Finding Co-occurring Text Phrases by Combining Sequence and Frequent Set Discovery

Helena Ahonen-Myka
Universität Tübingen
Wilhelm-Schickard-Institut für Informatik
Sand 13, D–72076 Tübingen, Germany
helena.ahonen@acm.org

Oskari Heinonen
Mika Klemettinen
A. Inkeri Verkamo
University of Helsinki
Department of Computer Science
P.O. Box 26 (Teollisuuskatu 23)
FIN–00014 University of Helsinki, Finland
{oheinone,mklemett,verkamo}@cs.helsinki.fi

Abstract

A significant amount of data resides in loosely structured text collections. The concept of text mining has recently been introduced in order to utilize these resources in data mining driven decision making. In our approach, we consider finding multi-term text phrases that tend to co-occur in the documents of a document collection. We combine and further develop two techniques, finding frequent sequences and finding frequent sets, and discuss their suitability for text mining. The process presented in this paper contains two major phases. In the first phase, maximal frequent sequences are extracted from documents, i.e., such sequences of words that are frequent in the document collection and that are not contained in any other longer frequent sequence. A sequence is considered to be frequent if it appears in at least σ documents, when σ is a given frequency threshold. For instance, we may require the sequences to occur in at least 10 documents. In the second phase, co-occurrences of the maximal frequent sequences are found by discovering frequent sets of the sequences, i.e., which sequences tend to co-occur in several documents. We have implemented the methods and experimented with a news collection. The experiments reveal many characteristics of textual data, which affect the further development and application of the methods.

1 Introduction

A significant amount of data resides in loosely structured text collections. Traditional text retrieval queries can be used for locating documents that contain given words or phrases, but even for a subject area specialist it is tedious and time-demanding to spot new interesting events and trends, both regularities and exceptions, in daily expanding document collections. The concept of *text mining* has recently been introduced in order to offer tools for utilizing text resources in data mining driven decision support. For instance, sequential patterns have been used in the PatentMiner system for discovering trends among patents [Lent *et al.*, 1997]; we have also studied and applied a similar kind of technique (see, e.g., [Ahonen *et al.*, 1997; 1998]). Association rule driven systems like KDT [Feldman *et al.*, 1996] and Document Explorer [Feldman *et al.*, 1997] do not consider full text as input, but use only key words attached to each document. We combine and develop these approaches, on the one hand, to extract suitable phrases from the documents for further processing and knowledge discovery, and on the other hand, to find co-occurrences among these phrases.

The strength of our method is that it employs a versatile technique for finding complex text phrases from full text, allowing, if desired, gaps between the words and variable word order in the phrases. The method itself does not restrict the types of phrases to, e.g., noun phrases, but all words and word forms can be considered. It is possible, however, to apply restrictions if they appear to be necessary. Although indexing and selecting key words are well-studied within information retrieval, these more flexible kind of phrases provide a possibility to improve the quality of retrieval and access to textual data.

In our approach (see Figure 1), we consider finding multi-term text phrases that tend to co-occur in the documents of a document collection. Before the discovery phase, the documents are preprocessed, which also includes removing a set of common and uninteresting words. To find co-occurring phrases, we first create a set of word sequences, or text phrases, for each single document. A set of phrases can be regarded as a content

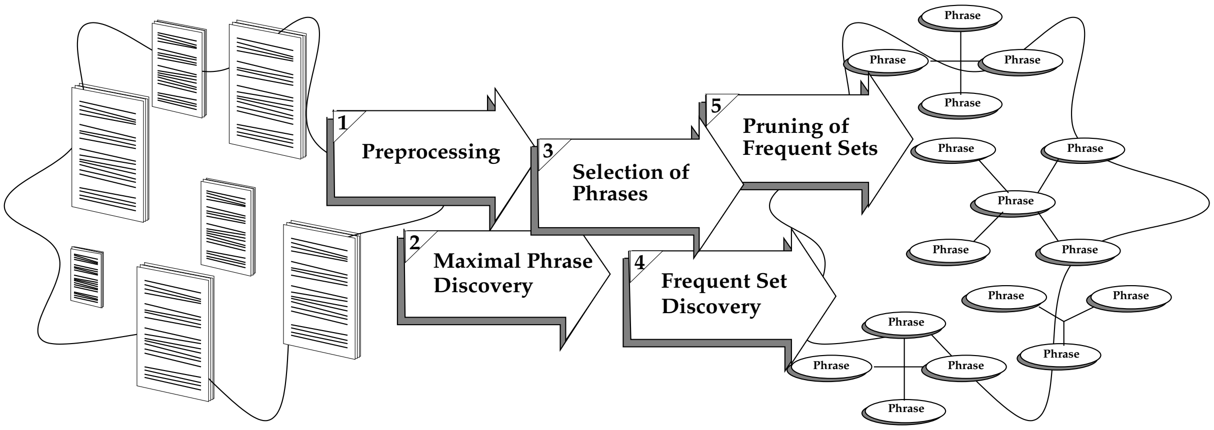

Figure 1: Phrase discovery from a document collection.

descriptor that should distinguish the document from other documents in the collection. Specifically, we discover *maximal frequent sequences*, i.e., such sequences of words that are frequent in the document collection and that are not contained in any other longer frequent sequence. A sequence is considered to be frequent if it appears in at least σ documents, when σ is a given frequency threshold. For instance, we may require the sequences to occur in at least 10 documents.

From these sets of descriptive phrases, we compute frequent sets, i.e., sets of phrases that occur together in the same documents frequently enough (as stated by a given frequency threshold). To reduce the number of frequent sets in large document collections, we utilize the concept of *equivalence class*: an equivalence class consists of phrases that are descriptive of almost the same documents. Using this concept, all frequent sets are scanned and phrases belonging to some equivalence class are replaced by the name of the class. Duplicate sets can now be removed.

The text mining process from preprocessing the data up to the utilization of discoveries is based on our framework [Ahonen *et al.*, 1997; 1998] which follows the general knowledge discovery process with some extensions; we have shown that general data mining methods are applicable to text analysis tasks such as descriptive phrase extraction. We have also shown that by shifting the focus in the preprocessing phase, based on the morphological analysis of the words and the selection of appropriate word classes, data mining can be used to obtain results for various purposes. For instance, we analysed Finnish legal texts with the help of a morphological analyser program, which gave us the base form and the morphological analysis of each word (without disambiguation). With only a selected number of word classes—e.g., nouns, proper nouns, and adjectives—we searched for both phrases and terms with free word order, i.e., co-occurring terms. Furthermore, we expanded our approach to cover more detailed text and language analysis, where we considered the morphological information only.

Our current approach using a combination of sequence and frequent set discovery has been tested on the Reuters newswire collection. The number of resulting co-occurring phrases and also their quality is reasonable.

The paper is organized as follows. In Section 2 the basic definitions concerning word sequences (phrases) and frequent sets are given. Then, in Section 3, we present our phrase co-occurrence extraction process in detail. Experiments are described in Section 4. Finally, Section 5 is a short conclusion.

2 Frequent Word Sequences and Frequent Sets

Assume S is a set of documents, and each document consists of a sequence of words.

Definition 1 *A sequence $p = a_1 \cdots a_k$ is a* subsequence *of a sequence q if all the items $a_i, 1 \le i \le k$, 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.*

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

Note that we only count one occurrence of a sequence in a document: several occurrences within one document do not make the sequence more frequent.

Definition 3 *A sequence p is a* maximal frequent (sub)sequence *in 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 the following, we use the term *phrase* to denote a word sequence that is a subsequence of some document.

Assume that we have discovered the maximal frequent phrases for each document in the collection. In a large collection of documents, some phrases, or even groups of phrases, may be descriptive of several documents. On the other hand, some phrases may seldom or never occur together, and hence they may be used for distinguishing documents. As an example, the phrases *data mining* and *knowledge discovery* may often be discovered in the same documents, while neither of them would probably be considered descriptive of the same document as, say, the phrase *device handler*. The co-occurrence of phrases can be characterized as frequent sets [Agrawal *et al.*, 1996]. Each document corresponds to a row in the document database, and is represented by a set of descriptive phrases.

Definition 4 *Let $R = \{A_1, \ldots, A_n\}$ be an ordered set, such that each A_i is a set of phrases descriptive of document i. A* frequent set *is a set of phrases that co-occur at least in σ sets in R, where σ is a given frequency threshold.*

Note that in different phases of the process, different frequency thresholds may be used.

3 Discovery of Co-occurring Text Phrases

The steps of the discovery process for finding co-occurring text phrases are the following:

1. Preprocessing
2. Maximal frequent phrase discovery
3. Computing frequent sets for the phrases
4. Pruning the collection of frequent sets
5. Attaching the frequent phrases and frequent sets to documents

The steps of the process are described in more detail in the following sections.

3.1 Text Preprocessing

There are certain special aspects in the preprocessing of text. Text consists of words, special characters, and structural information, and the preprocessing required depends heavily on the intended use of the results. In our case, all other elements except the words themselves are pruned away.

Further filtering of the data can be used to focus our discovery phase, e.g., to limit the number of results so that we are not overwhelmed by phrases consisting entirely of uninteresting words. For this purpose, we use a small stopword list consisting of definitely uninteresting items such as articles, pronouns, conjunctions, common adverbs, and non-informative verbs (e.g., *be*).

3.2 Extracting Descriptive Phrases

The goal of this phase is to find all maximal frequent phrases in the document collection. The outline of the method is presented in Algorithm 1; a more detailed description of the method can be found in [Ahonen, 1999a; 1999b].

Initial Phase: Collect All Frequent Pairs

In the *initial phase* of Algorithm 1 (steps 1–3) we collect all the ordered pairs, or 2-grams, (A, B) such that words A and B occur in the same document in this order and the pair is frequent in the document collection. Moreover, we restrict the distance of the words of a pair by defining a maximal gap; in our experiments we used a maximal gap of 2, meaning that at most 2 other words may be between the words of a pair.

Discovery Phase: Extract Maximal Phrases

The maximal frequent phrases are extracted in the *discovery phase* of Algorithm 1 (steps 4–19), which combines bottom-up and greedy approaches. A straightforward bottom-up approach is inefficient, since the basic version of the algorithm requires as many levels as is the length of the longest maximal phrase. When long maximal phrases exist in the collection, this can be prohibitive, since on every level the join operation increases exponentially the number of the grams contained in the maximal phrases. For example, in our experiments (19000 documents, frequency threshold 10) the longest phrase contained 25 words. In the straightforward approach we would have had to go through all the 25 levels, while using the combined approach we only have to search for the first four levels; although the greedy approach increases the workload during the first passes, the gain in efficiency is still substantial.

Algorithm 1 *Discovery of all maximal frequent subphrases in the document collection.*

Input: *S: a set of documents, σ: a frequency threshold*
Output: *Max: the set of maximal frequent phrases*

```
    // Initial phase: collect all frequent pairs.
1.  For all the documents d ∈ S
2.      collect all the ordered pairs within d
3.  Grams_2 = all the ordered pairs that are frequent in S
    // Discovery phase: build longer phrases by
    // expanding and joining.
4.  k := 2
5.  Max := ∅
6.  While Grams_k is not empty
7.      For all grams g ∈ Grams_k
8.          If g is not a subphrase of some m ∈ Max
9.              If g is frequent
10.                 max := Expand(g)
11.                 Max := Max ∪ max
12.                 If max = g
13.                     Remove g from Grams_k
14.             Else
15.                 Remove g from Grams_k
16.     Prune(Grams_k)
17.     Grams_{k+1} := Join(Grams_k)
18.     k := k + 1
19. Return Max
```

Expand We start from a set of frequent pairs. The occurrences of frequent pairs are stored; e.g.

AB: [11-12][31-32]

AC: [11-13][31-33]

BC: [12-13][22-23][32-33][42-43][52-53]

We take a pair and *expand* it by adding items to it, in a greedy manner, until the longer phrase is no more frequent. The occurrences of longer phrases are computed from the occurrences of pairs. All the occurrences computed are stored, i.e., the computation for ABC may help to compute later the frequency for $ABCD$. In the same way, we go through all pairs, but we only try to expand a pair if it is not already a subphrase of some maximal phrase, which guarantees that the same maximal phrase is not discovered several times. When all the pairs have been processed, every pair belongs to some maximal phrase. If some pair cannot be expanded, it is itself a maximal phrase. If we knew that every maximal phrase contains at least one unique pair, which distinguishes the phrase from the other maximal phrases, then one pass through the pairs would discover all the maximal phrases. As this cannot be guaranteed, the process must be repeated iteratively with longer k-grams.

In the expansion step (step 10) of Algorithm 1, all the possibilities to expand have to be checked, i.e., at any point, the new item can be added to the tail, to the front or in the middle of the phrase. If one expansion does not produce a frequent phrase, other alternatives have to be checked. The expansion is greedy, however, since after expanding successfully it proceeds to continue expansion, rather than considers alternatives for the expansion. The choice of items to be inserted is restricted by the k-grams, i.e., also after expansion the sequence is constructed from the existing k-grams.

Join In the next step, we *join* pairs to form 3-grams, e.g., if there exist pairs AB, BC, and BD, we form new phrases ABC and ABD. Then we make a pass over all these 3-grams, and, as with the pairs, we try to expand grams that are not subphrases of the known maximal phrases and that are frequent. We can always remove those grams that are themselves maximal phrases, since such a gram cannot be contained in any other maximal phrase. The discovery proceeds respectively, variating expansion and join steps, until there are no grams left in the set of grams.

Prune Often it is not necessary to wait until the length of the grams is the same as the length of the maximal phrase, in order to remove a gram from the set of grams. After a discovery pass over the set of grams, every gram is a subphrase of at least one maximal phrase. Moreover, any new maximal phrase that can be generated has to contain grams either from at least two maximal phrases or two grams from one maximal phrase in a different order than in the existing maximal phrase. Otherwise a new phrase would be a subphrase of an existing maximal phrase.

This motivates the *pruning* phase of the algorithm, which proceeds as follows. For every gram it is checked how the gram might join existing maximal phrases to form new phrases. If a new candidate phrase is not a subphrase of some existing maximal phrase, all subphrases of the candidate phrase are considered in order to find new frequent phrases that are not contained in any maximal phrase, remembering that if a phrase is not frequent, its superphrases cannot be frequent. If a frequent phrase is found, all its grams are marked. After all grams are processed, grams that are not marked are removed from the gram set. In the experiments, this pruning approach has been shown to be rather vulnerable for cases, where a set of several maximal phrases shares a longer subphrase, since a large number of subphrases has to be checked. Therefore, the development of the algorithm in the near future will concentrate on speeding up this part.

Extract frequent subphrases Above, a method for discovering maximal frequent phrases was presented. After all the maximal frequent phrases have been discovered, also their subphrases which are more frequent than the corresponding maximal phrase can be found efficiently. The more frequent subphrases are found for each maximal phrase in turn. First, a set of grams is formed from the pairs contained in the maximal phrase. For instance, if we are considering a maximal phrase $ABCD$, 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 phrase, i.e., in this case higher than 2. As a subphrase can also be a subphrase of some other subphrase of the maximal phrase, it is only considered interesting, if it is more frequent than any of the other subphrases that contain it. Hence, a 2-gram is included in the set of more frequent subphrases, if it is more frequent than the maximal phrase 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. 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 phrase.

3.3 Computing Co-occurrences of Phrases

In order to find out which phrases co-occur in the documents, frequent sets are formed from the descriptive phrases discovered in the previous phase. Input for the frequent set discovery consists of one line per each document, containing the descriptive phrases of the document.

The frequent set discovery method we have used is a straightforward apriori-like approach [Mannila *et al.*, 1994; Agrawal *et al.*, 1996]. First, all the frequent pairs are found from the input, i.e., all the pairs such that both of the phrases appear in at least σ documents, where σ is the frequency threshold given. Then larger sets are combined from the shorter ones, level by level, using the knowledge that a set of size k can only be frequent if all its $k-1$-subsets are frequent.

At least in our setting, the number of sets produced is rather large, due to the number of phrases that appear always or often together. Hence, a set of pruning steps is considered. First, we partition the phrases into equivalence classes: phrases X and Y belong to the same class if they are descriptive of almost the same documents. To find the equivalence classes, we scan the frequent pairs of phrases and if a pair almost always occurs together, they belong to the same equivalence class. Hence, if the frequency of the pair (A, B) (in the collection) divided by the frequency of A is greater than a given threshold, we add the phrase B to the set Det_A, consisting of the phrases determined by A. The phrase A itself is also included in Det_A. The equivalence class Eq_A of A will now contain all phrases X such that $Det_X = Det_A$. In our experiments we have used a confidence threshold of 0.9.

3.4 Pruning and Attaching Frequent Sets and Phrases to Documents

The equivalence classes as such already provide strong information about the phrases, but their ability to reduce the amount of frequent sets is even more important. Pruning proceeds as follows: in a frequent set, all the elements of an equivalence class are combined and renamed with the name of the equivalence class, and all duplicates are removed. Hence, the sets both shrink and their amount is decreased.

Another way of pruning the frequent sets is based on pruning small sets that are contained in some larger set, unless the smaller set is more frequent than the larger set. If the sets $\{A, B, C\}$, $\{A, B\}$, and $\{A, C\}$ all exist in the collection, the frequency of the latter two sets is always larger or equal to the frequency of the larger set. If the frequency of the set $\{A, B\}$ is not larger than the frequency of $\{A, B, C\}$, the smaller set does not add any information, and hence can be pruned away.

As, for each maximal frequent phrase and each frequent set, the set of corresponding documents is known, it is straightforward to collect all the phrases and sets of one document. These phrases and sets form a content descriptor of the document, which can be used for further processing in various applications.

More concrete examples of the presented steps and the resulting sets can be found in the next section.

4 Experiments

We have implemented the maximal frequent phrase and frequent set discovery algorithms in Perl. In this section we shortly describe phases of a process with the publicly available Reuters-21578 news collection[1]. The phases with their operations and results are summarised in Table 1. The hardware environment was Sun Enterprise 450 with 1 GB of main memory.

4.1 Data Preprocessing

The Reuters-21578 news collection contains about 19000 documents. The average length of the documents is 135 words. Originally, the documents contain 2.5 million words. After stopword pruning against a list of 400 stopwords, the amount of words is reduced to 1.3 million words, which are instances of 50,000 distinct words.

[1]http://www.research.att.com/~lewis/reuters21578.html

The list of stopwords contains, e.g., single letters, pronouns, prepositions, some for the Reuters collection common abbreviations (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.

4.2 Phrase Discovery

We have performed experiments using a frequency threshold of 15 for phrase discovery. In the initial phase, 12,071 frequent pairs were found. The pairs had altogether 427,775 occurrences, giving an average of 23 frequent pairs in one document. Additionally, there were 406 documents without any frequent pairs. Table 2 presents some performance figures of the execution of phrase discovery with the frequency threshold 15. Before the first pass over the set of grams, 2.36 min was needed to construct the initial data structures. As can be seen from the table, most of the time is spent during the first pass (or level 2), when also most of the maximal phrases, even the longer ones, are extracted. Another time consuming phase is the first pruning phase (level 4), basically due to a handful of maximal phrases that share a long common part. The number of maximal frequent phrases of various sizes are shown in Table 3.

As an example of phrases discovered for one document, consider the following phrases.

immediately_after	26
effective_april	63
company's_operations	20
unit_power	16
early_week	42
senior_management	28
nuclear_power_plant	26
–power_plant	55
–nuclear_power	42
–nuclear_plant	42
regulatory_commission	34
#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 phrases is based on frequency, it is possible that the phrases extracted tell more about the domain than about the details covered in this specific story.

In addition to the discovery of maximal frequent phrases, also more frequent subphrases were found. Instead of accepting subphrases that are only slightly more frequent, it was required that a subphrase occurs at least in seven additional documents compared to the corresponding maximal phrase. In the experiments, the number of maximal phrases that had subphrases which were more frequent than the maximal phrase itself was 2,264,

Phase	Operation	Result
0	*Original material.*	19,000 documents, 2.5 million words 135 words/document on average
1	*Preprocessing.* Removal of stopwords (400 distinct).	1.3 million words, 50,000 distinct words 70 words/document on average
2	*Phrase discovery.* Frequency threshold 15, gap size 2.	9,625 maximal frequent phrases
3	*Frequent set discovery.*	
3.1	*Frequent pairs generation.* Frequency threshold: 10 occurrences.	3,076,883 frequent pairs
3.2	*Equivalence class computing.* Class members are descriptive of almost the same documents. Confidence threshold: 0.9.	54 equivalence classes, 2–14 phrases/class
3.3	*Frequent set generation.* Frequency threshold: 10 occurrences.	194,907 frequent sets
4	*Pruning.*	
4.1	*Pruning I.* Combine phrases in an equivalence class and remove duplicate sets.	11,667 frequent sets
4.2	*Pruning II.* Remove subsets that do not add information w.r.t. their supersets.	1,791 frequent sets

Table 1: Summary of the phases of the experiment.

Level	2	3	4	5	6	7	8	9
Pass time over the set of grams (min)	26.12	2.33	0.40	0.04	0.01	0.03	0.02	0.00
Join time (min)	3.44	0.24	0.02	0.02	0.01	0.01	0.01	0.00
Prune time (min)	-	-	38.16	4.38	2.27	2.03	0.47	0.06
Grams in the set	12,071	4,451	2,934	343	361	501	373	67

Table 2: Performance figures of the phrase discovery phase.

Length	2	3	4	5	6	7	8	9	10	11	12	13
$\sigma = 15$	7,664	1,320	353	146	65	17	8	4	13	12	13	5

Length	14	15	16	17	18	19	20	21	22	23	24	25
$\sigma = 15$		1	1		1				2			

Table 3: The number of maximal phrases of various lengths.

with the average of two subphrases. In the sample above, all the possible subphrases of the phrase "nuclear power plant" are more frequent. The phrase "#electric co" is an example of a phrase that is a subphrase of some maximal phrase that does not occur in this document.

4.3 Frequent Set Discovery and Pruning

After the discovery of phrases for each document, the frequent sets of maximal phrases were formed. The input for the frequent set discovery contained one line per document, like in the following.

afternoon_session lower_rates current_conditions ...
mine_production ounces_silver u.s._imports ...
tonnes_free_market tonnes_maize ec_commission ...

In the experiments, the frequency threshold was 10, i.e., the phrases of a frequent set have to appear together in at least 10 documents. When frequent sets were generated from the input (phase 3.3, Table 1), the result contained 194,907 sets. For the first pruning step, equivalence classes were computed from the frequent pairs, resulting in 54 classes with 2–14 phrases each. Some examples of the equivalence classes ($\{\cdot\}_{\mathsf{Eq}}$) can be seen in the following.

$\{$tonnes_canadian_rapeseed, japanese_crushers_bought$\}_{\mathsf{Eq}}$
$\{$transcanada_pipelines, dome_petroleum_ltd$\}_{\mathsf{Eq}}$
$\{$interest_rate, interest_futures$\}_{\mathsf{Eq}}$
$\{$resigned_chief_executive, resigned_chief_officer_ltd$\}_{\mathsf{Eq}}$

The largest equivalence classes typically contain slightly differing versions of the same phrases, see Figure 2 for an example.

Combining the phrases of each equivalence class, as explained in Section 3.3, the number of frequent sets decreased into 11,667 sets. Furthermore, subsets which were not more frequent than any frequent set that contained them were pruned. After that, 1,791 frequent sets remained. Finally, for each document the corresponding frequent sets were attached to the set of phrases for the document. From the total of 19,000 documents, 7,767 documents could be described by at least one frequent set. For instance, one document contained the following frequent phrases with frequencies over the whole document collection.

cyclops_corp	31
co_market	17
received_offer	30
acquisition_corp	71
stock_corp	35
market_stock	52
jefferies_co	18
audio_video_affiliates_inc	16
–video_inc	26
#offer_share	81
#co_stock	28

For this document, one frequent set over the whole document collection was found, namely,

$\{$audio_video_affiliates_inc,
$\{$cyclops_corp, tender_offer_cyclops$\}_{\mathsf{Eq}}\}$ 13

That is, "audio_video_affiliates_inc" occurred together with either "cyclops_corp" or "tender_offer_cyclops" in 13 documents. Note that one of the phrases in the equivalence class does not occur in this document, due to the generalisation effect of the equivalence class computation. More frequent sets would be discovered if more frequent subphrases of maximal phrases were considered as well. This, however, would also necessitate more effective pruning.

4.4 Result Analysis and Discussion

The final goal of our study is to find automatic ways to represent documents in a compact form which can be used both in computations, e.g., in clustering, and in user interfaces. In the latter case, the representation—in a suitably rendered form—could act as a mediating level between a user and fulltext documents. Moreover, the representation should facilitate further knowledge discovery, e.g., discovering trends and surprising events.

The most important result of the experiments is that although the starting point is unrestricted text, the number of generated phrases and frequent sets is not overwhelming but rather reasonable. Hence, for each document we can generate frequent phrases and sets that do not contain too much overlapping information.

As the discovery of maximal frequent phrases is time-demanding, the frequency threshold of that phase has to be in the range of 10–15. In documents most words, and hence also phrases, occur rather seldom, which means that a threshold of, e.g., 15 already neglects many phrases. Therefore, the maximal phrases generated tend more to characterize the topics, or domains, of the documents, not the single events that are specific to this document. Hence, as a base for the discovery of surprising information, the maximal phrases may not be useful as such. However, they might facilitate creating knowledge about domains, which knowledge in turn might help identifying phrases that are somehow unexpected.

In our experiments, the frequency threshold for discovery of frequent sets was 10. As the amount of data in this phase is much smaller than in the previous phase, and the process also as such is easier, the threshold could probably be set much lower. Based on our results, frequent sets of phrases seem a promising way to characterize relationships of phrases in a document, and in this way it is possible to find an inner structure for the set of phrases, e.g., to define which phrases form a domain.

Both in the phrase discovery phase and in the set discovery phase, most problems are caused by the fact that in the documents the same information is said in slightly varying ways, generating possibly long phrases that have small variations. These variations do not add to the information value of the phrase set of a document, and

debentures_company's_common_share_premium_price_debt_years_issue_rated
debentures_convertible_company's_stock_share_representing_premium_stock_price_terms...
debentures_company's_common_stock_share_premium_price_set_non-callable_issue_rated
debentures_company's_common_stock_share_premium_price_debt_set_non-callable_years...
debentures_company's_common_stock_representing_stock_terms_debt_set_non-callable...
debentures_company's_common_stock_share_premium_price_terms_non-callable_years_issue...
debentures_company's_common_share_premium_stock_debt_set_non-callable_years_issue...
debentures_company's_common_stock_share_premium_price_terms_set_non-callable_years...
debentures_company's_common_stock_representing_stock_terms_debt_set_non-callable_years...
debentures_company's_common_share_premium_price_terms_debt_years_issue_rated
debentures_company's_common_stock_representing_stock_price_debt_non-callable_years...
debentures_company's_common_stock_representing_stock_price_set_non-callable_years...

Figure 2: An example of a large equivalence class.

they may generate a large amount of frequent sets. Partially, the equivalence classes presented solve the problem, as these variations usually end up in the same class. However, a better solution might be to combine these phrases already after the phrase discovery phase, e.g., by representing them as regular expressions.

There are also some other aspects that could be taken into consideration in developing the method. Frequent single words for each document could be combined with phrases in the frequent set discovery, to complement the characterization of documents. Also names (e.g., of people, companies, and places) could be found separately and included in the frequent set discovery phase. Efficient methods for extracting names exist; our method may sometimes ignore names, since they do not appear in one document frequently enough, at least not in the same form.

5 Conclusion

We consider finding multi-term text phrases that tend to co-occur in the documents of the document collection. The strength of our method is that it employs a versatile technique for finding complex text phrases allowing gaps.

To find the phrases, we first create for each document a set of word sequences, or text phrases. Specifically, we discover maximal frequent sequences, i.e., such sequences of words that are frequent in the document collection and that are not contained in any other longer frequent sequence. From these sets of text phrases, frequent sets are computed. As there can easily be document collections with a large number of documents, also the number of frequent sets can be quite prohibitive. To alleviate this problem, we utilize the concept of equivalence classes in limiting the number of resulting sets.

The approach has been tested with a collection of news stories. The number of resulting co-occurring phrases and also their quality is reasonable. The experiments reveal many characteristics of textual data, which affect the further development and application of the methods.

Acknowledgments

This work has been partially supported by the EC/TMR Marie Curie research training grant #ERBFMBICT972821 (Helena Ahonen-Myka), and grants by the 350th Anniversary Foundation of the University of Helsinki (Oskari Heinonen) and the Nokia Foundation (Mika Klemettinen).

References

[Agrawal *et al.*, 1996] Rakesh Agrawal, Heikki Mannila, Ramakrishnan Srikant, Hannu Toivonen, and A. Inkeri Verkamo. Fast discovery of association rules. In Usama M. Fayyad, Gregory Piatetsky-Shapiro, Padhraic Smyth, and Ramasamy Uthurusamy, editors, *Advances in Knowledge Discovery and Data Mining*, pages 307–328. AAAI Press, Menlo Park, California, USA, 1996.

[Ahonen *et al.*, 1997] Helena Ahonen, Oskari Heinonen, Mika Klemettinen, and A. Inkeri Verkamo. Mining in the phrasal frontier. In Jan Komorowski and Jan Zytkow, editors, *Proceedings of the First European Symposium on Principles of Data Mining and Knowledge Discovery (PKDD'97)*, number 1263 in Lecture Notes in Artificial Intelligence, pages 343–350, Trondheim, Norway, June 1997. Springer-Verlag.

[Ahonen *et al.*, 1998] Helena Ahonen, Oskari Heinonen, Mika Klemettinen, and A. Inkeri Verkamo. Applying data mining techniques for descriptive phrase extraction in digital document collections. In *Advances in Digital Libraries (ADL'98)*, Santa Barbara, California, USA, April 1998. IEEE Computer Society Press.

[Ahonen, 1999a] Helena Ahonen. Finding all maximal frequent sequences in text. In *ICML99 Workshop, Machine Learning in Text Data Analysis*, Bled, Slovenia, 1999.

[Ahonen, 1999b] Helena Ahonen. Knowledge discovery in documents by extracting frequent word sequences.

Library Trends, 1999. Special Issue on Knowledge Discovery in Databases, to appear.

[Feldman *et al.*, 1996] R. Feldman, I. Dagan, and W. Klösgen. Efficient algorithms for mining and manipulating associations in texts. In *Cybernetics and Systems, Volume II, The Thirteenth European Meeting on Cybernetics and Systems Research*, Vienna, Austria, April 1996.

[Feldman *et al.*, 1997] Ronen Feldman, Willi Kloesgen, and Amir Zilberstein. Document Explorer: Discovering knowledge in document collections. In Zbigniew W. Ras and Andrzej Skowron, editors, *Proceedings of Tenth International Symposium on Methodologies for Intelligent Systems (ISMIS'97)*, number 1325 in Lecture Notes in Artificial Intelligence, pages 137–146, Charlotte, North Carolina, USA, October 1997. Springer-Verlag.

[Lent *et al.*, 1997] Brian Lent, Rakesh Agrawal, and Ramakrishnan Srikant. Discovering trends in text databases. In David Heckerman, Heikki Mannila, Daryl Pregibon, and Ramasamy Uthurusamy, editors, *Proceedings of the Third International Conference on Knowledge Discovery and Data Mining (KDD'97)*, pages 227–230, Newport Beach, California, USA, August 1997. AAAI Press.

[Mannila *et al.*, 1994] Heikki Mannila, Hannu Toivonen, and A. Inkeri Verkamo. Efficient algorithms for discovering association rules. In *Knowledge Discovery in Databases, Papers from the 1994 AAAI Workshop (KDD'94)*, pages 181–192, Seattle, Washington, USA, July 1994.