An efficient any language approach for the integration of phrases in document retrieval

Antoine Doucet, Helena Ahonen-Myka

An efficient any language approach for the integration of phrases in document retrieval

Antoine Doucet · Helena Ahonen-Myka

Abstract In this paper, we address the problem of the exploitation of text phrases in a multilingual context. We propose a technique to benefit from multi-word units in *adhoc* document retrieval, whatever the language of the document collection. We present principles to optimize the performance improvement obtained through this approach. The work is validated through retrieval experiments conducted on Chinese, Japanese, Korean and English.

Keywords Multiword expressions · Document retrieval · Endogenous resources

1 Introduction

As opposed to words, the higher content specificity of phrases is a strong motivation for their extraction. The potential improvement that may be obtained by using phrases in document retrieval is supported by the behavior of users. In an analysis of the query log of the Excite search engine (more than 1.5 million queries), Williams et al. (2004) found that 8.4% of the queries contained explicit phrases, that is, they included at least two words enclosed in quotes. Even more interestingly, the authors found it beneficial to treat 40% of the queries without quotation marks as phrases rather than independent words. Consequently, there is no doubt that an efficient technique to use phrases may bring solid improvement to document retrieval applications. In a context such as the Web, where numerous languages coexist in enormous collections for which scaling is a key issue, it is crucial to use techniques

A. Doucet (✉)
Department of Computer Science, University of Caen, Caen, France
e-mail: Antoine.Doucet@info.unicaen.fr

H. Ahonen-Myka
Department of Computer Science, University of Helsinki, Helsinki, Finland
e-mail: Helena.Ahonen-Myka@cs.helsinki.fi

that are language independent. All our work is entirely corpus independent (and in particular, language independent), only relying on knowledge present *inside* the document collection being processed.

We will present related work on the use of phrases in document retrieval in Sect. 2. After that, Sect. 3 will present our contribution in full details. In Sect. 4, we will present the phrases to be used in the evaluation framework and give details on the way they are extracted from the document collections. The experimental framework of this paper is presented in Sect. 5 and its results are presented and discussed in Sect. 6, before the conclusions are drawn in Sect. 7.

2 Use of phrases in document retrieval

Work on the use of phrases in IR has been carried out for more than 25 years. Early results were very promising. However, unexpectedly, the constant growth of test collections caused a drastic fall in performance improvements. Salton et al. (1975) showed a relative improvement in average precision, measured over ten recall points, between 17 and 39%. Fagan (1989) reiterated the exact same experiments with a 10 MB collection and obtained improvements from 11 to 20%. This negative impact of the collection size was later confirmed by Mitra et al. (1997) over a 655 MB collection, improving the average precision by only one percent. Turpin and Moffat (1999) revisited and extended this work to obtain improvements between 4 and 6%.

In our opinion, this does not contradict the idea that adding document descriptors accounting for word order is likely to improve the performance of IR systems. One problem is the extraction of the phrases, while another difficult related problem is to find efficient ways to benefit from those phrases. This need was illustrated by work of Lewis (1992) and Vechtomova (2005), who both decided to involve human experts in the process. Both obtained small improvements, suggesting that the techniques to exploit the extracted phrases can also be improved.

There are various ways to exploit phrase descriptors. The most common technique is to consider phrases as supplementary terms of the vector space, using the same technique as for word terms. In other words, phrases are thrown into the bag of words. However, according to Strzalkowski and Carballo (1996), using a standard weighting scheme is inappropriate for mixed feature sets (such as single words and phrases). In such cases, the weight given to the least frequent phrases is considered too low. Their specificity is nevertheless often crucial in order to determine the relevance of a document, but while weighting phrasal matches, the interdependency between a phrase and its word components is another difficult issue to account for.

Vechtomova (2005) introduced an advanced matching technique. Its contribution was to address the problem of overlapping phrases, in a way that accounts for the relative positions of occurrence of the words they contain. The problem of overlapping phrases occurs for phrases of more than *two* words. Given a query phrase *ABC*, it is the question of how to evaluate a document that contains the phrase *ABC* and a document that contains the phrases *AB* and *BC* separately.

For each query phrase, a pass through the document collection is done, to retain every occurrence of terms of the query phrase and their original positions in the document.

Terms that form the keyphrase or one of its sub-phrases are gathered into so-called "windows". Each window is weighted by the inverted document frequency (idf) of the words that compose it and the distance that separated them originally:

$$\text{WindowWeight}(w) = \sum_{i \in w} \text{idf}_i \times \frac{n}{(\text{span} + 1)^p},$$

where i is a word occurring in the window w, n is the number of words in the window w, span is the distance between the ith and last word of the window, and p is a tuning parameter, arbitrarily set to 0.2. The score attributed to each document is calculated as the sum of the weights of the phrases it contains, where the weight of a phrase a in a document is defined as follows:

$$\text{PhraseWeight}(a) = \frac{(k+1) \times \sum_{w=1}^{n} \text{WindowWeight(w)}}{k \times \text{NF} + n},$$

where n is the number of windows w extracted for the phrase a, k is a phrase frequency normalization factor, arbitrarily set to 1.2. and NF is a document length normalization factor:

$$\text{NF} = (1 - b) + b \times \frac{\text{DocLen}}{\text{AveDocLen}},$$

where DocLen and AveDocLen are the document length and the average document length in the corpus (number of words), and b is a tuning constant, set to 0.75.

A major drawback is the computational complexity of this process. In this method, there is no static phrase index that gives a phrasal representation of the document collection. It is only at query-time that a representation of the collection is built that only contains the terms of the query. Such heavy processing in response to a query is quite problematic, as users usually expect to obtain results promptly.

In practice, the method has only been used for re-ranking the 1,000 best documents returned to a query by a vector space model relying on single word features. The results demonstrate a performance improvement in terms of average precision, which is unfortunately not statistically significant. They also confirm a common observation when using phrases for document retrieval: compared to the use of single word features only, improvement is observed at high recall levels, while the impact is negative at lower levels.

In the following section, we will introduce a new technique for computing phrase-based document similarity. We will then apply it to document retrieval.

3 An advanced phrase-matching technique

3.1 Basic concepts of document retrieval

The task of document retrieval consists of selecting a set of documents in a collection, in response to a user's request. The user initially formulates her information need, as a question in natural language, for example, or as a set of

keywords or keyphrases. We refer to the formulation of an information need as a *topic*, following the TREC-terminology.[1]

A document retrieval system compares the topic to each document of the collection to obtain a document-wise similarity value [also called Retrieval Status Value (RSV)]. The documents are then ranked, topic- and RSV-wise, the documents with a higher RSV being considered more likely to answer to the user's information need.

3.2 Problem definition and goals

Problem definition. Given a set of sequences that describe the documents of a collection, how can we determine to what extent the sequence $p_1 \ldots p_n$, issued from the document collection, corresponds to the sequence $q_1 \ldots q_m$, found in a user query? And how can we subsequently rank the documents according to how well we think they answer to the query?

We propose an approach that consists in comparing a set of descriptive phrases extracted from the document collection, to a set of *keyphrases* from the query. Given a query, every document receives a reward for every sequence it contains that matches a *keyphrase* of the query. This bonus generally differs for each different phrase. Note that from here onwards, the term *keyphrase* will be used to refer to a phrase found in a query.

A base weight. The most informative lexical associations should notably be promoted, using statistical information such as term and inverted document frequency.

Longer matches are better matches. Further, it is natural to wish that longer matches should receive a higher reward. If a query contains the keyphrase "XML structured information retrieval", the most appropriate documents are those whose descriptors contain this exact sequence, followed by those containing a subsequence of size 3 (e.g., "structured information retrieval"), and finally by documents containing a subpair of the keyphrase (e.g., "structured information" or "information retrieval").

Adjacency should not be required. Clearly, a phrasal descriptor containing the pair "XML retrieval" has a relationship with the keyphrase "XML structured information retrieval". This illustrates the fact that natural language is richer in variety than only recurrent adjacent word sequences.

But adjacency is generally a stronger indicator. We should, however, bear in mind the general rule that the more distant two words are, the less likely they are to be related. And the degree to which the relatedness of two words is affected by distance certainly varies greatly with different languages.

Inverted usage. An extension of the previous comments about word adjacency is that we should also try to take into account the fact that words might as well occur in inverted order, while still not necessarily being adjacent. For example, a phrase "retrieval of XML" triggers interest with respect to the earlier keyphrase "XML structured information retrieval".

[1] Text Retrieval Conference, http://www.trec.nist.gov/.

Jones and Sinclair (1974) give the example of the pair "hard work", where throughout their document collection, the words "hard" and "work" are occurring together in arbitrary order, and with a variable distance between them. Of course, in English, not all collocations are this relaxed, and others are exclusively rigid, for example the pair "Los Angeles" is very unlikely to occur in a different order, or with other words inserted. They term those two types of collocations as *position dependent* and *position free* collocations. By attributing a positive score to matches and ignoring misses, we can get around this problem. If we look for phrasal document descriptors containing "Angeles Los" or for the occurrence of "Los" and "Angeles" separated by other words, and we fail to find any, it will not worsen the retrieval performance. Whereas finding that a document about "retrieval of XML" is relevant to a query about "XML retrieval" is evidently better than failing to observe it.

In the next subsection, we will introduce our approach to the problem. It aims at taking into account all the observations above in a sensible way.

3.3 Document score calculation

Our approach exploits and combines two complementary document representations. One is based on single word terms, in the vector space model, and the other is a phrasal description, taking the sequential nature of text data into account.

Once documents and queries are represented within those two models, a way to estimate the relevance of a document with respect to a query remains to be found. We must sort the document list with respect to each query, which is why we need to compute a *Retrieval Status Value (RSV)* for each document and query. Below, we will explain how we calculate two separate RSVs, one for a word features vector space model and one for our phrasal description.

The reason to compute an RSV value based on the word-feature vector space model in addition to a phrasal RSV is due to the fact that the latter may not be sufficiently discriminating. A query may for instance contain no keyphrases, and a document may be represented with no phrasal descriptor. However, there can of course be correct answers to such queries, and such documents may be relevant to some information needs. Also, all documents containing the same matching phrases get the same phrasal RSV. If the phrasal description is small, it is necessary to find a way to break ties. The cosine similarity measure based on word features is very appropriate for that.

To combine both RSVs into one single score, we must first make them comparable by mapping them to a common interval. To do so, we used *Max Norm*, as presented by Lee (1995), which permits to bring all positive scores within the range [0,1]:

$$\text{New Score} = \frac{\text{Old Score}}{\text{Max Score}}$$

Following this normalization step, we aggregate both RSVs using a linear interpolation factor λ representing the relative weight of scores obtained with each technique.

Fig. 1 Topic 47

```
<Keywords>
 "concurrency control"
 "semantic transaction management"
 "application" "performance benefit"
 "prototype" "simulation" "analysis"
</Keywords>
```

$$\text{Aggregated Score} = \lambda \cdot \text{RSV}_{\text{Word\_Features}} + (1 - \lambda) \cdot \text{RSV}_{\text{Phrasal}},$$

where details on the computation of both RSVs are given in the rest of this section.

The evidence of previous experiments with the INEX collection (Doucet and Ahonen-Myka 2004) showed good results with an intuitive weighting scheme: weighting the single word RSV with the number of distinct word terms in the query (let a be that number), and the phrasal RSV with the number of distinct word terms found in keyphrases of the query (let b be that number). Thus:

$$\lambda = \frac{a}{a+b}$$

For example, in Fig. 1, showing topic 47 of the INEX collection, there are 11 distinct word terms and 7 distinct word terms occurring in keyphrases. Thus, for this topic, we have $\lambda = \frac{11}{11+7} \approx 0.61$.

Word features RSV. This first document representation is a standard vector space model, of which all features are single words. It represents a baseline model that our goal is to improve by the addition of sequential information from our second document model.

The index term vocabulary W includes every word found in the document collection, without preselection. Further, the words are left in their original form, no lemmatization or stemming being performed. This guarantees generality, as this can be done in an equally simple way for document collections written in any language.

In our vector space model, each document is represented by a $||W||$-dimensional vector filled in with a weight standing for the importance of each word token with respect to the document. To calculate this weight, we use a term-frequency normalized version of term-weighted components, as described by Salton and Buckley (1988), that is:

$$\text{tfidf}_w = \frac{\text{tf}_\text{w} \cdot \log \frac{|\text{D}|}{\text{df}_\text{w}}}{\sqrt{\sum_{w_i \in W} \left(\text{tf}_{w_i} \cdot \log \frac{|D|}{\text{df}_{\text{w}_\text{i}}} \right)^2}}$$

where tf_w and df_w are the term and document frequencies of the word w, and $|D|$ is the total number of documents in the collection D.

The vector space model offers a very convenient framework for computing similarities between documents and queries. Among the number of techniques to compare two vectors, we chose cosine similarity because of its computational efficiency. By normalizing the vectors, which we do in the indexing phase, $\text{cosine}(\vec{d_1}, \vec{d_2})$ indeed simplifies to the vector product $(d_1 \cdot d_2)$.

We have already expanded on the weaknesses and the amount of information that such a simple model cannot catch. This is why we will complement this model with a phrasal one, bringing sequential information into the document model, and aiming to carry it on into document retrieval.

3.4 Phrasal RSV

Given a set of n-grams (keyphrases) is attached to each document, we ought to define a procedure to match a phrase describing a document and a keyphrase. Our approach consists in decomposing keyphrases of the query into key pairs. Each of these pairs is bound to a score representing its inherent *quantity of relevance*. Informally speaking, the quantity of relevance of a key pair tells how much it makes a document relevant to contain an occurrence of this pair. This value depends on a basic measure of the importance of the pair (its *base weight*, which can be its inverted document frequency, for example) combined with a number of modifiers, meant to take into account the distance between two words of a pair, to penalize their possible inverted usage, and so on.

3.4.1 Definitions

Let D be a document collection and $K_1 \dots K_m$ a keyphrase of size m. Let K_i and K_j be two words of $K_1 \dots K_m$. We define the quantity of relevance associated to the key pair K_iK_j as:

$$Q_{\text{rel}}(K_iK_j) = \text{Base\_Weight}(K_iK_j, D) \cdot \text{Integrity}(K_iK_j),$$

where Base_Weight(K_iK_j, D) represents the general importance of K_iK_j in the collection D. A possible measure of this kind is the statistical significance of the pair, or its specificity, measured in terms of inverted document frequency:

$$\text{idf}(K_iK_j, D) = \log\left(\frac{|D|}{\text{df}(K_iK_j)}\right),$$

3.4.2 Integrity modifier

When decomposing the keyphrase $K_1 \dots K_m$ into pairs, the *Integrity Modifier* of the key pair K_iK_j is defined as the combination of a number of modifiers:

$$\text{Integrity}(K_iK_j) = \text{adj}(K_iK_j) \cdot \text{inv}(K_iK_j) \cdot \text{dup}(K_iK_j).$$

3.4.3 Non-adjacency penalty

Adj(K_iK_j) is a score modifier meant to penalize key pairs formed from non-adjacent words. Let $d(K_i, K_j)$ be the distance between K_i and K_j, that is, the number of other words appearing in the keyphrase between K_i and K_j ($d(K_i, K_j) = 0$ means that K_i and K_j are adjacent). We define:

$$\text{adj}(K_iK_j) = \begin{cases} 1, & & \text{if } d(K_i,K_j) = 0 \\ \alpha_1, & 0 \le \alpha_1 \le 1, & \text{if } d(K_i,K_j) = 1 \\ \alpha_2, & 0 \le \alpha_2 \le \alpha_1 & \text{if } d(K_i,K_j) = 2 \\ \ldots & & \\ \alpha_{m-2}, & 0 \le \alpha_{m-2} \le \alpha_{m-3}, & \text{if } d(K_i,K_j) = m-2 \end{cases}$$

Accordingly, the larger the distance between the two words, the lower the quantity of relevance attributed to the corresponding pair. In the experiments, we set only a base value of non-adjacency penalty adj_pen that is raised to the power of the distance between the two words of the key pair. In other words, $\alpha_{d(K_i,K_j)} = \text{adj\_pen}^{d(K_i,K_j)}$. In practice, choosing the example value of 0.9 for adj_pen means that the base matching quantity awarded to documents containing K_iK_j is lowered by 10% for every other word occurring between K_i and K_j in the original keyphrase.

A further possibility is to define a maximal distance between two words by setting, for example, $\alpha_k = 0$, for k greater than a given maximal distance threshold. A maximal distance of 5 was suggested for English document collections (Jones and Sinclair 1974).

3.4.4 Inversion penalty

$\text{Inv}(K_iK_j)$ is another score modifier used to penalize key pairs K_iK_j that occur in the opposite order in the original keyphrase:

$$\text{inv}(K_iK_j) = \begin{cases} 1, & \text{if } K_i \text{ occurs before } K_j. \\ \text{inv\_pen} \le 1, & \text{otherwise.} \end{cases}$$

Clearly, the non-adjacency and inversion penalties are strongly language- and domain-dependent. The less relative word positions matter, the lower those penalties should be. For a theoretical document collection where relative word positions have no importance, we should have inv_pen $= 1$ and, for $0 \le l \le (m-2)$, $\alpha_l = 1$.

3.4.5 Duplication bonus

A result of the creation of non-adjacent and inverted key pairs is that, whenever one word occurs more than once in a query, the list of word pairs representing the query may contain duplicates. Rather than incrementing a corresponding number of matching quantities, we decide to remove the duplicates, and keep one occurrence of the key pair together with its highest associated matching quantity. This highest matching quantity is further increased by $\text{dup}(K_iK_j)$, a relative weight increase awarded to those pairs occurring several times in the original keyphrase.

3.4.6 Maximal matching distance

Observe that the question of which parts of a document descriptor can be matched with a pair was left open. If the phrasal descriptors are maximal frequent sequences, it is a sensible option to allow for an unlimited gap between each two words of the

Table 1 Quantity of relevance stemming from various indexing phrases with respect to a keyphrase query *ABCD*

Document	Description	Quantity of relevance
d_1	*AB*	Bw(*AB*)
d_2	*ACD*	Bw(*CD*) + α_1 Bw(*AC*) + α_2 Bw(*AD*)
d_3	*AFB*	Bw(*AB*)
d_4	*ABC*	Bw(*AB*) + Bw(*BC*) + α_1 Bw(*AC*)
d_5	*ACB*	Bw(*AB*) + α_1 Bw(*AC*) + $\alpha_1 \cdot$ inv_pen $\cdot$ Bw(*CB*)

Bw stands for Base_Weight

descriptor, because by definition, if *ABCD* is frequent, then so are *AB*, *AC*, *AD*, *BC*, *BD*, and *CD*. In the general case, however, we allow for the possibility to use a maximal matching distance $\max_d$. We try to match two words of a phrasal descriptor against a key pair only if there are no more than $\max_d$ other words occurring between them.

3.4.7 Example

To illustrate these definitions, let us have a look at the decomposition of the keyphrase *ABCD*. It is decomposed into 12 tuples (pair, integrity modifier):

$$(AB, 1), (AC, \alpha_1), (AD, \alpha_2), (BC, 1), (BD, \alpha_1), (CD, 1),$$
$$(BA, \text{inv\_pen}), (CA, \alpha_1 \cdot \text{inv\_pen}), (DA, \alpha_2 \cdot \text{inv\_pen}),$$
$$(CB, \text{inv\_pen}), (DB, \alpha_1 \cdot \text{inv\_pen}), (DC, \text{inv\_pen}).$$

Let us compare this keyphrase to the documents d_1, d_2, d_3, d_4 and d_5, represented respectively by the phrasal descriptors *AB*, *ACD*, *AFB*, *ABC* and *ACB*. The maximal matching distance max_d is set higher than 1. The corresponding quantities of relevance brought by each matching subpart of the keyphrase *ABCD* are shown in Table 1.

Assuming equal Base_Weight values, we observe that the quantities of relevance form an order matching the desirable properties that we had wished for in Sect. 3.2. The longest matches rank first, and matches of equal size are untied by relative word positions (adjacency and inversion). Moreover, non-adjacent matches (*AC* and *ABC*) are taken into account, unlike in many other phrase representations (Mitra et al. 1997).

4 Maximal frequent sequences

Originating from data mining, maximal frequent sequences (MFSs) are very appropriate descriptors for taking into account the sequential essence of text. The generality of the technique permits an application to documents written in any language without any modifications. This is the main reason why we decided to use these descriptors, as they allow to use a consistent approach for the extraction of phrases from all document collections, and only make adaptations in the way we *exploit* these descriptors in a task-based framework.

Maximal frequent sequences were introduced by Ahonen-Myka (1999). In short, they are defined by a minimal frequency threshold and are iteratively expanded for as long as this process does not bring the frequency below the minimal threshold. This permits to extract sequences of any length, and hence offers a very compact phrasal description. Because there is no extraction algorithm that permits to efficiently extract the MFS set of a sufficiently large document collection, we actually relied on an available implementation of MFS_MineSweep (Doucet and Ahonen-Myka 2006) to extract an approximation of the MFSs of each document collection. These techniques are described in more details in the rest of this section.

4.1 Definitions

Definition 1 A sequence $p = a_1 \ldots a_k$ 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 a sequence p is a subsequence of a sequence 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 Fig. 2.

The *interestingness* of a subsequence is usually defined with respect to a set of *constraints*, which are assumed to represent some natural restrictions in the domain. In practice, the constraints are also used to reduce computational costs. The most common constraint is the *minimum frequency*. The frequency of a (sub)sequence can be, e.g., the number of text fragments that contain it.

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 US **retaliation against foreign unfair trade practices**.
3. Washington charged France, West Germany, the UK, Spain and the EC Commission with **unfair practices** on behalf of Airbus.

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, among others, the following frequent sequences in our sample set of sentences: *"congress retaliation against foreign unfair trade practices"* and *"unfair practices"* (Fig. 2).

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.

Fig. 2 A set of sentences from the Reuters-21578 collection

As we will see shortly, the special characteristics of text data usually prohibits discovering all frequent subsequences. Instead, the patterns of interest can be restricted to be *maximal frequent subsequences*.

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.

In our example, the sequence *"unfair practices"* is not maximal, since it is a subsequence of the sequence *"congress retaliation against foreign unfair trade practices"*, which is also frequent. The 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-g *"congress retaliation against foreign unfair trade practices"* would need to be replaced by 21 bigrams. With MFSs, no restriction is put on the maximal length of the phrases. We can thus obtain a very compact representation of the regularities of text. The rest of this section will focus on the problem of the efficient extraction of the set of MFSs of a document collection.

4.2 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 sequence. 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. An important idea in MineMFS is to allow the computation of frequent $(n + 1)$-sequences without enumerating all the frequent n-sequences. The first step of the algorithm (extensively described in (Ahonen-Myka and Doucet 2005)) is to collect all the frequent 2- and 3-g. The main idea is to pick a 3-g and try to combine it with other items in a greedy manner, i.e., as soon as the 3-g 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 last frequent sequence is a maximal one. This step is known as the *expansion step*. When all the frequent 3-g have been processed in this way, those that cannot be used to form a new maximal frequent sequence of size more than 3 are pruned. The remaining ones are joined to produce candidate 4-g that will be used in a new iteration of the process relying on 4-g seeds. This process is repeated until no new maximal frequent sequence can be discovered.

Limitations. Even though the use of minimal and maximal frequency thresholds permits to reduce the burstiness of word distribution, it also causes the algorithm to miss a number of truly relevant word associations. For sufficiently large collections,

the MineMFS process fails to produce results as the convergence towards the resulting set of MFSs takes too long (see Doucet and Ahonen-Myka (2006) for details). This can only be avoided with excessive minimal and maximal frequencies, in which case the set of MFSs produced is small and contains mostly non-interesting descriptors. One reason may be the pruning step, which relies on the heavy process of going through the set of n-grams, and comparing each one of them to every other n-gram with which they can form an $(n + 1)$-gram. Numerous checks have to be computed in this step, if a new item can be added between every two adjacent words of a possibly long sequence. The number of possible positions of insertion shall be problematic.

4.3 Partitioning the document collection to approximate the MFS set efficiently

When we try to extract the maximal frequent sequences of a large document collection, their number and the total number of word features in the collection pose a clear computational problem and do not actually permit to obtain any result.

To bypass this complexity problem, MFS_MineSweep (Doucet and Ahonen-Myka 2006) was presented to decompose a collection of documents into several disjointed subcollections, small enough so that the set of maximal frequent sequences of each subcollection can be extracted efficiently. Joining all the sets of MFSs, an approximate of the maximal frequent sequence set for the full collection can be obtained. Figure 3 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. We will now discuss the relation between the approximation produced by MFS_MineSweep and the actual MFS set.

4.4 Nature of the resulting set of phrasal descriptors

The main drawback of MFS_MineSweep is the loss of the maximality property. During the second step of the method, the MFS sets of each subcollection are gathered into a single set of content descriptors. As there is no clear way to join a sequence (maximal frequent in a subcollection) to its subsequence (maximal frequent in another), both sequences are added to the final set of descriptors, resulting in the loss of the maximality property. This loss implies that the content description produced by MFS_MineSweep is always less or equally compact to the MFS set of the whole document collection.

In the paper introducing MFS_MineSweep (Doucet and Ahonen-Myka 2006), the authors ran an extensive evaluation, aiming to evaluate and compare the quality and quantity of the set of descriptors extracted using MineMFS and MFS_MineSweep. While the first motivation for developing MFS_MineSweep was that MineMFS is simply unable to produce results for sufficiently large collections, it also turned out that it produced more descriptors, especially when applied to homogeneous partitions of the document collection. Indeed, the ability to separately apply MineMFS on

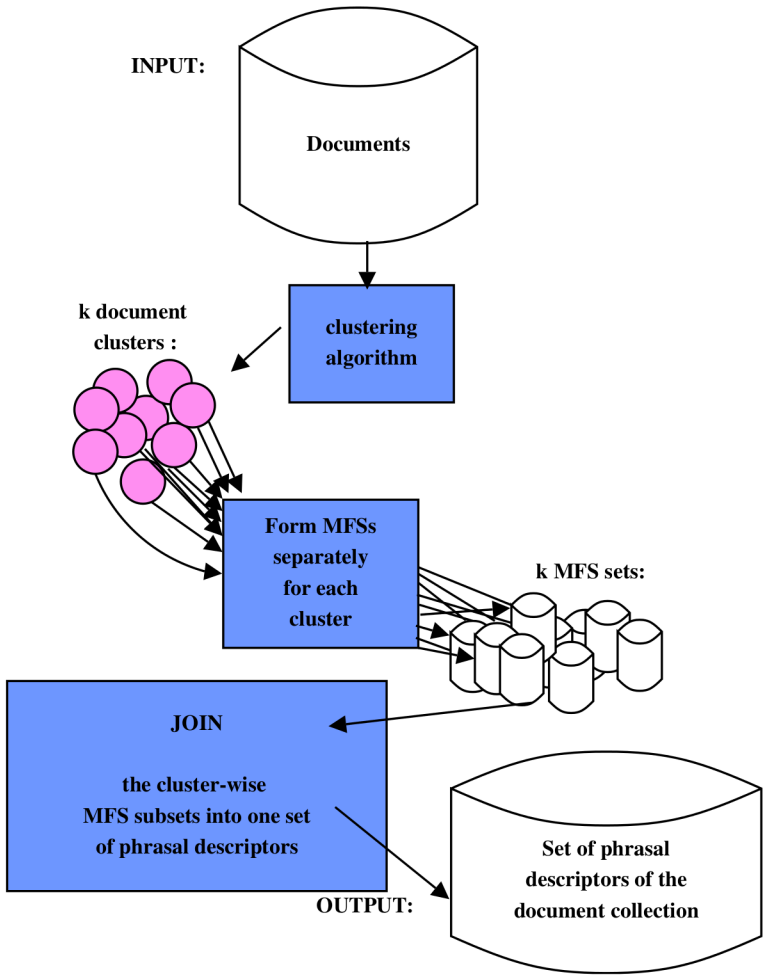

Fig. 3 The different phases of MFS_MineSweep

disjoint subcollections permits locally using looser frequency thresholds to capture more word associations.

To summarize, the effect of this divide-and-conquer process is two-sided: (1) a qualitative and quantitative increase of the document description, (2) a loss in terms of compactness of the description. Since our contribution to the use of phrasal descriptors in document retrieval will be evaluated based on retrieval efficiency, the question of the compactness of the descriptors is irrevelant. In particular, because the computation of the phrasal RSV starts with a decomposition of keyphrases into word pairs, the fact that some sequences of the collection description are subsequences of others is unimportant.

From a linguistic point of view, we shall make a similar observation. Take for instance the set of sentences already presented in Fig. 2. If the frequency threshold

is 2, we extract a maximal frequent sequence "*congress retaliation against foreign unfair trade practices*". One may regret that good MWE candidates such "*unfair practices*", "*trade practices*" and "*retaliation against*" are not singled out in the descriptor set because they are subsequences of an MFS. However, for the same reason, it is not an issue in the context of the application to document retrieval, since the MFS will be decomposed into word pairs, with the weighting scheme described in Sect. 3. Therefore, each of the MWEs above will be taken into account. Naturally, many other sequences will be taken into account, such as sequences of stopwords, or sequences of distant words, but our phrase matching method is precisely meant to lower the weight of such sequences in the resulting phrase RSV.

Another reason why we aim at having most of the MWEs included in the set of descriptors, with little worries about having many other word sequences than the MWEs, is that we believe that the search queries will carry the semantics. Hence, matching the search queries versus the our set of descriptors will essentially concern the most meaningful descriptors and leave the other ones off.

5 Experimental framework

We will now present our practical approach to the evaluation of a set of phrases as content descriptors in the application domain of document retrieval.

5.1 Evaluation measures in document retrieval

The effectiveness of a document retrieval system is measured by comparing the document ranking it generates to the set of *relevance assessments*, a list of the documents of the collection that were judged as relevant and not by domain experts.

Precision measures the proportion of relevant answers among those submitted. Recall measure the relative number of relevant documents found. Since those two measures are interdependent, evaluation methods are generally based on a combination of these two measures. An approach to estimate the quality of a list of retrieved documents is to plot a *recall-precision graph*. The graph is drawn by extrapolation from a number of data points. Typical data points are measures of precision at every 10% of recall, i.e., at recall $0, 0.1, 0.2, \ldots,$ and 1.

For example, the precision at recall 0.4 measures the proportion of all documents the user has to go through in order to find 40% of the relevant documents.

A subsequent popular measure of the quality of a ranked list of documents is the *average precision* over a number of points of recall. For example, for the data points at every 10% of recall, we talk about 11-*point average precision*. Reading the ranked document list from top to bottom, we can also calculate the precision each time a true positive is encountered. By averaging all those precision values together for one query, we obtain a popular measure, the *average precision (AP)*. The *mean average precision (MAP)* is the average of AP across all the queries of a test set. It is central to the evaluation of this work.

5.2 Open questions and protocol of the experiments

In the experiments, we will apply our novel matching technique on MFSs. Our purpose is not to evaluate the quality of MFS as indexing terms for document retrieval, but to see whether our matching technique permits to efficiently improve document retrieval performance.

To answer this question, we will need to produce three runs:

- **WVSM**, a retrieval run that follows the vector space model, built with word term features only.
- **SEQ-Big**, VSM with all the bigrams occurring in sequences: In this run, all the bigrams occurring in an MFS are added to the vector space. For example, with an MFS *ABCD*, the bigrams *AB*, *AC*, *AD*, *BC*, *BD* and *CD* are thrown into the bag of words.
- **SEQ-Adv**, advanced use of sequences: This run applies the technique we presented in Sect. 3.

To answer our question about the performance of our phrase-matching algorithm (SEQ-Adv), we can notably measure the results of SEQ-Adv against the use of phrasal descriptors as a set of frequent pairs used to augment the vector space (SEQ-Big). Naturally, we will also compare those two approaches to the word features baseline (WVSM).

5.3 Tuning our matching technique

Although we expect that the techniques presented in this article can be applied to any language and any type of document, we can make conjectures about document collections for which our phrase-based similarity measure will typically perform better and worse. The following hypotheses are to be verified in the experiments.

- H1: Because our matching technique can account equally for multi-word units whose words occur at various relative positions, we believe that it will bring higher improvement for languages where the relative positions of words are less important (*hypothesis H1*). A corresponding family of languages is known as that of agglutinative languages. The low importance of relative positions is due to the fact that word-modifying morphemes are typically *agglutinated* to the corresponding word, meaning that changing its position seldom changes its role in the sentence. Typical agglutinative languages are, e.g., Turkish, Finnish and Japanese. In the opposite situation, where relative word positions are most important, we do not expect great performance from our matching technique. This situation is that of isolating languages, such as Chinese, Vietnamese, Samoan, or to a lesser extent, English.
- H2: The number of multi-word units that are regularly and consistently used throughout a document collection is generally known to be greater if that collection is specialized. Typically, more multi-word units tend to occur in a technical document than in a newspaper article. Our second hypothesis (H2) is that the improvement brought by our technique will be greater for a more specialized document collection.

Table 2 SEQ-Adv

	Max_d	inv_pen	adj_pen
Adj_Baseline	0	0	Not def.
Balanced	5	0.5	0.8
No_Inv	5	0	0.8
Dist_pen	5	0.5	0.2
Max_d	10	0.5	0.8

The five different runs of the advanced matching technique and their different parameter values for maximal distance (max_d), inversion (inv_pen) and non-adjacency penalty (adj_pen)

As we have seen in Sect. 3, our matching technique functions with a number of parameters to be applied to the key phrases, namely, inversion and non-adjacency penalties, duplication bonus, and maximal matching distance. In this paper, we will also present a few experiments to determine suitable parameter values for each document collection. Naturally, in real-life applications, this would not always be possible. We can, however, give guesses on what would be good parameters, depending on the nature of the document collection.

The same train of thoughts that led us to formulating hypotheses H1 and H2 also leads us to thinking that agglutinative languages and specialized collections will benefit from a higher maximal distance than isolating languages and general collections. To inflict a lower penalty to pairs occurring in inverse order or with many other words between them should similarly benefit agglutinative languages and specialized collections, rather than isolating languages and general collections. To verify these assumptions, we will run our advanced matching technique for each collection with five different sets of parameters. The corresponding five runs are described in Table 2. "Adj_Baseline" rejects inversion, and only considers adjacent words of the key phrase. The run "Balanced" is meant to integrate some of each spice: each parameter is represented with a reasonable general value. Each of the last three runs emphasizes one of the three parameters, as compared to the run "Balanced". For example, "Dist_pen" emphasizes the distance penalty because it lowers the weight of pairs formed from distant words, by setting adj_pen to 0.2 instead of 0.8.

To perform the set of experiments needed, we will now introduce the document collections, upon which our techniques will be applied.

5.4 Presentation of the document collections

Two appropriate collections are the NTCIR collection,[2] and the INEX collection.[3] The INEX collection is a collection of computer science journal articles written in English. The NTCIR collection contains news-feed documents in four distinct languages, namely, English, Japanese, Chinese and Korean. The corresponding collections will permit to confirm or disprove the domain-independence claim we made about our technique, by comparing the results we obtain with scientific and

[2] Details available at http://www.research.nii.ac.jp/~ntcadm/index-en.html.

[3] Details available at http://www.inex.is.informatik.uni-duisburg.de/2005/.

news-feed articles in English, i.e., specialized and non-specialized terminology. Since Chinese is a typical isolating language, and Japanese a typical agglutinative one, we will also be able to measure the performance of our technique on radically different languages.

5.4.1 NTCIR

With the aim to promote information retrieval research on East Asian languages, the Japanese National Institute of Informatics (NII) has made a number of collections of newspaper articles available in English, Japanese, Chinese and Korean under the acronym NTCIR, standing for "NII Test Collection for IR systems". Since these collections are meant for evaluating the performance of document retrieval systems, they are provided with a set of topics and associated manual relevance assessments. A sample of a topic in English is shown in Fig. 4. Our experiments will only use the concept element (<CONC>) that gathers keywords relevant to the topic. As a general rule, keyphrases are comma-separated, which simplifies greatly their extraction from the topics.

In the experiments, we used the NTCIR-3 document collections, statistics about which are summarized in Table 3.

```
<TOPIC>
<NUM>013</NUM>
<TITLE>NBA labor dispute</TITLE>
<DESC>To retrieve the labor dispute between the two
parties of the US National Basketball Association at
the end of 1998 and the agreement that they reached.
</DESC>
<NARR>The content of the related documents should
include the causes of NBA labor dispute, the relations
between the players and the management, main
controversial issues of both sides, compromises after
negotiation and content of the new agreement, etc. The
document will be regarded as irrelevant if it only
touched upon the influences of closing the court on
each game of the season.</NARR>
<CONC>NBA (National Basketball Association), union,
team, league, labor dispute, league and union,
negotiation, to sign an agreement, salary, lockout,
Stern, Bird Regulation.</CONC>
</TOPIC>
```

Fig. 4 An NTCIR topic in English

Table 3 Number of documents and fully assessed topics in the NTCIR-3 collection, per language

Language	Documents	Topics
Chinese	381,681	42
Japanese	220,078	42
Korean	66,146	30
English	22,927	30

5.4.2 INEX

The document collection of the Initiative for the Evaluation of XML retrieval (INEX)[4] is a 494 MB collection of 12,107 English-written computer science articles from IEEE journals.

We carried out experiments based on the set of 30 topics and corresponding assessments of the 1st INEX initiative. We have only used the Keyword element of each topic, of which an example was shown earlier in Fig. 1.

6 Results

An important point of this article is the development of language- and domain-independent techniques. This is put in practice in the following experiments. We have used no list of stopwords, and have applied no stemming. The only exception we made to this rule is in fact applicable to all languages: sentences are delimited by punctuation. We, hence, used every item in the text as a feature, with the exception of punctuation marks (e.g., periods, commas, parentheses, exclamation and question marks). For English, we extracted sequences at the word level (space-delimited), whereas for Asian languages, we performed the extraction at the character level.

MFS extraction. We applied MFS_MineSweep to all document collections using sentence subcollections formed with the k-means algorithm where k was uniformly set to 1 per 50,000 sentences (see Doucet and Ahonen-Myka (2006) for details).

6.1 Results and discussion

6.1.1 Tuning the matching parameters

For each collection, our novel matching technique will be applied to the MFS-based collection representation to produce one retrieval run (SEQ-Adv). This requires finding good parameter values for each collection. We have computed the five runs described in Table 2 for each collection, and we will use the results to determine the score of SEQ-Adv, and to verify the Hypotheses H1 and H2, claiming that our technique should do best for agglutinative languages and specialized collections, as opposed to isolating languages and general collections. The hypotheses further suggested that agglutinative languages and specialized collections should benefit more from raising the maximal distance or lowering the distance and inversion penalties than isolating languages and general collections. This is what we will check with the five runs presented in Table 2, whose corresponding results are given in Table 4.

The confirmation of our assumptions is clear for Chinese, whose isolating nature is shown by the best performance observed when only adjacent non-inverted pairs are considered. As compared to the "Balanced" parameter values, both suppressing

[4] Available at http://www.inex.is.informatik.uni-duisburg.de/2005/.

Table 4 Summary of mean average precision for the five different variations of SEQ-Adv

	Adj_Baseline	Balanced	No_Inv	Dist_pen	Max_d
NTCIR-CH	**0.1885**	0.1818	0.1837	0.1846	0.1820
NTCIR-JP	0.2232	0.2154*	**0.2246**	0.2190	0.2189
NTCIR-KR	0.1370	**0.1498***	0.1477*	0.1378	**0.1499***
NTCIR-EN	0.2186	0.2180	**0.2208**	0.2162	0.2180
INEX(EN)	**0.04370**	0.04193	0.04193	0.04193	0.04193

A starred result (*) indicates that it differs significantly from Adj_Baseline, following paired *t*-test with 95% confidence interval ($p < 0.05$)

inverted pairs and penalizing distance more heavily are beneficial. The only feature for which we cannot confirm our assumptions is the augmentation of the maximal distance. The results are then very similar to those of the "Balanced" run.

The same idea is confirmed with NTCIR-KR, where the agglutinative nature of the Korean language is shown by the domination of the runs in which few restrictions are applied on relative word positions. Using adjacent non-inverted pairs only (0.1370) and emphasizing the distance penalty (0.1378) perform far worse than the other three attempts. Increasing the maximal distance permitted the best performance (0.1499), but the improvement over the balanced parameter set was not significant. Surprisingly, allowing for the inversion of the word pairs affected the results negatively.

Japanese is a very typical agglutinative language, yet we observed the same phenomenon. The run that does not account for inverted pairs is the best-performing of all. The second best is obtained with adjacent non-inverted pairs. However, we could verify that allowing for a longer distance is beneficial for the Japanese collection, as with other things equal, we obtained better results with a maximal distance of 10 (0.2189) than with a maximal distance of 5 (0.2154).

When varying the parameter values, it turns out to be impossible to study the evolution of the results for the two English collections for the simple reason that there is nearly no evolution. The reason why there is no difference in using a maximal distance of five or ten words is that no two English words are connected if there are more than five other words between them, as was shown by Jones and Sinclair (1974). The other parameter variations produce insignificant differences.

Now that we have determined suitable parameter values for our matching technique for each document collection, we can present a summary of our results in Table 5. The results will be further analyzed in the following sections.

6.1.2 Better results for agglutinative languages and specialized collections (Hypotheses H1 and H2)

Agglutinative and isolating languages (H1) For the four NTCIR collections, if we compare the results obtained with the word term vector space model (column WVSM) to those obtained with our technique (column SEQ-Adv), we can notice that our technique provides better results for Chinese and Japanese, while it is

Table 5 Summary of mean average precision for our experiment set

	WVSM	SEQ-Big	SEQ-Adv
NTCIR-CH	(0.1705)	0.1327*	**0.1885***
NTCIR-JP	(0.2151)	0.1480*	**0.2246***
NTCIR-KR	(**0.1707**)	0.1049*	**0.1499**
NTCIR-EN	(0.2555)	**0.2692**	0.2208*
INEX(EN)	(0.04193)	**0.04935***	0.04370

A starred result (*) indicates that it differs significantly from the WVSM Baseline, following paired *t*-test with 95% confidence interval

beaten for English and Korean. Hypothesis H1 was that our technique would perform better for agglutinative languages than for isolating languages. Chinese and Japanese respectively are often cited as very typical of the isolating and agglutinative families of languages. Additionally, English is considered isolating and Korean agglutinative.

Hence, our results do not confirm H1, as we obtained an increase in MAP for both Chinese (+10.6%) and Japanese (+4.4%), while the outcome was a decrease for both English (−13.6%) and Korean (−12.2%).

Specialized and general collections (H2) By similarly opposing the differences between the MAP results of the word terms vector space model (WVSM) and of our technique (SEQ-Adv) for the specialized INEX collection and the NTCIR English news-feed collection, we can observe that only the INEX collection obtains better results with SEQ-Adv (+4.2%). The specificity of the collection truly seems to make a difference, as opposed to the MAP decrease observed with the English NTCIR collection (−13.6%).

H2 is therefore confirmed, as we obtain **better performance for the specialized collection**.

6.2 Impact of our matching technique

Looking at Table 5, we can extend the comments we made as we verified the hypotheses H1 and H2. As compared to the word term vector space model (WVSM), the impact of our matching technique was beneficial for NTCIR-CH (+10.6%), NTCIR-JP (+4.4%) and the collection of English-written computer science articles of INEX (+4.2%). On the other hand, the retrieval of NTCIR-KR (−12.2%) and NTCIR-EN (−13.6%) was more successful with a word-based vector model.

As mentioned in the protocol of the experiments, to truly evaluate the impact of our technique and not the impact of MFSs as descriptors for document retrieval, we should actually compare the results of SEQ-Adv to those of SEQ-Big. SEQ-Big is the approach where the adjacent bigrams occurring in the set of phrasal descriptors are added as extra dimensions of the vector space model. The comparison of our technique to SEQ-Big shows a decrease for both English collections, −11.4% for the INEX collection and −18.0% for NTCIR-EN. A very clear improvement is,

however, observed for all three Asian languages. For Japanese, the MAP improvement is as high as +51.2%. Comparably high benefits are observed for Chinese (+42.0%) and Korean (+42.3%).

The main difference between the way we processed the English and Asian document collections is that we formed words in the English collection, while we worked at the character level for the three Asian collections. This difference of granularity may be a good explanation for the clear improvement brought by our technique in one case, and for the harm it did in the other. This would indicate that the benefit of MFS-based descriptors is linked to the granularity of the items at hand, with same-sized sequences of small items being more useful than those of large items. In other words, a sequence of five characters would be more beneficial than a sequence of five words, because a sequence of five words is too specific. Consequently, our technique permits a higher improvement versus a 2-g baseline, when the grams represent smaller items, e.g., characters rather than words.

7 Conclusion

We presented a novel technique for measuring the similarity of phrasal document descriptors and combining it to word-based vector space similarity measures. We applied our technique to the problem of document retrieval, where we compared the MFS-based phrasal representations of documents to sets of keyphrases describing user needs.

Due to a number of adjustable parameters, our method allows accounting for occurrences of the words of a phrase over a longer span, or in a different order. These usages may be gradually penalized, as compared to an exact phrase occurrence, i.e., adjacent words occurring in the same order. This approach permits taking a wide variation of word usages into account.

It notably deals with the problem of overlapping phrases, as described by Vechtomova (2005). She states the problem of overlapping phrases as the fact that, given a query *ABC*, a document containing the exact match *ABC* and a document containing *AB* and *BC* separately both obtain the same score at the state of the art. A subsequent issue is that the weight of the word *B* becomes artificially heavier than that of *A* and *C*, because *B* is present in both pairs *AB* and *BC*. Our technique permits eradicating this problem, since it can also take the pair *AC* into account. Hence, the distance one between *A* and *C* in the first document (with *ABC*) ensures that it gets a better score than the second document (with *AB* and *BC*). Another consequence is that the weights of *A* and *C* are increased along with that of *B*, avoiding to unbalance the individual term weights within the phrase. A weakness, however, remains with this approach: the word terms that belong to a long phrase appear in numerous subpairs, and hence their artificial weight increase is more important than that of a word occurring in a shorter phrase. Notably, the weight of individual word terms that do not occur in a keyphrase is made lower in comparison to that of word terms occurring in a keyphrase. A solution would be to normalize the weight of terms upon the number and size of the phrases they occur in. This problem is not straightforward, as was recently suggested by work of Robertson et al. (2003)

who proposed to subtract the individual weight of words that occurred redundantly in keyphrases and obtained very disappointing results.

As compared to throwing all descriptors in a bag of words, our similarity measure greatly improves the results for the NTCIR collections in Chinese, Japanese and Korean, with encouraging amelioration ranging between +42% and +51%. This suggests that exploiting languages at a character level may well be the appropriate case for applying our technique with worthwhile improvement.

References

Ahonen-Myka, H. (1999). Finding all frequent maximal sequences in text. In D. Mladenic & M. Grobelnik (Eds.), *Proceedings of the 16th international conference on machine learning ICML-99 workshop on machine learning in text data analysis, Ljubljana, Slovenia*, pp. 11–17.

Ahonen-Myka, H., & Doucet, A. (2005). Data mining meets collocations discovery. In *Inquiries into words, constraints and contexts*, pp. 194–203.

Doucet, A., & Ahonen-Myka, H. (2004). Non-contiguous word sequences for information retrieval. In *Proceedings of ACL-2004, workshop on multiword expressions: Integrating processing*. Barcelona, Spain, pp. 88–95.

Doucet, A., & Ahonen-Myka, H. (2006). Fast extraction of discontiguous sequences in text: A new approach based on maximal frequent sequences. In *Proceedings of IS-LTC 2006, information society—language technologies conference*. Ljubljana, Slovenia, pp. 186–191.

Fagan, J. L. (1989). The effectiveness of a nonsyntactic approach to automatic phrase indexing for document retrieval. *Journal of the American Society for Information Science, 40*, 115–132.

Jones, S., & Sinclair, J. M. H. (1974). English lexical collocations: A study in computational linguistics. *Cahiers de Lexicologie, 24*, 15–61.

Lee, J. H. (1995). Combining multiple evidence from different properties of weighting schemes. In *Proceedings of the 18th annual international ACM SIGIR conference on research and development in information retrieval*, pp. 180–188.

Lewis, D. D. (1992) Representation and learning in information retrieval. Ph.D. thesis, University of Massachusetts at Amherst.

Mitra, M., Buckley, C., Singhal, A., & Cardie, C. (1997). An analysis of statistical and syntactic phrases. In *Proceedings of RIAO97, computer-assisted information searching on the internet*, pp. 200–214.

Robertson, S. E., Zaragoza, H., & Taylor, M. (2003). Microsoft Cambridge at TREC-12: HARD track. In: *TREC*. pp. 418–425.

Salton, G., & Buckley, C. (1988). Term-weighting approaches in automatic text retrieval. *Information Processing and Management: An International Journal, 24*(5), 513–523.

Salton, G., Yang, C., & Yu, C. T. (1975). A theory of term importance in automatic text analysis. *Journal of the American Society for Information Science, 26*(1), 33–44.

Strzalkowski, T., & Carballo, J. P. (1996). Natural language information retrieval: TREC-4 report. In *Text REtrieval Conference*, pp. 245–258.

Turpin, A., & Moffat, A. (1999). Statistical phrases for vector-space information retrieval. In *Proceedings of the 22nd ACM SIGIR conference on research and development in information retrieval*, pp. 309–310.

Vechtomova, O. (2005). The role of multi-word units in interactive information retrieval. In *Proceedings of the 27th ECIR, Spain*, pp. 403–420.

Williams, H. E., Zobel, J., & Bahle, D. (2004). Fast phrase querying with combined indexes. *ACM Transactions on Information Systems, 22*(4), 573–594.