frequent n-gram. This permits to avoid a number of useless frequency counts. Most approaches are fueled by the same idea of pruning a number of "candidate frequent sequences", to avoid costly frequency counts.

Zaki (2001) presented $SPADE$, an advanced technique for the discovery of sequential patterns. Its architecture relies on a vertical database that fastens frequency counts and a lattice-theoretic approach permits to reduce the search space. Unfortunately, the main weakness of $SPADE$ is that it still enumerates all the candidate sequences by forming candidate (n+1)-sequences through the combination of each two n-sequences. DFS_Mine (Tsoukatos and Gunopulos, 2001) was subsequently designed to try to discover n-sequences without enumerating all the frequent sequences of length (n-1). This is done by storing two lists, containing "minimal non-frequent sequences" (because their supersequences are necessarily infrequent) and "maximal frequent sequences" (because their subsequences are necessarily frequent). A significant number of frequency counts can then be avoided. The problem with DFS_Mine is that the candidate (n+1)-sequences are formed by combining an n-sequence with the items of the database. While this may function with spatiotemporal data, the presented application of DFS_Mine, where the number of items is low, this is not reasonable for text, where the number of items (words) can be enormous.

2.2.2. Sequential Patterns and Text

The key particularity of text as a sequential data type is the number of items. For instance, the vocabulary of the widely known *Brown corpus* contains $50,406$ distinct words, whereas, e.g., biosequences have a very limited vocabulary: there are only 20 amino acids, and only 4 molecules containing nitrogen in DNA and RNA (A, C, G, and T). Another particularity of text is that the distribution of words is skewed. There is a small number of words that are very frequent, whereas the majority of words are infrequent. The words with moderate frequency are usually considered the most interesting and most informative.

These special characteristics of textual data have a strong influence on the discovery of interesting sequences in text. All the breadth-first, bottom-up approaches are failing quickly for a number of reasons. They permit pruning but require to keep in memory all the subsequences of two distinct lengths. They further generate a large number of candidates whose frequency is slow to count. Depth-first search takes less memory, but the number of items (words) to be intersected with a given sequence is prohibitive.

2.3. Sequential Pattern Mining in Text: $MineMFS$

MineMFS (Ahonen-Myka and Doucet, 2005) is a method combining breadth-first and depth-first search that is particularly well-suited for text. It extracts MFSs of any length, i.e., also very long sequences, and it allows an unrestricted gap between words of the sequence. In practice, however, text is usually divided into sentences or paragraphs, which indirectly restricts the length of sequences, as well as the maximal distance between two words of a se-

quence. The constraints used in the method are minimum and maximum frequency. Hence, words that are less (respectively, more) frequent than a minimum (respectively, maximum) frequency threshold are removed.

Algorithm. As for DFS_Mine, an important idea in $MineMFS$ is to compute frequent (n+1)-sequences without enumerating all the frequent n-sequences. It relies on a set of "n-gram seeds", initialized with the set of all frequent bigrams. The main idea is to pick an n-gram seed and try to combine it with other grams in a greedy manner, i.e., as soon as the n-gram seed is successfully expanded to a longer frequent sequence, other expansion alternatives are not checked, but only that longer frequent sequence is tentatively expanded again. This expansion procedure is repeated until the longer frequent sequence at hand can only be expanded to infrequent sequences. This sequence is maximal. When all the n-gram seeds have been processed, those that cannot be used to form a new maximal frequent sequence of size more than n are pruned. The remaining ones are joined to produce candidate (n+1)-gram seeds that will be used in a new iteration of the process. This process is repeated until no new maximal frequent sequence can be discovered.

Strengths. A main strength of $MineMFS$ versus DFS_Mine is the fact that the choice of items that may be inserted to expand an n-gram is restricted to the other non-pruned frequent n-grams. Whereas in DFS_Mine, an n-gram is expanded by trying to insert every (or most) frequent word, which is too costly for textual data. Further sophisticated pruning techniques permit restricting the depth-first search, which means only a few alternatives need to be checked to try to expand a sequence, despite the large vocabulary size.

Limitations. Even though the use of minimal and maximal frequency thresholds permits to reduce the burstiness of word distribution, it also causes the miss of a number of truly relevant word associations. For large enough collections, the $MineMFS$ process fails to produce results, unless excessive minimal and maximal frequencies are decided upon, in which case the set of MFSs produced is small and contains mostly non-interesting descriptors. One reason may be the pruning step, which runs through the set of n-grams and compares each two of them that may form an (n+1)-gram, by checking if a new item can be added between every two adjacent words. The number of possible positions of insertion shall be problematic.

3. Partitioning the Collection to Approximate the MFS set efficiently

We have seen that $MineMFS$ fails to extract the MFS set of a sufficiently large document collection. In this section, we will introduce *MFS_MineSweep*, a technique to decompose a collection of documents into several disjointed subcollections, small enough so that the MFS set of each subcollection can be extracted efficiently. Joining all the sets of MFSs, we obtain an approximate of the maximal frequent sequence set for the full collection. $MFS\_MineSweep$ permits extracting more and sharper descriptors from document collections of virtually any size.

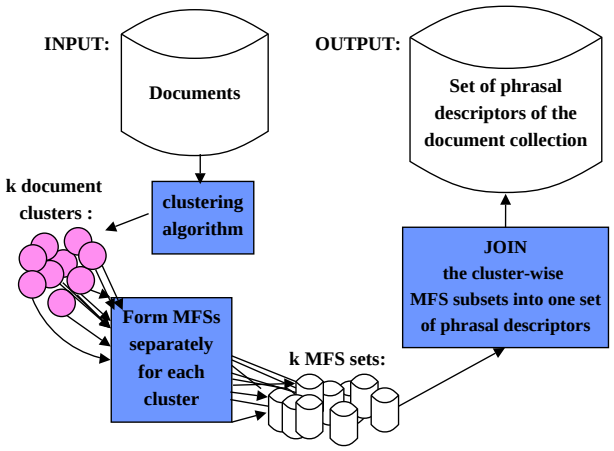

Figure 2: The different phases of $MFS\_MineSweep$.

Its main drawback is the loss of the maximality property, producing a less compact set of content descriptors.

3.1. Description and Claims

Our approach relies on the idea to partition the document collection into a set of homogeneous subcollections. The initial motivation to do this is that $MineMFS$ does not produce any result at all for sufficiently large document collections. Figure 2 describes the steps of $MFS\_MineSweep$. In the first phase, we apply $MineMFS$ on a number of disjoint subcollections, so as to obtain an MFS set corresponding to each subcollection. The second step is to gather the MFS sets of each subcollection to form a set of content descriptors for the whole collection. This gathering operation mainly consists in appending the sets of MFSs, as there is no clear way to join a sequence (maximal frequent in a subcollection) to its subsequence (maximal frequent in another). Only identical sequences can be merged. Thus, the maximality property is lost, and therefore, the content description of our pre-partitioning technique is always less or equally compact to that of the MFSs of the whole collection.

With this technique, we make two main claims that we will try to confirm or disprove in the evaluation. The main motivation for developing $MFS\_MineSweep$ is to efficiently obtain a more detailed description of the document collection (*Hypothesis H1*), as we can use looser frequency thresholds. This is easily understood by thinking of an extreme case; if a collection of $|D|$ documents is split into $|D|$ subcollections of size 1 and the minimal frequency is 1, we can obtain the corresponding sets of MFS instantly: each MFS set contains only one sequence of frequency 1, the unique document in the corresponding subcollection. No information is lost, but the content description is probably too large.

Our second main claim is about the optimal way to form the disjointed subcollections. We conjecture that more consistent subcollections permit to obtain better descriptors (*Hypothesis H2*). The main reason of this train of thought relies on the fact that a collection made of similar documents will contain more interesting MFSs than a collection made of dissimilar documents. Again, thinking of extreme

- d_1: Mary had a little lamb whose fleece was white as snow.
- d_2: A radio station called Sputnik broadcasts Russian programs in Saint-Petersburg and Helsinki. It was named after the first satellite ever launched.
- d_3: History changed on October 4, 1957, when the Soviet Union successfully launched Sputnik I. The world's first artificial satellite was about the size of a basketball, weighed only 183 pounds, and revolved around the Earth in about 98 minutes.
- d_4: Everywhere that Mary went, her lamb was sure to go.

Figure 3: A collection of four documents.

cases makes this point easier to see, as a collection where no two documents have a word in common will not contain any frequent sequences, except for the documents themselves (if the frequency threshold is 1).

For example, let us assume that we want to partition the collection of four documents presented in Figure 3 into 2 subcollections of 2 documents each, and use a minimal frequency of 2 for extracting MFSs from the subcollections. Only by clustering together the similar documents (d_1, d_4) and (d_2, d_3), will we obtain sequences of words, that is, *phrasal descriptors*. Those descriptors are: "Mary lamb was" for d_1 and d_4, and "Sputnik first satellite launched" for d_2 and d_3. Any other way to partition the collection produces an empty *phrasal description*.

4. Evaluating a Phrasal Text Description

To confirm or disprove the hypotheses we just made, we need measures to compare different sets of phrasal descriptors. Ideal metrics upon which to compare sets of descriptors should be able to evaluate two things: 1) the size of the phrasal text representation, and 2) the amount (and density) of information it contains.

In general, the problem of comparing two sets is not an easy one. A large quantity of work in the domains of document clustering and textual classification has proposed measures to compare different ways to partition document sets (Sebastiani, 2002). Unfortunately, we cannot exploit this work to solve our problem, because such techniques rely on the comparison of a given clustering (or classification) to a gold standard. In the general case of textual representation, without aiming at a specific application, there is no clear way to define a gold standard of the phrasal description of a document collection.

Fortunately, the problem we are facing here is a subproblem of the above. The sets we need to compare are indeed similar in nature. For example, a major difficulty in comparing general sequences would be the comparison of long grams to their subgrams. However, in the specific case where all the descriptors are MFS (either of the whole collection or of one of its subcollections), we can simplify the problem by normalizing each descriptor to a set of all its subpairs. This is because the unlimited distance allowed between any two words of an MFS ensures that the assertion "$ABCD$ is an MFS" implies "AB, AC, AD, BC, BD, and CD are frequent bigrams".

We can thus transform each set of phrasal descriptors into a set of comparable items, the frequent bigrams it contains. Let R_D be the phrasal description of a document collection D, and R_d be the corresponding set of phrases describing a document $d \in D$. We can write the corresponding set of word pairs as $bigrams(R_d)$. For $b \in bigrams(R_d)$, we also define df_b as the document frequency of the bigram b. Finally, we define the random variable X over the set $bigrams(R_d)$. For all $b \in bigrams(R_d)$:

$$p(X = b) = \frac{df_b}{\sum_{y \in \{ \bigcup_{d \in D} bigrams(R_d) \}} df_y},$$

where $\sum_{y \in \{\bigcup_{d \in D} bigrams(R_d)\}} df_y$ is the total number of bigram occurrences resulting from the phrasal description R_D. It can be thought of as the sample size.

Size of the representation of a document collection. The phrasal representation of a document collection can be seen as a set of associations between descriptive n-grams and documents. We define $|R_D|$ as the size of the phrasal representation R_D in a very intuitive way:

$$|R_D| = \sum_{d \in D} |R_d|.$$

Hence, $|R_D|$ is the number of document-phrase associations in the collection representation R_D.

Implied quantity of frequent bigrams in the representation. Several phrases may contain identical bigrams that represent the same document. To count the number of implied document-bigram associations permits to ignore redundant information stemming from the long descriptors. We shall therefore measure the quantity of information in the description with the number of document-bigram associations that correspond to the description R_D. This value is $bigram\_size(R_D)$, defined as follows:

$$bigram\_size(R_D) = \sum_{d \in D} |bigrams(R_d)|.$$

Hence, $bigram\_size(R_D)$ is the number of document-bigram associations stemming from the collection representation R_D.

Density of the description. To measure whether the description is loose or dense, we can use the two preceding metrics in a very simple way. By computing the ratio between the number of document-bigram associations in a document representation and its size, we obtain a relative measure of the number of document-bigram associations that can be avoided with longer n-grams:

$$Density(R_D) = \frac{bigram\_size(R_D)}{|R_D|}.$$

For example, a density value of 1.1 means that the bigram representation of R_D contains 10% more associations than the equivalent representation R_D. The higher $Density(R_D)$, the more storage space we save by using R_D instead of frequent pairs only.

Partitions (min,max)	Bigrams	Descriptors	Density
1 [MineMFS] (85,900)	147,000	126,000	1.17
2 (60-70, 900-1000)	841,000	819,000	1.03
3 (40, 650-715)	1,223,000	1,197,000	1.02
5 (25-30, 400-600)	1,605,000	1,574,000	1.02
10 (5-28, 72-350)	1,453,000	1,466,000	0.99
20 (10-28, 162-385)	1,643,000	2,555,000	0.64
50 (4-20, 60-208)	2,927,000	7,448,000	0.39
100 (3-45, 27-630)	3,570,000	11,038,000	0.32

Table 1: Reuters. Corresponding frequency ranges when every subcollection is computed within 4 and 5 minutes using $MineMFS$ directly and $MFS\_MineSweep$ on random partitions of size 2, 3, 5, 10, 20, 50 and 100.

Clusters (min,max)	Bigrams	Descriptors	Density
1 [MineMFS] (85,900)	147,000	126,000	1.17
2 (40-130, 660-1569)	554,000	568,000	0.97
3 (7-129, 180-1470)	449,000	498,000	0.90
5 (3-55, 47-1224)	995,000	993,000	1.00
10 (5-22, 58-671)	1,255,000	1,280,000	0.98
20 (3-14, 11-682)	1,767,000	1,904,000	0.93
50 (2-37, 5-289)	2,201,000	2,748,000	0.80
100 (2-28, 7-220)	2,932,000	4,597,000	0.64

Table 2: Reuters. Corresponding frequency ranges when every subcollection is computed within 4 and 5 minutes using $MineMFS$ directly and $MFS\_MineSweep$ on homogeneous partitions of size 2, 3, 5, 10, 20, 50 and 100.

5. Experiments and Results

5.1. Experiments and Results

In this section, we will detail and implement a set of experiments that permit to test our initial hypotheses. It is important to observe that the extraction of the set of MFSs is an independent process for each distinct subcollection. A profitable alternative is to run the extraction of the MFS sets in parallel, on distinct computers. The total running time is then the time of the slowest MFS set extraction, plus the time for splitting the document collection. The experiments are based on a set of desktops with a 2.80 Ghz processor and 1024Mb of RAM.

5.1.1. $MFS\_MineSweep$ extracts better, but less compact descriptors *(Hypothesis H1)*.

The claim of hypothesis *H1* is that we can extract more information using $MFS\_MineSweep$, although we then lose the maximality property, subsequently leading to a less compact description. To verify this, we experiment with the 16Mb Reuters-21578 newswire collection (Reuters-21578, 1987), which originally contains about $19,000$ non-empty documents. To place both techniques on equal grounds, we find a frequency range for every subcollection individually, such that the corresponding MFS extraction time is always between 4 and 5 minutes. This was achieved with a fairly simple heuristic, interrupting the process and decreasing the frequency range when the extraction was too slow, and increasing the frequency range after too fast an extraction. We then compare the resulting sizes, amounts and densities of information in Table 1. Note that every value resulting from a random partition into n subcollections is actually the average outcome of 10 distinct iterations of the random partitioning and evaluation process. Experiments have shown the variance is very small.

MFS_MineSweep outperforms MineMFS. Our first observation is that both the number of descriptors and the number of equivalent bigrams are always much higher for $MFS\_MineSweep$ than for $MineMFS$. These numbers increase with the number of partitions.

The description is less compact. Consequently, the density of the phrasal representations is decreasing with the number of subcollections. What we did not expect is that the density ratio goes down to values below 1, meaning that the number of equivalent bigrams is less than the number of phrasal descriptors. This steep density decrease expresses more than the loss of the maximality property. A lower density means that the number of descriptors is growing faster than the number of bigrams. When we split the collection into more disjoint subcollections, this means that more and more of the new descriptors we find are only new combinations of bigrams that we already found when we split the collection in less partitions. This sharp decrease in density is in fact an indication that the discriminative power of the phrasal description is peaking, and that further augmentations of the number of partitions will be comparatively less and less worthwhile.

The hypothesis *H1* is verified, an increase in the number of subcollections is followed by a more exhaustive, but less compact document description. We shall suspect that with homogeneous partitioning, a rise in the number of subcollections will increase their internal similarity and facilitate the discovery of new descriptors, with a strong discriminating power. This is to be verified in the following subsection.

5.1.2. The more homogeneous the subcollections, the better the descriptors *(Hypothesis H2)*.

To support *H2*, we use the same newswire collection and compare the size, amount and density of information obtained when splitting the collection into random and homogeneous subcollections. In the experiments, we formed homogeneous subcollections with the well-known k-means clustering algorithm. We used the publicly available clustering tool implemented by George Karypis at the University of Minnesota[1]. The phrasal descriptors resulting of homogeneous subcollections are evaluated in Table 2.

MFS_MineSweep outperforms MineMFS. What we had observed with random partitions is confirmed with homogeneous collections. We get a more exhaustive description of the document collection if we use $MFS\_MineSweep$ than if we use $MineMFS$ alone.

To permit an easier direct comparison, the quantities and densities of information obtained with random and homogeneous partitions are presented in Table 3.

Homogeneity provides better discrimination. We can observe that when the number of partitions rises, the density

[1]CLUTO, http://www-users.cs.umn.edu/∼karypis/cluto/

Partitions	Random	Homogeneous
2	841,000 (1.03)	554,000 (0.97)
3	1,223,000 (1.02)	449,000 (0.90)
5	1,605,000 (1.02)	995,000 (1.00)
10	1,453,000 (0.99)	1,255,000 (0.98)
20	1,643,000 (0.64)	1,767,000 (0.93)
50	2,927,000 (0.39)	2,201,000 (0.80)
100	3,570,000 (0.32)	2,932,000 (0.64)

Table 3: Reuters. Quantities and densities of information when every subcollection is computed within 4 and 5 minutes using $MFS\_MineSweep$ on random and homogeneous partitions of size 2, 3, 5, 10, 20, 50 and 100.

of the description resulting from homogeneous subcollections decreases slowly, whereas the steep is much sharper for random partitions. The fact that the description densities resulting from homogeneous collections remain nearly stable shows that there is room to improve the discriminative power of phrasal descriptions if we partition the document collection in even more clusters. The reason is simple. The descriptors extracted from random subcollections are ones that are present all over the collection. Splitting the collection into more subsets permits finding more of those frequent n-grams, formed by the same frequent words, but we reach a point where we only find combinations of the same frequent words originating from different subcollections. On the other hand, homogeneous subcollections permit gathering similar documents together, excluding non-similar documents. Hence, the frequency range can be adapted to extract the specifics of each subcollection. In the homogeneous case, increasing the number of subcollections permits embracing more specificities of the document collections, whereas in the random case, it only permits catching more descriptors of the same kind.

Clustering is safer. As opposed to random partitioning, clustering provides $guarantees$. It is more reliable, because it ensures result. The strength of random partitioning is it gives good results and permits MFS extraction in predictable times. But these facts are only true *on average*. The problem if we use random partitioning is that we should, in fact, run several iterations to protect ourselves from an "unlucky" draw. We mentioned earlier that running several random iterations increases the exposure to factors of difficult extraction. Another issue with averaging numerous iterations is practical. Assume document d was represented 3 times by $gram_A$, and 1 time by $gram_B$ and $gram_C$, what should be the average document description of d? Because the extraction of MFS sets from homogeneous subcollections is unique and needs to be done only once, it is generally less costly in the end.

6. Conclusion

In this paper, we introduced $MFS\_MineSweep$, a new solution for the extraction of compact phrasal descriptors from sequential data. We further defined metrics for the evaluation of such descriptions. We presented experiments on textual data that showed the capacity of $MFS\_MineSweep$ to extract a better description efficiently, by applying an MFS extraction algorithm on partitions of the document collection. Our approach permits to obtain a more exhaustive description faster. This improvement is strengthened by the possibility to run the costliest computations in parallel. We further established that the use of homogeneous partitions improves the quality of the description.

7. References

Rakesh Agrawal and Ramakrishnan Srikant. 1995. Mining sequential patterns. In Yu and Chen, editors, *11th International Conference on Data Engineering*, pages 3–14, Taipei, Taiwan. IEEE Computer Society Press.

Helena Ahonen-Myka and Antoine Doucet. 2005. Data mining meets collocations discovery. In *Inquiries into Words, Constraints and Contexts, Festschrift for Kimmo Koskenniemi*, pages 194–203. CSLI Publications, University of Stanford.

Kenneth W. Church and Patrick Hanks. 1990. Word association norms, mutual information, and lexicography. *Computational Linguistics*, 16(1):22–29.

Gaël Dias. 2003. Multiword unit hybrid extraction. In *Workshop on Multiword Expressions of the 41st ACL meeting. Sapporo. Japan.*

Mandar Mitra, Chris Buckley, Amit Singhal, and Claire Cardie. 1987. An analysis of statistical and syntactic phrases. In *Proceedings of RIAO97, Computer-Assisted Information Searching on the Internet*, pages 200–214.

Reuters-21578. 1987. Text categorization test collection.

Fabrizio Sebastiani. 2002. Machine learning in automated text categorization. *ACM Computing Survey*, 34(1):1–47.

Ilias Tsoukatos and Dimitrios Gunopulos. 2001. Efficient mining of spatiotemporal patterns. In *Proceedings of the 7th International Symposium on Advances in Spatial and Temporal Databases (SSTD)*, pages 425–442.

Mohammed J. Zaki. 2001. Spade: An efficient algorithm for mining frequent sequences. *Machine Learning*, 42(1-2):31–60.