ISSN:2249-5789
Alekhya V et al , International Journal of Computer Science & Communication Networks,Vol 2(5), 601-606

Descriptive Phrase Extraction in Text mining

Alekhya V[1], B. Govinda Laxmi[2]

*(M.Tech, Department of CSE, Sri Sivani college of engineering, Andhra Pradesh, India)
** (Associate professor, Department of CSE, Sri Sivani college of engineering, Andhra Pradesh, India

Abstract

Recently various algorithms have been proposed for text documents to mining frequent patterns. But how to efficiently find these patterns is still an open issue in text mining domain. Traditionally, texts have been analyzed by using various information retrieval related methods, such as full-text analysis, and natural language processing. However, only few examples of data mining in text, particularly in full text, are available. In this paper we present a framework for text mining using descriptive phrase extraction. The framework follows the general knowledge discovery process, thus containing steps from preprocessing to the utilization of the results. We apply generalized episodes and episode rules data mining method. We introduce a weighting scheme that helps in pruning out redundant or non-descriptive phrases. Several experiments have been conducted on various data sets to calculate the performance of the proposed technique.

Keywords: Text Mining, Knowledge Discovery, Data Mining, Pattern Mining

1. Introduction

Huge amount full-text document collections are available for end user. The user may require an overall view of the text document collection such as which topics are covered, what kind of documents exists, and so on. In some cases, user may need to search a specific piece of data in the document. On the other hand, some users may be interested in the language itself, e.g., in word usages or linguistic structures. Hence in recent years, knowledge discovery and data mining have attracted a great deal of attention with an imminent need for turning such data into useful information and knowledge. Many applications, such as market analysis and business management, can benefit by the use of the information and knowledge extracted from a large amount of data. Knowledge discovery can be viewed as the process of nontrivial extraction of information from large databases, information that is implicitly presented in the data, previously unknown and potentially useful for users. Recently, we have seen the exuberant appearance of very large heterogeneous full-text document collections, available for any end user. The variety of users' wishes is broad. The user may need an overall view of the document collection: what topics are covered, what kind of documents exist, are the documents somehow related, and so on. On the other hand, the user may want to find a specific piece of information content. At the other extreme, some users may be interested in the language itself, e.g., in word usages or linguistic structures. A common feature for all the tasks mentioned is that the user does not know exactly what he/she is looking for. Hence, a data mining approach should be appropriate, because by definition it is discovering interesting regularities or exceptions from the data, possibly without a precise focus.

Surprisingly enough, only a few examples of data mining in text, or text mining, are available. The most notable are the KDT system [1] and Document Explorer [2] used in mining Reuters news articles. Their approach, however, requires a substantial amount of background knowledge, and is not applicable as such to text analysis in general. An approach more similar to ours has been used in the Patent- Miner System for discovering trends among patents [3]. Traditionally, texts have been analyzed by using various information retrieval related methods, such as full-text analysis, and natural language processing. However, only few examples of data mining in text, particularly in full text, are available. In this paper we present a framework for text mining using descriptive phrase extraction. The framework follows the general knowledge discovery process, thus containing steps from preprocessing to the utilization of the results. We apply generalized episodes and episode rules data mining method [28]. We introduce a weighting scheme that helps in pruning out redundant or non-descriptive phrases. The rest of the paper is organized as section 2: discuss about the related work, section 3: presents the

Proposed Framework, section 4: discuss about Experimental setup, section 5: concludes the paper.

2. Related Work

In [13], weighting scheme is used for text representation in Rocchio classifiers. Later global IDF and entropy weighting scheme is proposed in [9] and improves performance by an average of 30 percent which are based on bag of words approach. The disadvantage of this approach is how to select a limited number of features among a large set to increase the efficiency and avoid overfitting [22]. To reduce the number of features, many approaches have been proposed such as Information Gain, Mutual Information, Chi-Square, Odds ratio, and so on. M.F. Caropreso [8] proposed a technique based on the combination of unigram and bigrams was chosen for document indexing in text categorization (TC) and it is evaluated on a variety of feature evaluation functions (FEF). H. Ahonen [7], presented an algorithm for text analysis by extracting co-occurring terms as descriptive phrases from document collections. Due to lower consistency of assignment and lower document frequency for terms the proposed algorithm does not improve the performance of the system. Hierarchical clustering [18], [19] used to derive the synonymy and hyponymy relations between keywords. The pattern evolution technique was introduced in [16] to improve the performance of term-based ontology mining. Apriori-like algorithms [6], [20], [24], PrefixSpan [21], [27], FP-tree [10], [11], SPADE, SLPMiner [23], and GST [12] have been proposed based on pattern mining. All the existing techniques mainly focus on developing efficient mining algorithms to discover patterns from a large data collection. However, searching for useful and interesting patterns and rules was still an open issue [14], [15], [17]. To overcome these issues, closed sequential patterns have been proposed for text mining in [25], concept of closed patterns in text mining improved the performance of text mining. Y. Li [17] proposed a two-stage model by combining term and pattern-based methods to improve the performance of information filtering.

3. Framework

In the proposed model, we consider text as sequential data in many respects similar to the data collected by various systems. Figure 1 describes the general knowledge discovery process for text processing. In the proposed model, information is presented as episodes and episode rules.

3.1 Episodes

Episode rules and episodes are modified concept of association rules and frequent sets which is applied to text. Text is a collection of feature vector and index, where feature vector consists of an ordered set of features and index contains information about the position of the word in the sequence. It is common practice that the sequence is represented in an increasing order of the indices. A feature can be

a word; e.g., base form, inflected word form, stem,
a grammatical feature; e.g., part of speech, case, number,
a punctuation mark or other special character, or
an SGML structure tag.

A text episode is defined as a pair $\alpha = (V, <)$ where V is a collection of feature vectors, and $<$ is a partial order on V. Given a text sequence S, a text episode $\alpha=(V,<)$ occurs within S if there is a way of satisfying the feature vectors in S using the feature vectors in V so that the partial order $<$ is respected. The feature vectors of V can be found within S in an order that satisfies the partial order $<$.

For an occurrence of the episode to be interesting, all feature vectors of the episode must occur close enough in S. What is close enough is defined by giving a limit, the window size W, within which the episode must occur. Hence, instead of considering all occurrences of the episode in S, we only examine occurrences within substrings S' of S where the difference of the indices of the feature vectors in S' is at most W. Moreover, since there may be several partially differing occurrences of the episode within the substring S', we restrict ourselves to the distinct minimal occurrences of the episode, for a formal definition of a minimal occurrence [5].

For example, take a text sequence where the feature vector contains the base form of the word, the part of speech, and the number of the words. The text knowledge discovery in databases is presented by the sequence

(*knowledge*_N_SG, 1) (*discovery*_N_SG, 2)
(*in*_PP, 3) (*database*_N_PL, 4)

For a window size of 2, this sequence contains the episode (knowledge_N_SG, discovery_N_SG), but does not contain the episode (knowledge_N_SG, database_N_PL).

The most useful types of partial orders are total orders and trivial partial orders, in total orders the feature vectors of each episode have a fixed order, such episodes are called serial but in trivial partial orders the order is not significant at all, such episodes are called parallel. The support of α in S with respect to a given window size W is defined as the number of minimal occurrences of α in S. Usually, we are only interested in episodes with a support exceeding a

given support threshold, meaning that they occur in the sequence frequently enough not to be considered accidental. An episode rule gives the conditional probability that a certain episode occurs within a window size interval, given that a subepisode has occurred (within the same or possibly smaller interval). An episode rule is defined as $\beta = \alpha[win_\beta - win_\alpha)$ where β and α are episodes, β is a subepisode of α, and win_β and win_α are window sizes, with $win_\beta < win_\alpha$. α is termed as the episode of the episode rule.

The confidence of the rule is the conditional probability that α and β occurs under the window size constraints specified by the rule. Since α includes all the feature vectors of β, we omit the feature vectors of β when representing the right-hand side of the rule. The right-hand side rule denotes the difference between α and β. This method allows us to discover serial and parallel episodes of a given support threshold and episode rules of a given confidence threshold for a collection of windowsizes with a fixed upper limit.

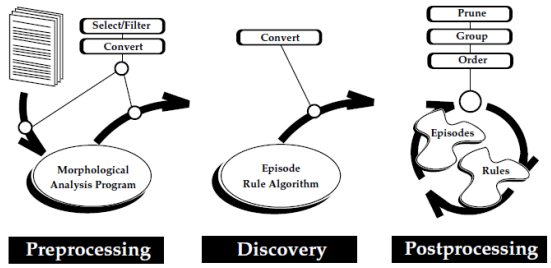

Fig 1: Knowledge discovery from textual representation into episodes and episode rules

3.2 Preprocessing the data

Figure 1 shows the preprocessing of the data. The process of obtaining useful information also relies on preprocessing the data before the discovery phase, and on postprocessing the results after the discovery phase. The preprocessing phase is very critical for efficient data processing [4]. Text consists of words, special characters, and structural information. The preprocessing required depends heavily on the intended use of the results. Typically, the data is homogenized by replacing special characters and structural information with symbols. Punctuation marks and structural information often need to be handled separately. Some of them may be ignored entirely, some of them may require special treatment, e.g., to find and process the sentence breaks. The longer distance between words of different sentences may be taken into account by inserting a larger gap in the indexing scheme [3].

Preprocessing may involve some amount of natural language analysis. Morphological analysis gives us detailed information of the data which may be included in the feature vector and to generalize the data. Filtering of the data is used to focus our discovery phase, to limit the number of results so that we are not overwhelmed by uninteresting rules, and to decrease the processing effort needed in the discovery phase. Pruning may be done either before or after the discovery phase. If we do not have a clear idea of what kind of regularities we are looking for it is often advisable to defer the pruning decisions to the post processing phase instead of heavy pruning in the preprocessing phase. On the other hand, with large collections of documents, efficiency has to be taken into account. In such cases, we should know what features are not need in the preprocessing, to limit the size of the search space and the time requirement of the discovery phase.

In the preprocessing phase, pruning can be used in two distinct ways. We may prune entire feature vectors, i.e., drop uninteresting items such as articles, prepositions, noninformative verbs, or punctuation marks, or select only some class of words (e.g., nouns) to be examined. We may also focus on some features of each word, e.g., by leaving only the base form of the word, or only part of the morphological information.

3.3 Postprocessing the results

The episode discovery method produces a large amount of episodes and episode rules. The problem is to define which episodes are sensible and useful. In data mining, typically, the relevance of episodes is very strongly dependent on the application. In our model, we assume measures of relevance are common to all documents, independent of the semantic content of the texts. However, different usage needs also effect applying the measures. Postprocessing involves pruning, grouping and ordering the results. The usability of the results may be enhanced by using knowledge on the rules and episodes of a single document and comparing it to similar information on the entire document collection.

3.4 Metrics

Length: The length of an episode is defined as how many number of feature vectors it contains. The length of a rule is the length of its episode. To scale the length we compare it to the highest possible value which is the maximal window size winmax.

$$l = \frac{len}{win_{max}} \qquad (1)$$

Tightness: Let len1 and len2 be the (absolute) lengths of the left-hand side of the rule and the entire rule, respectively, and let win1 and win2 be the window sizes of the left-hand side and the entire rule,

respectively. Furthermore, let diff be the mean of the differences of these measures, i.e.,

$$diff = \frac{(win_1 - len_1) + (win_2 - len_2)}{2} \quad (2)$$

The *tightness* t of the rule is computed as

$$t = 1 - \frac{diff}{win_{max}} \quad (3)$$

Mutual confidence: Let s1, s2, and s3 be the supports of the left-hand side, the right-hand side, and the entire rule respectively. The mutual confidence m of the rule is calculated as

$$m = \frac{(s_3/s_1) + (s_3/s_2)}{2} \quad (4)$$

The length of a phrase should be taken into account, since longer phrases should be preferred because they are more descriptive and it is always possible to reconstruct shorter phrases from longer ones. As longer phrases usually are less frequent, there may be need to compensate them in weighting. The upper limit for the window size also gives the upper limit for the length of a phrase. However, this may result in short phrases spreading out too much, since more slack is allowed within the phrase. Tightness reduces this effect and gives a way to decrease the weight of these possibly undesired phrases.

Mutual confidence attempts to reveal ties between left-hand side and right-hand side rule. The formula above prefers cases in which words appear often together and seldom with some other words. Other alternatives are possible, e.g., we might want to get phrases, in which one part is very frequent and appears with many other words. That is, we might want to find descriptive attributes to specify a frequent word. To create a ranking of rules within a document we calculate a weight for each rule combining the above measures. It can be done in several ways, depending on the emphasis we want to give to each measure. We consider the support of a rule to be a central factor, and hence, calculate the weight w by multiplying the support by a combination of the other measures.

The measures above give a weight to each phrase based on the knowledge we have about the document. Since we are interested in characteristic phrases that can also discriminate documents, we have to consider the distribution of the phrases in the collection. For instance, in the legal texts we have used, many legal terms appear in almost all documents. Hence, for each rule in a document we compute its inverse document frequency (IDF) that describes how common the rule is in the collection. The value is computed as

$$idf = -\log\left(\frac{\text{number of documents containing the rule}}{\text{number of documents in the collection}}\right) \quad (5)$$

4. Experimental Setup

To evaluate the use of knowledge discovery methods and the discovered knowledge in the context of text documents, we have made experiments with real data sets. We describe the data sets and the conversion of the original data into a suitable format for the analysis. Then, we present the experiments and their results.

4.1 Data sets and preprocessing

In the experiments we have taken 14 documents from Finnish legal texts in SGML format. The size of the documents varied between 2,500 and 60,000 words. The words includes punctuation marks, the number of real words is about 20% lower and the number of words and their morphological interpretations is about 20% higher, respectively. The SGML tags are not included in the word counts. After cleaning the data, the statutes were fed to a morphological analyser program called Fintwol (a product of Lingsoft, Inc.), which gives us the base form and the morphological analysis of each word. Note that Fintwol only looks at one word at the time and does not try to disambiguate using the word context. An example of the output is

rikoslain rikoslaki N GEN SG

which tells us that the word which occurred in the text is rikoslain, its base form is rikoslaki (in English, Criminal
Act), and the word is noun (N), genitive (GEN) and singular (SG). However, a word may have several interpretations, being even inflections of separate base forms.

Table 1: The Test Data and Results

Statute		Phrases			Pruned Phrases			Co-occ Terms				Pruned Co-occ Terms			
#	words	Epis	Size one	Rules	Epis	Size one	Rules	Epis	Size one	Epis	Size one	Epis	Size one	Epis	Size one
1	4273	39	36	3	35	32	1	37	30	118	54	35	28	81	31
2	38970	472	310	103	408	285	43	473	213	2362	425	443	197	1670	321
3	2622	29	24	7	26	21	4	28	19	61	32	25	17	48	27
4	19682	321	204	116	279	184	35	345	138	1532	271	323	125	1025	164
5	6427	90	78	14	86	74	7	74	59	1532	271	323	125	1025	164
6	61559	962	491	327	839	449	108	1090	349	4911	618	1025	315	3801	488
7	5815	87	66	17	78	58	8	90	46	311	89	78	38	238	59
8	20756	369	186	259	162	169	41	343	136	1601	241	270	122	850	189
9	3997	48	34	26	42	32	9	47	28	113	45	44	25	95	35
10	5491	59	48	14	55	46	6	57	38	186	64	54	35	135	52
11	7169	160	70	175	122	60	43	114	53	376	89	99	45	255	67
12	3445	32	28	4	30	26	2	35	23	124	43	31	19	90	31
13	4576	68	50	27	62	44	15	70	39	191	69	59	31	149	48
14	5239	48	40	12	42	36	4	61	33	110	53	47	29	84	42

Phrases and co-occurring terms

In our preliminary experiments, we have selected certain interesting parts of speech and considered two

simple test cases, the discovery of phrases and co-occurring terms.

For phrase discovery, we selected only nouns, proper nouns, and adjectives. The search for co-occurring terms, on the other hand, was first carried through with nouns and proper nouns, and then with nouns, proper nouns, verbs, and adjectives. We also used different window sizes for episodes and produced both parallel and serial episodes which are shown in Table 2 for the exact parameters used in the experiments. The results from discovering phrases and co-occurring terms were rather similar. Both approaches produced reasonable results in fact the main difference were some increase or decrease in the term/phrase frequencies. With co-occurring terms, the same effect occurred between the results obtained with or without a gap between the sentences.

4.2 Postprocessing by pruning and weighting

After the discovery phase we applied postprocessing using the pruning schemes. In the case of episode rules, we used all the three schemes, whereas the episodes were only pruned by comparing their supports to the supports of their superepisodes. The overview of the results can be seen in Table 1 as numbers of pruned rules and episodes. Note that the numbers of episodes given contain the episodes of length one as well, whereas all the rules contain at least two words. For comparison, we have included the number of all episodes as well as the number of episodes of size one. It is interesting to see that although all these documents are legal text there are significant differences: one cannot predict the number of rules, episodes, and frequent single words from the size of the documents. Even the pruning has varying effects.

Table 2: Test Parameter Values

Discovery Of	Episode type	Selected Parts Of Speech	Distinct Symbols	Episode Support Threshold	Episode Rule Conf Threshold	Gap Between Sentences	Window Sizes
Phrases	Serial	N,PROP,A	365,2693	10	0.2	5(+1)	1.6
Co-occuring terms	Parallel	N,PROP N,PROP,V,A	293.1904 457.3109	10	—	10(+1) 1	1.11

5. Conclusion

Various algorithms have been proposed for text documents to mining frequent patterns. But how to efficiently find these patterns is still an issue in text mining domain. Traditionally, texts have been analyzed by using various information retrieval related methods, such as full-text analysis, and natural language processing. However, only few examples of data mining in text, particularly in full text, are available. In this paper we present a framework for text mining using descriptive phrase extraction. The framework follows the general knowledge discovery process, thus containing steps from preprocessing to the utilization of the results. We apply generalized episodes and episode rules data mining method. We introduce a weighting scheme that helps in pruning out redundant or non-descriptive phrases. Simulation results relived that episodes and episode rules produced discriminate between documents. Both pre and post-processing have essential roles in pruning and weighting the results.

References

[1] 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, Apr. 1996.

[2] R. Feldman, W. Kloesgen, and A. Zilberstein. Document explorer: Discovering knowledge in document collections. In Z.W. Ras and A. 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, Oct. 1997. Springer-Verlag.

[3] B. Lent, R. Agrawal, and R. Srikant. Discovering trends in text databases. In D. Heckerman, H. Mannila, D. Pregibon, and R. Uthrysamy, editors, Proceedings of the Third International Conference on Knowledge Discovery and Data Mining (KDD'97), pages 227–230, Newport Beach, California, USA, Aug. 1997. AAAI Press.

[4] H. Mannila. Data mining: machine learning, statistics, and databases. In Proceedings of the 8th International Conference on Scientific and Statistical Database Management, pages 1–6, Stockholm, Sweden, 1996.

[5] H. Mannila and H. Toivonen. Discovering generalized episodes using minimal occurrences. In E. Simoudis, J. Han, and U. Fayyad, editors, Proceedings of the Second International Conference on Knowledge Discovery and Data Mining (KDD'96), pages 146–151, Portland, Oregon, USA, Aug. 1996. AAAI Press.

[6] R. Agrawal and R. Srikant, "Fast Algorithms for Mining Association Rules in Large Databases," Proc. 20th Int'l Conf. Very Large Data Bases (VLDB '94), pp. 478-499, 1994.

[7] H. Ahonen, O. Heinonen, M. Klemettinen, and A.I. Verkamo, "Applying Data Mining Techniques for Descriptive Phrase Extraction in Digital Document Collections," Proc. IEEE Int'l Forum on Research and Technology Advances in Digital Libraries (ADL '98), pp. 2-11, 1998.

[8] S.T. Dumais, "Improving the Retrieval of Information from External Sources," Behavior Research Methods, Instruments, and Computers, vol. 23, no. 2, pp. 229-236, 1991.
[9] J. Han and K.C.-C. Chang, "Data Mining for Web Intelligence," Computer, vol. 35, no. 11, pp. 64-70, Nov. 2002.
[10] J. Han, J. Pei, and Y. Yin, "Mining Frequent Patterns without Candidate Generation," Proc. ACM SIGMOD Int'l Conf. Management of Data (SIGMOD '00), pp. 1-12, 2000.
[11] Y. Huang and S. Lin, "Mining Sequential Patterns Using Graph Search Techniques," Proc. 27th Ann. Int'l Computer Software and Applications Conf., pp. 4-9, 2003.
[12] X. Li and B. Liu, "Learning to Classify Texts Using Positive and Unlabeled Data," Proc. Int'l Joint Conf. Artificial Intelligence (IJCAI '03), pp. 587-594, 2003.
[13] Y. Li, W. Yang, and Y. Xu, "Multi-Tier Granule Mining for Representations of Multidimensional Association Rules," Proc. IEEE Sixth Int'l Conf. Data Mining (ICDM '06), pp. 953-958, 2006.
[14] Y. Li and N. Zhong, "Interpretations of Association Rules by Granular Computing," Proc. IEEE Third Int'l Conf. Data Mining (ICDM '03), pp. 593-596, 2003.
[15] Y. Li and N. Zhong, "Mining Ontology for Automatically Acquiring Web User Information Needs," IEEE Trans. Knowledge and Data Eng., vol. 18, no. 4, pp. 554-568, Apr. 2006.
[16] Y. Li, X. Zhou, P. Bruza, Y. Xu, and R.Y. Lau, "A Two-Stage Text Mining Model for Information Filtering," Proc. ACM 17th Conf. Information and Knowledge Management (CIKM '08), pp. 1023-1032,2008.
[17] A. Maedche, Ontology Learning for the Semantic Web. Kluwer Academic, 2003.
[18] C. Manning and H. Schu¨ tze, Foundations of Statistical Natural Language Processing. MIT Press, 1999.
[19] J.S. Park, M.S. Chen, and P.S. Yu, "An Effective Hash-Based Algorithm for Mining Association Rules," Proc. ACM SIGMOD Int'l Conf. Management of Data (SIGMOD '95), pp. 175-186, 1995.
[20] J. Pei, J. Han, B. Mortazavi-Asl, H. Pinto, Q. Chen, U. Dayal, and M. Hsu, "Prefixspan: Mining Sequential Patterns Efficiently by Prefix-Projected Pattern Growth," Proc. 17th Int'l Conf. Data Eng. (ICDE '01), pp. 215-224, 2001.
[21] F. Sebastiani, "Machine Learning in Automated Text Categorization," ACM Computing Surveys, vol. 34, no. 1, pp. 1-47, 2002.
[22] M. Seno and G. Karypis, "Slpminer: An Algorithm for Finding Frequent Sequential Patterns Using Length-Decreasing Support Constraint," Proc. IEEE Second Int'l Conf. Data Mining (ICDM '02), pp. 418-425, 2002.
[23] R. Srikant and R. Agrawal, "Mining Generalized Association Rules," Proc. 21th Int'l Conf. Very Large Data Bases (VLDB '95), pp. 407-419, 1995.
[24] S.-T. Wu, Y. Li, Y. Xu, B. Pham, and P. Chen, "Automatic Pattern- Taxonomy Extraction for Web Mining," Proc. IEEE/WIC/ACM Int'l Conf. Web Intelligence (WI '04), pp. 242-248, 2004.
[25] Y. Xu and Y. Li, "Generating Concise Association Rules," Proc. ACM 16th Conf. Information and Knowledge Management (CIKM '07), pp. 781-790, 2007.
[26] X. Yan, J. Han, and R. Afshar, "Clospan: Mining Closed Sequential Patterns in Large Datasets," Proc. SIAM Int'l Conf. Data Mining (SDM '03), pp. 166-177, 2003.
[27] Ning Zhong, Yuefeng Li, and Sheng-Tang Wu, "Effective Pattern Discovery for Text Mining", IEEE Transactions on Knowledge And Data Engineering, VOL. 24, NO. 1, JANUARY 2012
[28] Helena Ahonen Oskari Heinonen Mika Klemettinen," Applying Data Mining Techniques for Descriptive Phrase Extraction in Digital Document Collections" ,IEEE,2008.