Knowledge Discovery in Documents by Extracting Frequent Word Sequences

HELENA AHONEN

ABSTRACT

AS ONE APPROACH TO ADDRESS THE NEW INFORMATION needs caused by the increasing amount of available digital data, the notion of knowledge discovery has been developed. Knowledge discovery methods typically attempt to reveal general patterns and regularities in data instead of specific facts, the kind of information that is hardly possible for any human being to find. In this article, a method for extracting *maximal frequent sequences* in a set of documents is presented. A maximal frequent sequence is a sequence of words that is frequent in the document collection and, moreover, that is not contained in any other longer frequent sequence. A sequence is considered to be frequent if it appears in at least n documents when n is the frequency threshold given. Frequent maximal sequences can be used, for instance, as content descriptors for documents: a document is represented as a set of sequences, which can then be used to discover other regularities in the document collection. As the sequences are frequent, their combination of words is not accidental. Moreover, a sequence has exactly the same form in many documents, providing a possibility to do similarity mappings for information retrieval, hypertext linking, clustering, and discovery of frequent co-occurrences. A set of sequences, particularly the longer ones, as such may also give a concise summary of the topic of the document.

INTRODUCTION

The research field of knowledge discovery in databases (or data mining) has in the last years produced methods for finding patterns and

Helena Ahonen, Wilhelm-Schickard-Institut für Informatik, University of Tübingen, Sand 13, D-72076 Tübingen, Germany
LIBRARY TRENDS, Vol. 48, No. 1, Summer 1999, pp. 160-181

regularities in structured data, mainly in databases. The studies have included efforts to utilize the existing data about, for example, clients, products, and competition. For instance, patterns in client behavior have been extracted. Unstructured data, particularly free running text, place new demands on knowledge discovery methodology. The representations of knowledge discovered are typically sets of frequently co-occurring items, or clusters of items, that seem to behave similarly in some sense. When the data are structured, they are usually easy to define, that is, what are the parts of data—the occurrence or behavior of the data—that are interesting? Regarding unstructured data, however, this is not at all obvious.

When documents are concerned, the words of these documents may appear to be natural item candidates. The more established fields of information retrieval and natural language processing have traditionally concentrated on words and phrases. The phrases may be linguistic phrases, usually noun phrases, or statistical phrases, which are most often frequent noun-noun or adjective-noun pairs. In data mining research, the solution has been to use keywords from a controlled vocabulary (Feldman & Dagan, 1995; Feldman, Dagan, & Klösgen, 1996), names (of people, companies, etc.), or rigid technical terms that usually are noun phrases. However, verb phrases may also carry important hints on acts and processes similar to the following sequences of words:

bank england provided money market assistance
board declared stock split payable april
boost domestic demand

In this article, a method for extracting frequent word sequences from documents is presented. These sequences can be used as items for further knowledge discovery, but they already represent nontrivial and useful knowledge about documents and the entire document collection. Particularly, the method is able to find the maximal frequent word sequences, which are sequences of words that are frequent in the document collection and, moreover, that are not contained in any other longer frequent sequence. A sequence is considered to be frequent if it appears in at least n documents, when n is the frequency threshold given. The specific demands, due to the characteristics of textual data, include the facts that frequent sequences can be very long and that the frequency threshold has to be set rather low to find any interesting sequences.

Frequent maximal sequences can be used, for instance, as content descriptors for documents: a document is represented as a set of sequences, which can then be used to discover other regularities in the document collection. As the sequences are frequent, their combination of words is not accidental, and a sequence has exactly the same form in many documents, giving a possibility to do similarity mappings for information retrieval, hypertext linking, clustering, and discovery of frequent

co-occurrences. A set of sequences, particularly the longer ones, may also give a concise summary of the topic of the document.

In the next section of this discussion, the entire word sequence discovery process is described, including preprocessing the documents, discovery of maximal sequences, assessing the quality of the discovered word sequences and, finally, the usage possibilities of the sequences. The last section presents experiments conducted using a newswire collection.

Extracting Frequent Word Sequences

The core of knowledge discovery are the algorithms that extract regular patterns in the data. In a practical application domain, however, it is essential to see knowledge discovery not just consisting of fast algorithms but as a process that starts from the data as it is available and ends up in the use of the discovered knowledge for some purpose. In the context of this article, the knowledge discovery process contains the following phases:

Preprocessing of the documents
Discovery of word sequences
Ordering the word sequences
Use of the discovered knowledge

The starting point of the process is ordinary running text. The discovery method to be presented in this section has been developed with such a document collection in mind—i.e., the documents are rather brief. Hence, if longer documents are to be used, it might be necessary to use some kind of fragmentation—e.g., consider each paragraph as a document or use some other advanced method for dividing text into fragments (Hearst, 1995; Heinonen, 1998).

Basic Definitions

In order to formulate the approach presented in this article, some terms have to be defined. Assume that there is a document collection that contains a set of documents. Each document can be seen as a sequence of words and word-like characters. Each word has a unique index that identifies the location of the word both in the document and in the document collection. For instance, in the following, portions of three documents can be seen:

(The,70) (Congress,71) (subcommittee,72) (backed,73) (away,74) (from,75) (mandating,76) (specific,77) (retaliation,78) (against,79) (foreign,80) (countries,81) (for,82) (unfair,83) (foreign, 84) (trade,85) (practices,86)

(He,105) (urged,106) (Congress,107) (to,108) (reject,109) (provisions,110) (that,111) (would,112) (mandate,113) (U.S.,114) (retaliation,115) (against,116) (foreign,117) (unfair,118) (trade,119) (practices,120)

(Washington,407) (charged,408) (France,409) (West,410) (Germany,411) (the,412) (U.K.,413) (Spain, 414) (and,415) (the,416) (EC,417) (Commission,418) (with,419) (unfair,420) (practices,421) (on,422) (behalf,423) (of,424) (Airbus,425)

Actually, the documents are seldom processed as such but rather preprocessed before knowledge is extracted. The possibilities and benefits of preprocessing are discussed in detail later in this article.

Usually knowledge discovery methods express the extracted knowledge as some kind of regular pattern appearing in the data. In the current approach, these patterns are represented as *word sequences.* Like the text itself, a word sequence consists of a sequence of words. It is said that a word sequence *occurs* in a document if all the words contained in the word sequence can be found in the document in the same order as within the word sequence. For instance, in the previous sample documents, the word sequence *(retaliation, against, foreign, unfair, trade, practices)* occurs in the first two documents in the locations (78, 79, 80, 83, 85, 86) and (115, 116, 117, 118, 119, 120). The word sequence *(unfair, practices)* occurs in all the documents, namely in locations (83, 86), (118, 120), and (420, 421).

Naturally, a very large number of word sequences can be found in any document, particularly if sequences of all lengths are considered. The set of all sequences, furthermore, does not contain any knowledge that would not be already contained in the text of the documents. On the contrary, any knowledge would be even more difficult to find. Hence, it is important to consider only word sequences that are frequent in the document collection. A word sequence is said to be *frequent* if it occurs in enough documents to equal at least the given *frequency threshold.* For instance, assuming that the frequency threshold is 10, a word sequence is frequent if it occurs in ten or more documents. Note that only one occurrence of a sequence in a document is counted: several occurrences within one document do not make the sequence more frequent. Different definitions in this respect are, of course, possible.

To restrict the number of word sequences further, a *maximal gap* between words in a sequence is given. That is, the original locations of any two consecutive words of the sequence can have only n words between them at the most if n is the maximal gap. Without this restriction, a large number of short frequent sequences, the words of which are located very far away in the text, would be found—e.g., in the previous example, the sequences *(Congress, foreign), (Congress, practices),* and *(against, practices).*

If a word sequence occurs in a document, it is also a *subsequence* of the sequence of words that constitutes the document. In a similar way, a sequence s is a subsequence of any sequence s' if all the words of s occur in s' in the same order as in s.

If some word sequence is frequent, all of its subsequences are frequent. Hence, if there exists a frequent word sequence of length 10,

1,012 frequent word sequences of length 2-9 are returned. Similarly, if *(dow, jones, industrial, average)* is a frequent word sequence in the collection, all the following sequences are found:

dow jones industrial average
dow jones
dow industrial
dow average
jones industrial
jones average
industrial average
dow jones industrial
dow jones average
jones industrial average

Often, however, the longest possible sequences are the most interesting, and their subsequences do not give more information. Hence, a sequence is returned only if it is a *maximal frequent sequence.* A word sequence *s* is a maximal frequent sequence in the document collection if there does not exist any sequence *s'* in the collection such that *s* is a subsequence of *s'* and *s'* is frequent in the collection. That is, a frequent word sequence is maximal if it is not contained in any other frequent word sequence.

Clearly, when a subsequence does not have any independent meaning, it is rather useless. However, sometimes a subsequence is much more frequent than the respective maximal frequent sequence, which indicates that the subsequence also appears in other contexts and, therefore, has a meaning beyond the maximal sequence context. For instance, the following two maximal frequent sequences contain clearly independent parts, which can be found if they also appear elsewhere—e.g., the name *"Oskar Lafontaine"* without the title *"finance minister"*:

finance minister Theo Waigel
finance minister Oskar Lafontaine
finance minister
Theo Waigel
Oskar Lafontaine

Preprocessing

The preprocessing phase is to transform documents into a representation that can be used by the discovery algorithm. Text includes various constituents—e.g., words, punctuation, special characters, and numbers. Basically, the discovery method accepts any sequence of these constituents and returns the frequent sequences. However, there are two reasons why some preprocessing is useful. First, some frequent words and characters may combine with other constituents in a way that can be regarded as

accidental—i.e., although the combination occurs frequently, it is more due to the frequency of the single words than due to the special meaning this combination carries. Second, the discovery method executes remarkably faster when some constituents have been removed, particularly words and characters that often occur several times within one document. Examples of these are prepositions like *"of"* or articles like *"the."* In the experiments presented in the last part of this discussion, a stoplist of 400 words was used. Furthermore, all numbers that occurred separately were removed. However, words like *"G7-countries"* remained. If the documents are in some structured form, like SGML (Standard Generalized Markup Language) or XML (Extensible Markup Language), either the structure tags have to be removed or the structure must be taken into account somehow. In the current approach, the structure is ignored.

In addition to pruning, the words can also be modified. For instance, stemming can be used to reduce inflected words to their stems. The benefit of stemming is that sequences, which have almost the same meaning but a slightly different form, are combined. If the words are not stemmed, some word sequences may be ignored, as they are not frequent enough, although the slight variations together might exceed the frequency threshold. For instance, if the following sequences both have a frequency of 7 and the frequency threshold is 10, neither of the sequences is discovered.

agricultural production
agricultural products

However, these sequences would guarantee a frequency of 14 for a stemmed sequence—e.g., *"agricultur product."*

On the other hand, stemming may join sequences that have rather different meanings. More sophisticated preprocessing may be provided by using some natural language processing like morphological analysis. Morphological analysis can reduce the words to their base forms, though at the same time retaining the knowledge on the inflected form of the word. Hence, it might be possible to combine the word forms only if the single forms are not frequent enough. This approach is not implemented yet. After all, when morphologically rich languages, like Finnish, are considered, morphological analysis clearly surpasses the stemming process. Moreover, grammatical features can be used to prune words. For instance, the discovery process may be easily restricted to nouns only. As an example of some preprocessing steps, consider the sentence

Documents are an interesting application field for data mining techniques.

The sentence can be first preprocessed using morphological analysis into the following tagged form:

(*document*_N_PL, 1) (*be*_V_PRES_PL, 2) (*an*_DET}, 3) (*interesting*_A_POS, 4)

(*application* _N_SG, 5) (*field* _N_SG, 6) (*for*_PP, 7) (*data*_N_SG, 8)
(*mining*_N_SG , 9)
(*technique*_N_PL, 10)

In the following step, uninteresting parts are pruned, resulting in

(*document* _N_PL, 1) (*interesting*_A_POS, 4) (*application* _N_SG, 5)
(*field*_N_SG, 6)
(*data*_N_SG, 8) (*mining*_N_SG, 9) (*technique*_N_PL, 10)

The text may also be pruned to contain, for example, only nouns,

(*document*_N_PL, 1) (*application*_N_SG, 5) (*field*_N_SG, 6) (*data*_N_SG, 8)
(*mining*_N_SG, 9) (*technique*_N_PL, 10)

or, depending on the goal of the research, the word itself may be discarded and only the morphological information retained:

(N_PL, 1) (A_POS, 4) (N_SG, 5) (N_SG, 6)
(N_SG, 8) (N_SG, 9) (N_PL, 10)

At the end of the preprocessing phase, the text is transformed into a long sequence with a running index. The word "###*document*###" is used to separate the documents in the sequence. The running index guarantees that each occurrence of a word has a unique location number.

Finding Maximal Frequent Word Sequences

As defined in an earlier section, maximal frequent sequences are sequences of words that appear often in the document collection and are not included in another longer frequent sequence. It is not possible to decide locally whether a sequence is frequent or not—i.e., all the instances of the sequence in the collection must be counted. Given a document with a length of twenty words, there are over 1 million possible sequences to be considered. Even if the distance of consecutive words is restricted to two, the number is still as large as 6,500. This number of candidate sequences is computationally prohibitive. Hence, a straightforward way, which includes generating the sequences for each document and then computing the sums of occurrences, is not possible in practice. Additionally, only a small fraction of sequences finally turn out to be frequent in the collection.

In the data mining community, several discovery methods have been presented. The related research includes discovery of frequent sets (Agrawal, Mannila, Srikant, Toivonen, & Verkamo, 1996) and discovery of sequential patterns (Agrawal & Srikant, 1995; Mannila, Toivonen, & Verkamo, 1995). In our context of textual data, a frequent set would be a set of words that co-occur frequently in documents—i.e., the order of the

words is not significant and one word may occur only once in a set. The approaches to discover sequential patterns are usually modifications of the methods for finding frequent sets. Most of these approaches use bottom-up processing: first, the frequent sets, or sequences, of size 1 are found, then longer frequent sequences are iteratively formed from the shorter ones, using the pruning principle that if a sequence is not frequent, none of the sequences in which it is included can be frequent. Finally, all the maximal sets or sequences are also found. The problem is that, when the maximal sequences are long, the number of shorter sequences is rather prohibitive. Hence, it is not possible to generate, or even consider, all the subsequences of the maximal sequences. Some approaches exist that attempt to compute the maximal sets directly; for example, in Gunopulos, Khardon, Mannila, and Toivonen (1997), a randomized algorithm is used. In Bayardo (1998), the particular goal is to handle large maximal sets. To summarize, the methods to discover sequential patterns also find maximal frequent sequences but, due to performance reasons, they do not succeed with text documents, whereas no other methods for directly finding frequent maximal sequences exist.

In the method presented in this article, the bottom-up approach is combined with the direct discovery of maximal sequences. The method is divided into two phases: initialization and discovery. First, in the initialization phase, all the frequent pairs are collected from the documents. This is done by collecting all the pairs in which the words of the pair have at most two words in between. The quantity of documents for each pair is then computed, and the pairs that occur frequently enough form the initial set of sequences for the discovery phase.

Assume the document collection contains the following six documents in which the words are replaced by letters:

1: (A,11) (B,12) (C,13) (D,14) (E,15)
2: (P,21) (B,22) (C,23) (D,24) (K,25)
3: (A,31) (B,32) (C,33) (H,34) (D,35) (K,36)
4: (P,41) (B,42) (C,43) (D,44) (E,45) (N,46)
5: (P,51) (B,52) (C,53) (K,54) (E,55) (L,56) (M,57)
6: (R,61) (H,62) (K,63) (L,64) (M,65)

In order to be counted as an instance of a pair in the initialization phase, two words in a document can have between them at most two other words. Assume that the frequency threshold is set to 2:

AB: 2 BE: 3 CK: 3 EL: 1 HM: 1 PC: 3
AC: 2 BH: 1 CL: 1 EM: 1 KE: 1 PD: 2
AD: 1 BK: 2 CN: 1 EN: 1 KL: 2 PK: 1
AH: 1 CD: 4 DE: 2 HD: 1 KM: 2 RH: 1
BC: 5 CE: 3 DK: 2 HK: 2 LM: 2 RK: 1
BD: 4 CH: 1 DN: 1 HL: 1 PB: 3 RL: 1

As longer sequences are constructed from the shorter ones, a container of the current sequences has to be maintained. This container is called the *set of grams*, and the sequences contained in this set are called *k*-grams in which *k* is the length of the gram. After the initialization phase, the set of grams contains all the pairs—i.e., 2-grams that have a frequency of at least 2: {AB, AC, BC, BD, BE, BK, CD, CE, CK, DE, DK, HK, KL, KM, LM, PB, PC, PD}.

The discovery phase, in turn, is divided into three steps: the expansion, the join, and the prune steps. In the expansion step, a new word is added to the pair and, if the resulting 3-gram is still frequent, the same is repeated until the resulting sequence is no more frequent. The last frequent sequence is now a maximal frequent sequence.

Continuing with the previous example, the processing starts from the pair AB. The set of grams is used to find candidate words to be added since only sequences that are constructed from the frequent pairs can be frequent. A candidate sequence ABC can be constructed using the grams, and its frequency, when checked against the documents, is found to be 2. A new attempt is made to add a word: the sequence ABCD is again found to be frequent. However, all other attempts to make the sequence longer fail since the possible candidate sequences ABCDE and ABCDK are not frequent. Hence, the sequence ABCD is a maximal frequent sequence.

The other pairs are processed in a similar way. However, to ensure that the same maximal sequences are not produced several times, a pair is not expanded if it is already a subsequence of some maximal sequence. When all the pairs have been processed, every pair belongs to some maximal sequence. If some pair cannot be expanded, it is itself a maximal sequence. A pair that is a maximal sequence can be removed from the set of grams as it cannot be a part of any other maximal sequence.

In the previous example, the pairs AC, BC, and BD are subsequences of ABCD and, hence, are not expanded. The pair BE can be expanded to the sequence BCDE, BK to BCDK, KL to KLM, and PD to PBCD. The pair HK cannot be expanded, and it is removed from the set of grams. All other pairs are subsequences of some maximal sequence. Hence, after the first expansion step, the set of maximal frequent sequences is {ABCD, BCDE, BCDK, KLM, PBCD, HK}.

In the expansion step, all the possibilities to expand may have to be checked—i.e., the new word can be added to the end, to the front, or to the middle of the sequence. If one expansion does not produce a frequent sequence, other alternatives have to be considered. However, when a suitable word is found, other alternatives can be ignored.

As seen earlier, the test for frequency of some sequence has to be done regularly. Due to efficiency reasons, it is not reasonable to check the frequency from the documents. Therefore, the occurrences of the frequent pairs found from the documents are stored in a separate data

structure. For instance, for the pairs AB, AC, and BC, the following occurrences are stored:

AB: [11-12][31-32]
AC: [11-13][31-33]
BC: [12-13][22-23][32-33][42-43][52-53]

If the frequency of the sequence ABC has to be found, the data structure is first searched to find the occurrences of AB, in this case [11-12] and [31-32]. Then those occurrences of BC that are continuations of the occurrences of AB are found, namely [12-13] and [32-33]. Finally the occurrences for the entire sequence ABC can be combined: [11-13] and [31-33]. The number of these occurrences is returned as the frequency of the sequence.

In the join step, the pairs—i.e., 2-grams—are joined to form 3-grams. For instance, if there are pairs AB, BC, and BD, new sequences ABC and ABD are formed. Within the example, the following 3-grams are included in the new set of grams, but the 3-gram KLM can be removed since it is already known to be maximal:

ABC	ACK	CDE	PCD	BKM
ABD	BCD	CDK	PCE	CKL
ABE	BCE	PBC	PCK	CKM
ABK	BCK	PBD	PDE	DKL
ACD	BDE	PBE	PDK	DKM
ACE	BDK	PBK	BKL	(KLM)

Now the expansion step is repeated: all 3-grams are processed and expanded. The difference, however, is that the 3-grams are not necessarily frequent. Actually, a 3-gram formed by joining two 2-grams does not necessarily occur at all in the documents. For instance, if there are pairs AB and BC but BC always occurs before AB in the documents, the sequence ABC never occurs. The 3-grams that are not frequent are pruned away from the set of grams.

The 3-grams ABC and ABD are subsequences of ABCD, which is maximal. The 3-gram ABE occurs only once in the documents and can be removed from the set of grams. In fact, there are only two 3-grams, namely PBE and PBK, that are frequent but are not already subsequences of maximal sequences. These grams are expanded to maximal sequences PBCE and PBCK, respectively.

After the expansion step, the set of grams looks like the following:

ABC	BCE	CDE	PBE	PCK
ABD	BCK	CDK	PBK	
ACD	BDE	PBC	PCD	
BCD	BDK	PBD	PCE	

The expansion and join steps are iterated in a similar way as explained above: *k*-grams are expanded to maximal sequences and then joined to

form *k+1*-grams. As all the grams that are itself maximal sequences are removed, as well as the grams that are not frequent, the processing stops when the set of grams is empty. In practice, when the sequences can be long (e.g., twenty-two words), it is necessary to apply pruning, which makes it possible to remove grams before the gram length is the same as the length of the maximal sequence. The prune step is explained shortly below.

The following 4-grams belong to the set of grams. The grams ABCD, BCDE, BCDK, PBCD, PBCE, and PBCK are maximal, the other grams become infrequent. Hence, the processing stops after this expansion step:

(ABCD) ABDK (BCDK) PBDE
ABCE ACDE (PBCD) PBDK
ABCK ACDK (PBCE) PCDE
ABDE (BCDE) (PBCK) PCDK

Finally, the set of maximal frequent sequences includes the sequences {*ABCD, BCDE, BCDK, KLM, PBCD, HK, PBCE, PBCK*}. All of these maximal sequences have a frequency of 2.

The basic version of the algorithm proceeds to as many levels as is the length of the longest maximal sequence. When there exist long maximal sequences, this can be prohibitive, since on every level the join operation increases exponentially the number of the grams contained in the maximal sequences. However, often it is unnecessary to wait until the length of the grams is the same as the length of the maximal sequence in order to remove the grams of the maximal sequence from the set of grams. As there must always be grams from at least two maximal sequences to form a new maximal sequence, we can check, for each maximal sequence *m*, whether there are any other maximal sequences that can still contribute in forming new sequences and, hence, can preclude the removal of the grams of *m*. If no such other maximal sequences are found, the maximal sequence is declared to be *final*.

For instance, the following principles can be used on the level *k* to decide whether a maximal sequence is final:

- A maximal sequence is final if it does not share any *k-1*-prefixes or suffixes of grams with another maximal sequence in at least *n* documents in which *n* is the frequency threshold.
- Let *M* be the set of maximal sequences that share a *k-1*-prefix or -suffix of a gram with the maximal sequence *m*. For each *m'* in *M*, we align *m* and *m'* to know which subsequences of *m* are also subsequences of *m'*. If there is a common subsequence which is longer than the maximal gap, the sequence *m' binds m*. Otherwise, if there are no subsequences, such that the respective positions in *m* have a distance larger than *k*, the sequence *m'* does not *bind m*. If no *m' in M* binds *m*, *m* is final.

The pruning of maximal sequences proceeds as follows. After a join step on level k, as the set of grams contains $k+1$-grams, the maximal sequences are processed. First, if a maximal sequence is of size $k+1$, the respective gram is removed from the set of grams. Second, for all the longer maximal sequences, it has to be decided whether the sequence is final. Finally, after all the maximal sequences have been considered, any gram that is a subsequence of final maximal sequences only is removed. Only the grams of a maximal sequence are removed; the sequence is not removed from the set of maximal sequences in order to assure that the new sequences are not already subsequences of some maximal sequence.

Above, a method for discovering maximal frequent sequences was presented. After all the maximal frequent sequences have been discovered, their subsequences that are more frequent than the corresponding maximal sequence can also be found. The more frequent subsequences are found for each maximal sequence in turn. Continuing the running example, first, a set of grams is formed from the pairs contained in the maximal sequence ABCD—i.e., the set contains the 2-grams {AB, AC, AD, BC, BD, CD}. Second, two grams are joined to form a 3-gram if the frequency of the 3-gram is higher than the frequency of the maximal sequence—i.e., in this case, higher than 2. Only one such 3-gram can be formed—i.e., BCD, which has a frequency of 4. As a subsequence can also be a subsequence of some other subsequence of the maximal sequence, it is only considered interesting if it is more frequent than any of the other subsequences that contain it. Hence, a 2-gram is included in the set of more frequent subsequences if it is more frequent than the maximal sequence, and either it is not included in any 3-gram or it is more frequent than any of the 3-grams in which it is contained. From the 2-grams in the set of grams above, only the gram BC fulfills the requirements. The grams BD and CD are more frequent than the maximal sequence ABCD, but these are contained in BCD and have the same frequency as it has, namely 4. These steps, joining of the k-grams to form k+1-grams and checking whether the k-grams fulfill the requirements, are repeated until the set of grams is empty. This procedure is done for each maximal frequent sequence. Within the previous example, the final set of more frequent subsequences looks like the following (the number in parentheses is the frequency of the sequence):

ABCD(2): BCD(4), BC(5)
BCDE(2): BCD(4), BCE(3), BC(5)
BCDK(2): BCD(4), BCK(3), BC(5)
KLM(2): no more frequent subsequences
PBCD(2): PBC(3), BCD(4), BC(5)
HK(2): no subsequences
PBCE(2): PBC(3), BCE(3), BC(5)
PBCK(2): PBC(3), BCE(3), BC(5)

Examples of both maximal frequent sequences and their more frequent subsequences, as extracted from a newswire text collection, can be found in the Examples and Experiments section.

Ordering and Pruning Word Sequences

Some decisions will now be presented which reduce the number of word sequences found in the documents. First, to be counted as a word sequence, the sequence cannot contain gaps that are longer than the maximal gap given. Second, the word sequence has to be frequent. Third, a word sequence has to be a maximal sequence or a subsequence of some maximal sequence such that the subsequence has to be more frequent than the respective maximal sequence. However, even after these restrictions, the number of word sequences found may be rather large, at least for some application purposes. In this section, some postprocessing opportunities are presented that make it possible to find an ordering for the word sequences. If the word sequences are ordered according to their assumed quality, the less valuable sequences can be pruned if necessary. Also, even a large set of sequences is easier to utilize if the sequences are presented in the order of their assumed quality.

A central problem with the knowledge discovery approaches is to define the notion of *interestingness.* The basic methods usually return large sets of information, and some extra effort is needed to evaluate the results and decide which are interesting for the given application.

With the approach presented in the previous sections, interestingness is in the first place determined by the frequency of the sequence. Actually, with long sequences, it is a very good measure since the frequency guarantees that the combination is not accidental. On the other hand, long sequences hardly appear so often that they lose their interestingness value. However, with single words and short sequences, it may happen that a high frequency indicates that the sequence has no describing power and that its interestingness is low. Hence, some further measures have to be considered to focus on good sequences.

In data mining, typically the relevance of a pattern is very strongly dependent on the application. One of our hypotheses is that there are measures of relevance that are common to all documents independent of the semantic content of the texts. However, different usage needs also affect application of the measures. In the following, a set of measures is introduced. Each of these measures attempts to describe one aspect of interestingness using knowledge gathered from the word sequences. The measures are:

- **Length:** The absolute length *wlen* of a word sequence is the number of words it contains. The relative length *len* is defined as *1 - (1/wlen).*

- **Frequency:** The frequency *f* of a word sequence is the number of documents in which the sequence occurs. We also consider separately the number of occurrences of a word sequence within a document. This frequency is denoted as *tf*.
- **Tightness:** If the first word of a sequence appears in location *x* and the last word in location *y*, the window of the occurrence is defined as $y - x + 1$. If *win* is the average of all the windows of the occurrences of a sequence, the tightness *t* of the sequence is $1 - ((win - wlen)/win)$.
- **Stability**: A word sequence *s* of absolute length *n* contains $P = \sum_{k=2}^{n-1} \binom{n}{k}$ subsequences.
  Assume there are subsequences $s_1, \ldots, s_k$ with the frequencies higher than the frequency of *s*, and for each s_i, $cont(s_i)$ is the number of subsequences s_i itself contains. Additionally, let *S* be the sum of these sets, i.e., $S = cont(s_1) + \ldots + cont(s_k)$. Furthermore, the relative frequency of these subsets is computed as $F = cont(s_1) \times f(s)/f(s_1) + \ldots + cont(s_k) \times f(s)/f(s_k)$ in which $f(s_i)$, $1 \leq i \leq k$, are the frequencies of the subsequences $s_1, \ldots, s_k$, respectively. The stability *st* of the word sequence *s* is now computed as $st = (F/S + (1 - S/P))/2$.
- **Inverse Document Frequency (IDF):** The IDF is computed as $idf = -\log$ (number of documents containing the sequence / number of documents in the collection). This measure is also scaled to receive values between $0 - 1$. The relative IDF is computed by dividing *idf* by the maximal IDF value *max_idf*, which is obtained when a word sequence occurs in *n* documents, in which *n* is the frequency threshold—i.e., $rel\_idf = idf / max\_idf$.

In order to illustrate the measures of tightness and stability, which are not as intuitive as the other measures, some examples are given in the following:

Tightness depicts the number of gaps that can be found, on the average, in the occurrences of the word sequence in the documents. All the occurrences are counted—i.e., also the duplicate occurrences in one document. In the first example of this article, three documents were presented. In these documents, a word sequence *(unfair, practices)* occurred in locations (83, 86), (118, 120), and (420, 421). The average window is now 3, and the length of the sequence is 2. Hence, the tightness is $1 - (3-2)/3 = 1 - 1/3 = 0.67$.

Stability attempts to reveal the ties between the constituents of the word sequence. If the subsequences of the sequence occur in the context of the sequence only, which is depicted by the fact that they have the same frequency as the sequence itself, the stability is 1. If, however, there are subsequences that are more frequent, the stability gets a value between 0 and 1.

In the following, one word sequence and its more frequent subsequences with frequencies are presented:

available export enhancement program initiative announced 11
—program initiative 22
—program announced 37
—enhancement initiative 25
—enhancement announced 20
—available export 25
—export initiative announced 19

As the length of the sequence is 6, the sequence has 56 subsequences—i.e., $P = 56$. Only subsequences longer than 2 are counted. Therefore, from the more frequent subsequences, only the last one, which itself has a length of 3, has subsequences of its own. The sum of more frequent subsequences is 8—i.e., $S = 8$. $F = 11/22 + 11/37 + 11/25 + 11/20 + 11/25 + 3 \times 11/19 = 3.97$. The stability is now computed as $st = (3.97/8 + (1 - 8/56)) / 2 = 0.68$.

The ranking of word sequences within a document is computed by combining the above measures. It can be done in several ways, depending on the emphasis given to each measure. As the frequency is considered to be the central factor, the weight w is computed by multiplying the frequency of a word sequence within a document by a combination of the other measures. In the experiments presented in this article, the weight is computed as $w = tf \times (len + t + st + rel\_idf) / 4$. After each word sequence has a weight, it is possible to select the best word sequences and prune away the sequences with a weight less than a given threshold. The threshold can have a predefined fixed value or a dynamic value depending on, for example, the number of the word sequences found in the collection.

Use of Frequent Word Sequences

In this section, some possible uses of frequent word sequences are discussed. These uses include applying word sequences within some existing applications, but this technology may also make available new ways to face information overload.

For instance, imagine the following situation. The user makes a query that may contain single words or a more complicated search expression. Instead of returning the documents that fulfill the conditions of the query, the search engine returns only the maximal frequent word sequences. The user can see the contexts of the query terms and use this knowledge to decide which maximal sequences correspond to his or her information need. Assume the user has given a query term *"agriculture."* If stemming is used, the following word sequences might be returned:

agricultural exports
agricultural production

agricultural products
agricultural stabilization conservation service
agricultural subsidies
agricultural trade
u.s. agriculture
agriculture department usda
agriculture department wheat
agriculture hearing
agriculture minister
agriculture officials
agriculture subcommittee
agriculture undersecretary daniel amstutz
common agricultural policy
ec agriculture ministers
european community agriculture
food agriculture
house agriculture committee

If the user chooses the sequence *"agricultural subsidies,"* the search engine can return the documents containing this sequence. Instead of choosing one sequence only, also several sequences, connected by Boolean operators, can be chosen.

Before seeing the entire documents, the user could also ask, Which other word sequences appear in the same documents with the sequence *"agricultural subsidies?"* That is, instead of the text of the documents, the set of word sequences for each document would be returned. This representation would give the user an overview to the documents, particularly if not only the plain word sequence is returned but also the entire sentence or paragraph in which the word sequence occurs.

Furthermore, the user could choose some documents to be viewed. If some document looks promising, the entire document can be used as a query to search other documents that are somehow similar. The similarity comparison that is needed to implement this functionality can be based on the frequent word sequences as a representation of a document.

In particular, the word sequences might support information seeking when the user does not have a very clear idea what he or she is looking for, and when the user is not familiar with the document collection. The frequent word sequences offer a mediating level, which can be used to expand even very short queries. As today ordinary users can directly search large full-text document collections without the help of a professional librarian, new challenges are to be addressed. According to some studies (Grefenstette, 1997), users tend to use few words only in their queries even if search engines offer more advanced search expressions. Hence, new simple tools should be offered users. The word sequences are close to the normal text and probably easy to accept.

In addition to the information searching process where the user has even some kind of an idea what he or she is looking for, the word sequences also support knowledge discovery where the user is looking for

new, interesting, and surprising knowledge contained in the documents. For instance, a discovery process might find associations between word sequences—e.g., that names of two companies seem to occur more often in the same documents than what is to be expected. Moreover, the documents can be clustered using the word sequences. Clusters give an overview to the collection, and these can also be a way to restrict the scope of further searching to a subset of the documents. Finally, as the word sequences inherently have exactly the same form in several documents, they can be used to automatically generate hypertext links between documents.

In practice, the set of word sequences for a document should probably be expanded by frequent content-bearing single words of the document. Some words that have an important meaning occur in many contexts, which may preclude them participating in frequent word sequences. Furthermore, in many languages other than English, composite words are favored instead of phrases of several words.

Examples and Experiments

The publicly available Reuters-21578 newswire collection, which contains about 19,000 documents, has been used for experiments (see http://www.research.att.com/~lewis/reuters21578.html). The average length of the documents is 135 words. Originally, the documents contained 2.5 million words. After stopword pruning against a list of 400 stopwords, the amount of words is reduced to 1.3 million, which are instances of 50,000 distinct words.

The list of stopwords contains, for example, single letters, pronouns, prepositions, some common abbreviations from the Reuters collection (e.g., pct, dlr, cts, shr), and some other common words that do not have any content-bearing role in text phrases. On the other hand, many common verbs are not stopwords since they may add some necessary context into phrases (e.g., call, bring, and sell). In addition to stopwords, all numbers are removed from the documents. Stopword pruning is necessary, since the efficiency of the algorithm may be ruined if there are many words that appear several times in one document. Frequency of words as such is not a problem, although they naturally increase the size of data, and sequences containing very frequent words only may not be very interesting.

Experiments have been performed with frequency thresholds 10 and 15. Table 1 summarizes how many pairs and occurrences were found in the initial phase.

Table 1.
Frequent Pairs and Number of Their Occurrences

Frequency	10	15
Frequent pairs	22,273	2,071
Occurrences	556,522	427,775
Pairs/document (average)	30	23

Additionally, there were 406 documents without any frequent pairs. Table 2 shows some performance figures of the execution with a frequency threshold of 10. Before the first pass over the set of grams, 3.40 minutes were needed to construct the initial data structures. As can be seen from the table, most of the time is used on the first level (actually level 2), when most of the maximal sequences, also the longer ones, are extracted. Although the longest maximal sequences are twenty-five words long, the process ends after the fourth round. Hence, the pruning of grams of the maximal sequences is effective.

Table 2.
TIME AND SPACE CONSUMPTION WITH A FREQUENCY 10

Level	2	3	4
Pass time over the set of grams (min)	44.30	2.33	0.06
Join time (min)	7.27	0.19	0.00
Grams in the set	22,273	4,849	152
Grams that not-in-maximal when seen	17,089	3,898	136
New maximal sequences	17,089	796	32
Grams that are maximal	13,984	709	31
Nonfrequent grams	0	3,102	104
Final maximal sequences	17,007	877	33
Maximal sequences remain	82	1	0
Grams after *join*	12,042	1,544	21
Grams after pruning	4,849	152	0

With a frequency 15, the total time of the execution was 30.42 minutes and the algorithm proceeded, as with the frequency 10, through levels 2, 3, and 4. The numbers of frequent maximal sequences of various sizes are shown in Table 3.

Table 3.
NUMBERS OF MAXIMAL FREQUENT SEQUENCES OF DIFFERENT LENGTH

Length	2	3	4	5	6	7	8	9	10	11	12	13
σ=10	43,273	8,417	1,963	764	321	116	38	34	21	18	15	10
σ=15	21,635	4,169	1,002	394	171	62	22	19	10	9	7	6

Length	14	15	16	17	18	19	20	21	22	23	24	25
σ=10	4	2	2		16	2	2	1	4			2
σ=15	2		1	1		8	1	1	1	2		

As an example of word sequences discovered for one document, consider the following sequences:

power station	11
immediately after	26
co operations	11
effective april	63
company's operations	20
unit nuclear	12
unit power	16
early week	42
senior management	28
nuclear regulatory commission	14
—regulatory commission	34
nuclear power plant	26
—power plant	55
—nuclear power	42
—nuclear plant	42
electric co	143

The document describes how the Nuclear Regulatory Commission ordered one nuclear power plant to be shut down after determining that operators were sleeping on duty. Moreover, the action of the electric company to improve the situation is stated. As the discovery of word sequences is based on frequency, the word sequences extracted tell more about the domain than about the details covered in this specific story.

In addition to the discovery of maximal frequent word sequences, more frequent subsequences were also found. Instead of accepting subsequences that are only slightly more frequent, it was required that a subsequence occur at least in seven documents more than for the corresponding maximal sequence. In the experiments, the number of maximal sequences that had subsequences that were more frequent than the maximal sequence itself was 2,264 within the average of two subsequences. As an example of a long word sequence that has several more frequent subsequences, consider the following sequence:

> previous business day treasury latest budget statement
> balances tax loan note accounts fell respective days
> treasury's operating cash balance totaled

The frequency of the sequence is 13 and contains the following subsequences that are more frequent:

operating cash	29
cash balance	29
business day	33
treasury statement	24
statement tax	22
statement loan	25
latest statement	23
previous day	25
previous treasury	21
budget statement	22

budget tax	37
tax loan	23
tax note	27

Finally, for each word sequence, the measures described in an earlier section were computed. In Table 4, the measures for the document already discussed can be seen. The word sequence *nuclear regulatory commission*, and hence also the sequence *regulatory commission*, occurs in the document twice, which contributes to the high weight. It can be seen from the tightness (*t*) values that the sequences *power station*, *nuclear regulatory commission*, and *nuclear power* are very rigid compositions, as they never allow other words within them. The sequence *nuclear power plant* receives rather poor stability (*st*) value, since all of its subsequences are clearly more frequent. Furthermore, the sequence *electric co* has the worst relative IDF (*rel_idf*) value due to its frequency in the document collection.

Table 4.
WEIGHTING MEASURES FOR WORD SEQUENCES OF ONE DOCUMENT (F = FREQUENCY, T = TIGHTNESS, ST = STABILITY, REL_IDF = RELATIVE INVERTED DOCUMENT FREQUENCY, LEN = LENGTH)

Word Sequence	*f*	*t*	*st*	*rel_idf*	*len*	*weight*
power station	11	1.00	-	0.99	0.50	0.83
immediately after	26	0.98	-	0.87	0.50	0.78
co operations	11	0.63	-	0.99	0.50	0.71
effective april	63	0.98	-	0.75	0.50	0.74
company's operations	20	0.65	-	0.91	0.50	0.68
unit nuclear	12	0.67	-	0.98	0.50	0.71
unit power	16	0.55	-	0.94	0.50	0.66
early week	42	0.92	-	0.81	0.50	0.74
senior management	28	0.98	-	0.86	0.50	0.78
nuclear regulatory commission	14	1.00	0.54	0.95	0.67	1.58
- regulatory commission	34	0.99	-	0.84	0.50	1.54
nuclear power plant	26	0.99	0.29	0.87	0.67	0.70
- power plant	55	0.93	-	0.77	0.50	0.74
- nuclear power	42	1.00	-	0.81	0.50	0.77
- nuclear plant	42	0.77	-	0.81	0.50	0.69
electric co	143	0.86	-	0.65	0.50	0.67

CONCLUSION

In this article, a method for extracting maximal frequent word sequences from documents was presented, and its possible uses in several tasks were discussed. A maximal frequent word sequence is a sequence of words that is frequent in the document collection and, moreover, is not contained in any longer frequent sequence. A sequence is considered to be frequent if it appears in at least n documents, when n is the frequency threshold given.

The process for extracting useful word sequences contains several phases. In the preprocessing phase, some common words and numbers are pruned from documents, and the documents are transformed into a sequence of words. In the discovery phase, on the one hand, longer word sequences are constructed from shorter ones; on the other hand, maximal sequences are discovered directly in order to avoid a prohibitive number of intermediate sequences. Subsequences which are more frequent than the corresponding maximal sequence can also be found in a separate postprocessing step. When each document has received a set of frequent word sequences, the sequences are ordered according to a set of measures that assess the quality of the sequences. If necessary, the set of sequences can be pruned based on the ordering.

The representation of a document as a set of frequent word sequences can form a basis for several different information searching tools. First, the word sequences can serve as content descriptors that can be used for similarity computations needed—e.g., in clustering, automatic hypertext link generation, and matching documents with a query. Second, after a user query, the word sequences can be shown as an intermediate representation of the documents before the user makes the final decision to see the entire texts. Third, the word sequences may act as a set of features for further knowledge discovery. For instance, associations between word sequences, and hence between documents, can be discovered.

The discovery method and ordering principles have been implemented in the Perl programming language. In experiments, the Reuters-21578 newswire collection was used. Further research in the area includes applying the method to the information searching tasks described earlier.

REFERENCES

Agrawal, R.; Mannila, H.; Srikant, R.; Toivonen, H.; & Verkamo, A. I. (1996). Fast discovery of association rules. In U. M. Fayyad, G. Piatetsky-Shapiro, P. Smyth, & R. Uthurusamy (Eds.), *Advances in knowledge discovery and data mining* (pp. 307-328). Menlo Park, CA: AAAI Press.

Agrawal, R., & Srikant, R. (1995). Mining sequential patterns. In *Proceedings of the 11th International Conference on Data Engineering* (March 6-10, 1995, Taipei, Taiwan) (pp. 3-14). Los Alamitos, CA: IEEE Computer Society Press.

Bayardo, R. J. (1998). Efficiently mining long patterns from databases. In L. Haas & A. Tiwary (Eds.), *SIGMOD '98* (Proceedings of the 1998 ACM SIGMOD Conference on Management of Data: June 1-4, 1998, Seattle, Washington) (pp. 85-93). New York: Association for Computing Machinery Press.

Feldman, R., & Dagan, I. (1995). Knowledge discovery in textual databases. In U. M. Fayyad & R. Uthurusamy (Eds.), *Proceedings of the First International Conference on Knowledge Discovery and Data Mining* (August 1995, Montreal, Canada) (pp. 112-117). Menlo Park, CA: AAAI Press.

Feldman, R.; Dagan, I.; & Klösgen, W. (1996). Efficient algorithms for mining and manipulating associations in texts. In R. Trappl (Ed.), *Cybernetics and systems research: The Thirteenth European Meeting on Cybernetics and Systems Research* (pp. 949-954). Vienna, Austria: Taylor & Francis.

Grefenstette, G. (1997). Short query linguistic expansion techniques: Palliating one-word queries by providing intermediate structures to text. In M. T. Pazienza (Ed.), *Information extraction: A multidisciplinary approach to an emerging information technology* (No. 1299: Lecture Notes in Artificial Intelligence) (pp. 97-114). New York: Springer.

Gunopulos, D.; Khardon, R.; Mannila, H.; & Toivonen, H. (1997). Data mining, hypergraph transversals, and machine learning. In *Proceedings of the Sixteenth ACM SIGACT-SIGMOD-*SIGART Symposium on Principles of Database Systems, PODS 1997 (Tuscon, Arizona, May 12-14, 1997) (pp. 209-216). New York: Association for Computing Machinery Press.

Hearst, M. A. (1995). Tilebars: Visualization of term distribution information in full text information access. In *CHI '95: Human Factors in Computing Systems: "Mosaic of Creativity"* (May 7-11, 1995, Denver, Colorado) (pp. 59-66). New York: Association for Computing Machinery.

Heinonen, O. (1998). Optimal multi-paragraph text segmentation by dynamic programming. In *COLING-ACL '98* (36th Annual Meeting of the Association for Computational Linguistics and the 17th International Conference on Computational Linguistics, COLING-ACL '98) (pp. 1484-1486). Montreal, Canada: Université de Montreal.

Mannila, H.; Toivonen, H.; & Verkamo, A. I. (1995). Discovering frequent episodes in sequences. In U. M. Fayyad & R. Uthurusamy (Eds.), *Proceedings of the First International Conference on Knowledge Discovery and Data Mining* (August 1995, Montreal, Canada) (pp. 210-215). Menlo Park, CA: AAAI Press.