ENHANCED INNER PATTERN EVOLVING APPROACH FOR DISCOVERED PATTERNS

A PROJECT REPORT

Submitted by

SREE DEVI C

in partial fulfillment for the requirement of award of the degree of

MASTER OF ENGINEERING

in

COMPUTER SCIENCE AND ENGINEERING
Department of Computer Science and Engineering
KUMARAGURU COLLEGE OF TECHNOLOGY,
COIMBATORE 641 049
(An Autonomous Institution Affiliated to Anna University, Chennai)

APRIL 2013

ENHANCED INNER PATTERN EVOLVING APPROACH FOR DISCOVERED PATTERNS

A PROJECT REPORT

Submitted by

SREE DEVI C

in partial fulfillment for the requirement of award of the degree of

MASTER OF ENGINEERING

FACULTY OF COMPUTER SCIENCE AND ENGINEERING
KUMARAGURU COLLEGE OF TECHNOLOGY,
COIMBATORE 641 049
(An Autonomous Institution Affiliated to Anna University, Chennai)

APRIL 2013

BONAFIDE CERTIFICATE

Certified that this project work titled **" ENHANCED INNER PATTERN EVOLVING APPROACH FOR DISCOVERED PATTERNS"** is the bonafide work of Mrs. SREE DEVI C,who carried out the research under my supervision. Certified further, that to the best of my knowledge the work reported herein does not form part of any other thesis or dissertation on the basis of which a degree or award was conferred on an earlier occasion on this or any other students.

N.JAYAPATHI M.Tech.,	Dr.D.CHANDRAKALA M.E.,Ph.D.,
HEAD OF THE DEPARTMENT	**SUPERVISOR**
Dept of Computer Science and Engineering	**Associate Professor**
Kumaraguru College of Technology	Dept of Computer Science and Engineering
Coimbatore-641609	Kumaraguru College of Technology
	Coimbatore- 641 04.

Submitted for the Project Viva-Voce examination held on _____.

---------- ----------

Internal Examiner **External Examiner**

\

ABSTRACT

Text mining is the process of deriving high quality information from text. High quality information is typically derived through the devising of pattern updating. The large number of patterns are generated by using data mining approaches, but the effective use and update of these patterns is still an open research issue. Most existing text mining methods adopted term-based approaches suffer from the problems of polysemy and synonymy. Polysemy is a word has multiple meanings, and synonymy is multiple words having the same meaning.

The proposed approach focus on the development of a knowledge discovery model to effectively use and update the discovered patterns and apply it to the field of text mining. This method introduces innovative and effective pattern discovery technique which includes the processes of pattern deploying and pattern evolving. This method improve the effectiveness of using and updating discovered patterns for finding relevant and interesting information. To update pattern effectively threshold value is assigned in evolution process to reduce noisy pattern and to achieve high accuracy.

ஆய்வுச்சுருக்கம்

உரை சுரங்க முறை என்பது உரையிலிருந்து உயர்தர தகவல் பெறும் முறை ஆகும்.உயாதர தகவலானது புள்ளியியல அமைப்பு கற்றல் மற்றும் போக்குகள் கண்டுபிடிக்கும் துறையின் மூலம் பெறப்படுகிறது.தகவல் சுரங்க முறையில் உருவாக்கப்படும் அதிகப்படியான போக்குகளை திறம்பட நிர்வகிப்பது மற்றும் மாற்றுவது என்பது ஆய்வு நிலையிலே உள்ளது.பெரும்பாலான ஏற்கனவே உள்ள உரை சுரங்க அணுகு முறைகள் ஒரு வார்த்தை பல அர்த்தம் மற்றும் ஒரே பொருளுடையதாக இருக்கும் தன்மைகளால் பாதிக்கப்படுகிறது.முன் மொழியபபட்ட அணுகுமுறையானது ஒரு அறிவு கண்டுபிடிப்பு மாதிரியை பயன்படுத்தி கண்டு பிடிக்கப்பட்ட போக்குகளை திறம்பட கையாள உரை சுரங்க முறையை கையாளுகிறது.இந்த அணுகுமுறையானது கண்டுபிடிகப்பட்ட போக்குகள் தேவையான மற்றும் சுவாரஸ்யமான தகவல்களை பெறுவதற்காக புதுப்பிக்கப்படுகிறது.அவ்வாறு போக்குகளை திறம்பட புதுப்பிக்க ஒரு மதிப்பானது நிர்னயிக்கப்படுகிறது. துல்லியத்தை அதிக படுத்துவதன் மூலம் செயல்திறன் மதிப்பு மேம்படுத்தப்படுகிறது.

ACKNOWLEDGEMENT

First and foremost, I would like to thank the Lord Almighty for enabling me to complete this project. I express my profound gratitude to **Padmabhusan Arutselvar Dr.N.Mahalingam, B.Sc., F.I.E, Chairman**, **Dr.B.K. Krishnaraj Vanavarayar**, **B.com., BL, Co-Chairman, Mr. M. Balasubramaniam, M.Com., M.B.A, Correspondent, Mr.Sankar Vanavarayar**, **M.B.A., PGDIEM, Joint Correspondent** and **Dr.S.Ramachandran Ph.D., Principal** for providing the necessary facilities to complete my project.

I take this opportunity to thank **Prof.N.Jayapathi M.Tech**., Head of the Department, Department of Computer Science and Engineering, for his support and motivation. Special thanks to my Project Coordinator **Dr.V.Vanitha M.E., Ph.D.**, Senior Associate Professor, Department of Computer Science and Engineering, for arranging brain storming project review sessions.

I register my sincere thanks to my Guide **Dr.D.Chandrakala M.E., Ph.D.,** Associate Professor, Department of Computer Science and Engineering. I am grateful for her support, encouragement and ideas. I would like to convey my honest thanks to all **Teaching** and **Non Teaching Staff** members of the department and my classmates for their support.

I dedicate this project work to my **Parents** for no reasons but feeling from bottom of my heart that without their love, this work would not be possible.

- SREE DEVI C

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

KDD	Knowledge Discovery Process
SVM	Support Vector Machine
FP	Frequent Pattern
TC	Text Categorization
COG	Conceptual Ontological Graph
PTM	Pattern Taxonomy Model
PDM	Pattern Deploying Method
IPE	Inner Pattern Evolving

CHAPTER 1

INTRODUCTION

1.1 TEXT MINING

Text mining is a growing new field that attempts to collect meaningful information from natural language text. It may be loosely characterized as the process of analyzing text to extract information that is useful for particular purposes. Compared with the kind of data stored in databases, text is unstructured, amorphous, and difficult to deal with algorithmically. Nevertheless, in modern culture, text is the most common vehicle for the formal exchange of information

The field of text mining usually deals with texts whose function is the communication of factual information or opinions, and the motivation for trying to extract information from that text automatically .

1.1.1Data Mining:

- A step in the knowledge discovery process consisting of particular algorithms (methods), produces a particular enumeration of patterns (models) over the data which is shown in the Fig.1.1

Knowledge Discovery Process: The process of using data mining methods (algorithms) to extract (identify) what is knowledge according to the specifications of measures and thresholds, using a database along with any necessary preprocessing or transformations.

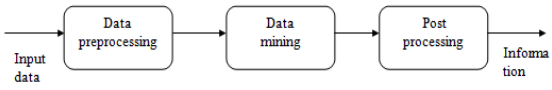


Fig.1.1 KDD Process

Data in Data Mining:

- Relational Databases
- Data Warehouses
- Transactional Databases
- Advanced Database Systems
  - Object-Relational
  - Multimedia
  - Text
  - Heterogeneous and Distributed

1.1.2 Text mining and data mining

Just as data mining can be loosely described as looking for patterns in data, text mining is about looking for patterns in text. However, the superficial similarity between the two conceals real differences. Data mining can be more fully characterized as the

extraction of implicit, previously unknown, and potentially useful information from data The information is implicit in the input data it is hidden, unknown, and could hardly be extracted without recourse to automatic techniques of data mining.

1.1.3 Mining plain text

This section describes the major ways in which text is mined when the input is plain natural language, rather than partially-structured Web documents. The problems that involve in extracting information for human consumption text summarization and document retrieval are discussed in this chapter and the task of assessing document similarity either to categorize documents into predefined classes or to cluster them in“natural” ways are also discussed.

1.1.4 Extracting information for human consumption

In some situations the information mined from text is expressed in a form that is intended for consumption by people rather than computers. The result is not “actionable” in the sense discussed above, and therefore lies on the boundary of what is normally meant by “text mining”.

1.1.5 Text summarization

A text summarizer strives to produce a condensed representation of its input, intended for human consumption. It may condense individual documents or groups of documents. The output of text compression algorithms is certainly not human readable form. Summarization differs from many other forms of text mining in that there are

people, namely professional abstractors, who are skilled in the art of producing summaries and carry out the task as part of their professional life.

1.1.6 Document retrieval

Given a corpus of documents and a user’s information need expressed as some sort of query, document retrieval is the task of identifying and returning the most relevant documents. Traditional libraries provide catalogues (whether physical card catalogues or computerized information systems) that allow users to identify documents based on surrogates consisting of metadata salient features of the document such as author, title, subject classification, subject headings, keywords.

1.1.7 Information retrieval

Information retrieval might be regarded as an extension to document retrieval where the documents that are returned are processed to condense or extract the particular information sought by the user. Thus document retrieval could be followed by a text summarization stage that focuses on the query posed by the user, or an information extraction stage using some techniques. The granularity of documents may be adjusted so that each individual subsection or paragraph comprises a unit in its own right, in an attempt to focus results on individual nuggets of information rather than lengthy documents.

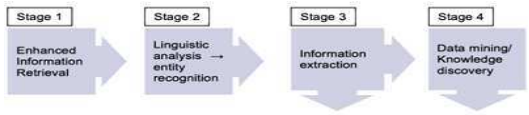


Fig.1.2Components of Text Mining

Text mining, drawing on techniques from information retrieval, natural language processing, information extraction and data mining/knowledge discovery are shown in the Fig.1.2.

1.2 COSTS, BENEFITS, BARRIERS AND RISKS ASSOCIATED WITH TEXT MINING

- Costs include access, transaction, entry, staff and infrastructure costs.
- Benefits include: efficiency; unlocking hidden information and developing new knowledge; exploring new horizons; improved research and evidence base; and improving the research process and quality.
- Broader economic and societal benefits were also highlighted, such as cost savings and productivity gains, innovative new service development, new business models and new medical treatments.
- Consulters in general felt that there were significant barriers to uptake of text mining. These include: legal uncertainty, orphaned works and attribution requirements; entry costs. 'noise' in results; document formats.

LITERATURE SURVEY

1.3.1Learning to Classify Texts Using Positive And Unlabelled data

Shehata (2007) proposed traditional text classification, a classifier is built using labeled training documents of every class. Given a set of documents P of a particular class (called positive class) and a set U of unlabeled documents that contains documents from class P and also other types of documents (called negative class documents), there is need to build a classifier to classify the documents in U into documents from P and documents not from P.

The key feature of this problem is that there is no labeled negative document. It combines the Rocchio method and the SVM technique for classifier building.

1.1.1.1methods:

Finding reliable negative documents METHOD 1: Rocchio

This method treats the entire unlabeled set U as negative documents and then uses the positive set *P* and U as the training data to build a Rocchio classifier.

- The classifier is then used to classify U.
- Those documents that are classified as negative are considered (reliable) negative data, denoted by RN.

The reason that Rocchio works well is as follow:

In the positive class based learning, the unlabeled set U typically has the following characteristics:

1. The proportion of positive documents in U is very small. Thus, they do not affect very much.

2. The negative documents in U are of diverse topics. Thus, in the vector space, they cover a very large region

The problem with this method is it does not perform well because it tends to classify too many negative documents as positive documents (low precision for positive). SVM using RN and P as training data can correct the situation and produce a more accurate final classifier.

1.3.1.2 Rocchio with Clustering

This new method can further purify RN without removing too many negative documents from RN.

1.3.1.3 Classifier building

This step builds the final classifier by running SVM iteratively with the document sets P and RN or RN, depending on the method used in step 1. Let Q be the set of the remaining unlabeled documents, Q = U.RN (or U RN.)

Advantage

1. Uses best classifier method.
2. Both Roc-SVM and Roc-Clu-SVM are robust with different numbers of positive documents.

Disadvantage

1. Its strategy of extracting the initial set of strong negative documents could easily go wrong without sufficient\ positive data.
2. Even when the number of positive documents is large, it may go wrong

1.1.2 Feature Selection and Feature Extraction for Text Categorization

Sebastiani (2002) proposed good categorization performance is achieved using a statistical classifier and a proportional assignment strategy. The optimal feature set size for word-based indexing is found to be surprisingly low (10 to 15 features) despite the large training sets. Syntactic indexing phrases, clusters of these phrases, and clusters of words are all found to provide less effective representations than individual word.

1.3.2.1 Categorization Method

1. Binary Categorization

In order to compare text categorization output with an existing manual categorization probability estimates with explicit binary category assignments should be replaced.

Previous work on statistical text categorization has often ignored this step, or has not dealt with the case where documents can have zero, one, or multiple correct categories.

- Feature Selection WORDS-DF2: Starts with all words tokenized by parts. Capitalization and syntactic class ignored. Stop words discarded based on syntactic tags. Tokens consisting solely of digits and punctuation removed.

- WC-MUTINFO-135: Starts with WORDS-DF2, and discards words occurring in fewer than 5 or more than 1029 (7%) training documents. RNN clustering used 135 meta features with value equal to mutual information between presence of the word and presence of a manual indexing category.
- PHRASE-DF2: Starts with all simple noun phrases bracketed by parts. Stop words removed from phrases based on tags. Single word phrases discarded. Numbers replaced with the token NUMBER. Phrases occurring in fewer than 2 training documents removed.
- PC-W-GIVEN-C-44: Starts with PHRASE-DF2. Phrases occurring in fewer than 5 training documents removed. RNN clustering uses 44 meta features with value equal to our estimate of P(W =1IC = 1) for phrase W and category C.

1.3.2.2 Feature extraction

The definition of meta features is a key issue to reconsider. Phrases have low frequency, this approach use meta features corresponding to bodies of text large enough that could expect co occurrences of phrases within them. This simply gives many opportunities for accidental co occurrences to arise, without providing a sufficient constraint on the relationship between phrases (or words). The fact that clusters captured few high quality semantic relationships, even when an extremely conservative clustering method was used, suggests that using other clustering methods with the same meta feature definitions is not likely to be effective.

Advantage

Statistical classifier trained on manually categorized documents to achieve quite effective performance in assigning multiple, overlapping categories to documents.

Disadvantage

1. Clustering of words and phrases based on syntactic context is a promising approach .
2. Pruning out of low quality phrases is not considered
3. Practical systems, combinations of knowledge-based and statistical approaches are likely to be the best strategy

1.1.3 Mining Frequent Patterns without Candidate Generation

J. Han (2000) proposed mining frequent patterns in transaction databases, time-series databases, and many other kinds of databases has adopted an Apriori-like candidate set generation-and-test approach. Apriori like algorithm is costly and tedious to repeatedly scan the database and check a large set of candidates by pattern matching. The proposed structure is a novel frequent pattern tree (FP-tree) structure, which is an extended prefix-tree structure for storing compressed, crucial information about frequent patterns.

A compact data structure for Frequent Pattern Tree is constructed in the following steps:

1. Scan the transaction database DB once and collect the set of frequent items *F* and their supports.
2. Sort *F* in support descending order as *L*, the list of frequent items.
3. Create the root of an FP-tree, *T*, which is "null".
4. For each transaction Trans in DB do the following:

   Select and sort the frequent items in Trans according to the order of L.

If T has child N such that N. item-name = p. item-name, then increment N's count by 1 (where p-first element); else create a new node N, and let its count be 1, its parent link be linked to T.

The input to the FP-tree construction is a transaction DB and a minimum support threshold ξ and the output is its frequent pattern tree, FP-tree.

FP-growth scales much better than Apriori. This is because as the support threshold decreases the number as well as the length of frequent item sets increase dramatically.

The advantages of FP-growth over other approaches are

1. A large database is compressed into a highly condensed, smaller data structure, which avoids costly, repeated database scans.
2. FP-tree-based mining adopts a pattern fragment growth to avoid the costly generation of a large number of candidate sets.

4.1.1 Machine Learning in Automated Text Categorization

Sebastiani (2002) proposed the most effective approach to the problem seemed to be that of manually building automatic classifiers by means of knowledge engineering techniques, i.e. manually defining a set of rules encoding expert knowledge on how to classify documents under a given set of categories. In the '90s, with the booming production and availability of on-line documents, automated text categorization has witnessed an increased and renewed interest, prompted by which the machine learning paradigm to automatic classifier construction has emerged and definitely superseded the knowledge-engineering approach.

Within the machine learning paradigm, a general inductive process (called the learner) automatically builds a classifier (also called the rule, or the hypothesis) by "learning", from a set of previously classified documents, the characteristics of one or more categories. Main approaches that have been taken towards automatic text

categorization within the general machine learning paradigm. Issues pertaining to document indexing, classifier construction, and classifier evaluation were discussed.

- The automatic assignment of documents to a predefined set of categories, which is the main topic of this paper, the term has also been used to mean.
- The automatic definition of such a set of categories (nowadays universally referred to as clustering), or
- The automatic assignment of documents to a set of categories which is not pre defined(a task nowadays universally referred to as (free text) indexing).

Automated TC has now established itself as one of the major areas in the information: systems discipline, because of a number of reasons:

- Its domains of application are numerous and important, and given the proliferation of documents in digital form they are bound to increase dramatically in both number and importance.
- It is indispensable in many applications in which the sheer number of the documents to be classified and the short response time imposed by the application make the manual alternative implausible.
- It can dramatically improve the productivity of human classifiers in those applications in which no classification decision can be taken without a final human expert judgment by providing tools that quickly "suggest" plausible decisions.
- It has reached effectiveness levels comparable to those obtained in manual TC with the involvement of trained professionals. The effectiveness of manual TC is not 100% anyway and, perhaps more importantly, it is not going to be improved by the progress of research. On the contrary, the levels of effectiveness of automated TC are growing at a steady pace, and even if it is plausible to hypothesis that they will eventually reach a plateau well below the 100% level, this plateau will probably be higher that the effectiveness levels of manual TC.

Disadvantage

Concerning other more radically different media, the current situation is not as bright. Current research on automatic document categorization under thematic categories mostly focuses on the text medium interesting attempt at image categorization based on a textual metaphor. The reason for this is that capturing real semantic content in the automatic indexing of non-textual media is still an open research problem. While there are systems that attempt to detect content e.g. in images by recognizing shapes, color distributions and texture, the general problem of image semantics is still unsolved.

The main reason for this fact is that natural language, the language of the text medium, admits far fewer variations than the "languages" employed by the other media. For instance, while the concept of a house can be "triggered" by relatively few natural language expressions such as house, houses, home, housing, inhabiting, etc., it can be triggered by far more images: the images of all the different houses that exist, of all possible colors and shapes, viewed from all the possible perspectives, from all the possible distances

1.3.5 A Concept-based Model for Enhancing Text Categorization

Shehata (2007) proposed a new concept-based model that analyzes terms on the sentence and document levels rather than the traditional analysis of document only is introduced. The concept-based model can effectively discriminate between non-important terms with respect to sentence semantics and terms which hold the concepts that represent the sentence meaning. The proposed model consists of concept-based statistical analyzer, conceptual ontological graph representation, and concept extractor

The term which contributes to the sentence semantics is assigned two different weights by the concept-based statistical analyzer and the conceptual onto-logical graph representation. These two weights are combined into a new weight. The concepts that have maximum combined weights are selected by the concept extractor.

Each sentence is labeled by a semantic role labeler that determines the terms which contribute to the sentence semantics associated with their semantic roles in a sentence. Each term that has a semantic role in the sentence, is called concept. Concepts can be either a word or phrase and it is totally dependent on the semantic structure of the sentence.

The concept-based model analyzes each term within a sentence and a document using the following three components. The first component is the concept-based statistical analyzer that analyzes each term on the sentence and the document levels. After each sentence is labeled by a semantic role labeler, each term is statistically weighted based on its contribution to the meaning of the sentence. This weight discriminates between non-important and important terms with respect to the sentence semantics.

The second component is the Conceptual Ontological Graph (COG) representation which is based on the conceptual graph theory and utilizes graph properties. The COG representation captures the semantic structure of each term within a sentence and a document, rather than the term frequency within a document only. After each sentence is labeled by a semantic role labeler, all the labeled terms are placed in the COG representation according to their contribution to the meaning of the sentence. Some terms could provide shallow concepts about the meaning of a sentence, but, other terms could provide key concepts that hold the actual meaning of a sentence.

Each concept in the COG representation is weighted based on its position in the representation. Thus, the COG representation is used to provide a definite separation among concepts that contribute to the meaning of a sentence. Therefore, the COG representation presents concepts into a hierarchical manner. The key concepts are captured and weighted based on their positions in the COG representation. At this point, concepts are assigned two different weights using two different techniques which are the concept-based statistical analyzer and the COG representation. It is important to note that both of them achieve the same functionality, in different ways.

The third component, which is the concept extractor, combines the two different weights computed by the concept-based statistical analyzer and the COG representation to denote the important concepts with respect to the two techniques. The extracted top concepts are used to build standard normalized feature vectors using the standard vector space model (VSM) for the purpose of text categorization.

Weighting based on the matching of concepts in each document, is showed to have a more significant effect on the quality of the text categorization due to the similarity's insensitivity to noisy terms that can lead to an incorrect similarity measure. The concepts are less sensitive to noise when it comes to calculating normalized feature vectors. The explanations of the important terms, which are used in this approach, are listed as follows:

- Verb-argument structure: (e.g John hits the ball). "hits" is the verb. "John" and "the ball" are the arguments of the verb "hits
- Label : A label is assigned to an argument. e.g: "John" has subject (or Agent) label. "the ball" has object (or theme) label
- Term: is either an argument or a verb. Term is also either a word or a phrase (which is a sequence of words.

Disadvantage

1. No experimental work to web document categorization.
2. Investigatation for the usage of such models on other corpora and its effect on document categorization results, compared to that of traditional methods is not considered.

1.3.6 PrefixSpan: Mining Sequential Patterns Efficiently by Prefix-Projected Pattern growth

H. Pinto, Q. Chen et al (2001) proposed sequential pattern mining is an important data mining problem with broad applications. It is challenging since one may need to examine a combinatorial explosive number of possible subsequence patterns. A novel sequential pattern mining method, called Prefix Span (i.e., **Prefix**-projected **S**equential **pa**ttern mining). Its general idea is to examine only the prefix subsequences and project only their corresponding postfix subsequences into projected databases. In each projected database, sequential patterns are grown by exploring only local frequent patterns.

To further improve mining efficiency, two kinds of database projections are explored: level-by-level projection and bi-level projection. Moreover, a main-memory-based pseudo-projection technique is developed for saving the cost of projection and speeding up processing when the projected (sub)-database and its associated pseudo-projection processing structure can fit in main memory. The performance study shows that bi-level projection has better performance when the database is large, and pseudo projection speeds up the processing substantially when the projected databases can fit in memory.

Advantage

It overcome the following problems
1. Potentially huge set of candidate sequences:
2. Multiple scans of databases.

Disadvantage

1. It fail to focus on patterns with time constraints, time windows and/or taxonomy, and other kinds of time-related knowledge.
2. No pattern growth-based sequential pattern mining methodology for effectively mining DNA databases.

1.3.7 Automatic Pattern-Taxonomy Extraction for Web Mining

Xu (2004) proposed model for discovering frequent sequential patterns, phrases, which can be used as profile descriptors of documents. It is unquestionable to obtain numerous phrases using data mining algorithms. However, it is difficult to use these phrases effectively for answering what users want. Therefore, this pattern taxonomy extraction model which performs the task of extracting descriptive frequent sequential patterns by pruning the meaningless ones. The model then is extended and tested by applying it to the information filtering system.

1.3.7.1 Pattern taxonomy model

A new pattern-based model PTM (Pattern Taxonomy Model) for the representation of text documents. Pattern taxonomy is a tree-like structure that illustrates

the relationship between patterns extracted from a text collection. Fig.1.3 illustrates an example of the pattern taxonomy for the frequent patterns, where the nodes represent frequent patterns and their covering sets; non closed patterns can be pruned; the edges are "is-a" relation.

After pruning, some direct "is-a" retaliations may be changed, for example, pattern t6 would become a direct sub pattern of {t3; t4; t6} after pruning non closed patterns. Smaller patterns in the taxonomy, for example pattern {t6} are usually more general because they could be used frequently in both positive and negative documents; and larger patterns, for example pattern {t3; t4; t6}, in the taxonomy are usually more specific since they may be used only in positive document.

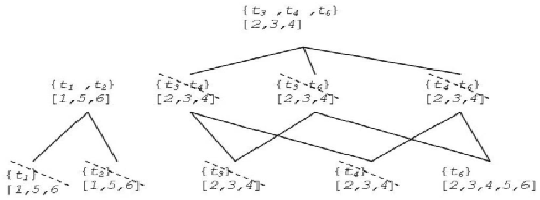


Fig.1.3.Pattern taxonomy

If the relative supports of the (n+1)Terms patterns are greater than or equal to min_sup, patterns is stored (i.e., frequent (n+1)Terms patterns). Repeat SP Mining recursively if there exists at least one frequent (n+1)Terms pattern and pass them as the

value of the first parameter; otherwise, the program returns. As a result, the output of the algorithm is the set of the frequent maximum sequential patterns. By pruning meaningless patterns, which have been proven to be the source of the 'noise' the problem of 'over fitting' is solved.

Disadvantage

In order to improve the accuracy, one major direction is to extract and use the interesting information from negative or unlabeled documents.

1.3.8 Deploying Approaches for Pattern Refinement in Text Mining

Propose two novel pattern deploying algorithms to effectively exploit discovered patterns for the text mining problem.

1.3.8.1 Deploying Method

Li (2006) propose two methods to deploy discovered patterns in this study. The first one is pattern deploying method (PDM) and the second one is pattern deploying with relevance function (PDR).

Pattern Deploying Method (PDM):

In order to deploy discovered patterns and acquire deployed support of each term in these patterns, pattern composition operation is use to join two patterns. Let termset(P) = {t|(t, f) 2 P} be the termset of pattern P, where f denotes term frequency in pattern P.

Pattern Deploying with Relevance Function (PDR)

PDR is a deploying method with the use of relevance function. This approach utilizes a probabilistic method to estimate the term weight. The main difference between PDM and PDR is that in the latter method, the support of pattern is involved in the estimation of term weights but not in the former method.

Advantage

Effective usage of discovered patterns

Disadvantage

1. It fail to use the information from the non-relevant documents to improve the system in term of effectiveness
2. An adaptive algorithm of learning the change of the characteristics of context is also needed

1.3.9. Effective Pattern Discovery for Text Mining

Ning Zhong (2012) proposed that there are two fundamental issues regarding the effectiveness of pattern-based approaches: low frequency and misinterpretation. Given a specified topic, a highly frequent pattern (normally a short pattern with large support) is usually a general pattern, or a specific pattern of low frequency.

If the minimum support is decreased, a lot of noisy patterns would be discovered. Misinterpretation means the measures used in pattern mining (e.g., "support" and "confidence") turn out to be not suitable in using discovered patterns to answer what users want.

The difficult problem hence is how to use discovered patterns to accurately evaluate the weights of useful features (knowledge) in text documents. Evaluates term weights according to the distribution of terms in the discovered patterns rather than the distribution in documents for solving the misinterpretation problem. It also considers the influence of patterns from the negative training examples to find ambiguous (noisy) patterns and try to reduce their influence for the low-frequency problem referred as pattern evolution.

In order to solve the above paradox, this method presents an effective pattern discovery technique, which first calculates discovered specificities of patterns and then evaluates term weights according to the distribution of terms in the discovered patterns rather than the distribution in documents for solving the misinterpretation problem. It also considers the influence of patterns from the negative training examples to find ambiguous (noisy) patterns and try to reduce their influence for the low-frequency problem.

1.3.9.1 Inner Pattern Evolution

In this how to reshuffle supports of terms within normal forms of d-patterns based on negative documents in the training set. The technique will be useful to reduce the side effects of noisy patterns because of the low-frequency problem. This technique is called inner pattern evolution here, because it only changes a pattern's term supports within the pattern. A threshold is usually used to classify documents into relevant or irrelevant categories.

A noise negative document nd in D is a negative document that the system falsely identified as a positive, that is weight(nd)> Threshold(dp). In order to reduce the noise, this approach track which d-patterns have been used to give rise to such an error. The offering is part of the sum of supports of terms in a d-pattern where these terms also appear in a noise document.

There are two types of offenders: 1) a complete conflict offender which is a subset of nd; and 2) a partial conflict offender which contains part of terms of nd. The basic idea of updating patterns is explained as follows: complete conflict offenders are removed from d-patterns first. For partial conflict offenders, their term supports are reshuffled in order to reduce the effects of noise documents. IP Evolving is used to estimate the threshold for finding the noise negative documents.

1.3.9.2IP shuffling:

The task of algorithm Shuffling is to tune the support distribution of terms within a d-pattern. A different strategy is dedicated in this algorithm for each type of offender. As stated in the algorithm Shuffling, complete conflict offenders (d-patterns) are removed

since all elements within the d-patterns are held by the negative documents indicating that they can be discarded for preventing interference from these possible "noises"

The proposed model includes two phases: the training phase and the testing phase. In the training phase, the proposed model first calls algorithm PTM to find d-patterns in positive documents based on a min sup, and evaluates term supports by deploying d patterns to terms. It also calls algorithm IP Evolving to revise term supports using noise negative documents in D based on an experimental co efficient. In the testing phase, it evaluates weights for all incoming document. The incoming documents then can be sorted based on these weights.

CHAPTER 2

IMPLEMENTATION

2.1 EXISTING SYSTEM

Term based methods suffer from the problems of polysemy and synonymy, where polysemy means a word has multiple meanings, and synonymy is multiple words having the same meaning. The semantic meaning of many discovered terms is uncertain for answering what users want. Phrase based approaches could perform better than the term based ones, as phrases may carry more "semantics" like information. Although phrases are less ambiguous and more discriminative than individual terms, the likely reasons for the discouraging performance include: 1) phrases have inferior statistical properties to terms, 2) they have low frequency of occurrence, and 3) there are large numbers of redundant and noisy phrases among them.

There are two fundamental issues regarding the effectiveness of pattern-based approaches:low frequency and misinterpretation. Given a specified topic, a highly frequent pattern (normally a short pattern with large support) is usually a general pattern, or a specific pattern of low frequency. If the minimum support is decreased, a lot of noisy patterns would be discovered. Misinterpretation means the measures used in pattern mining (e.g., "support" and "confidence") turn out to be not suitable in using discovered patterns to answer what users want. The difficult problem hence is how to use discovered

patterns to accurately evaluate the weights of useful features (knowledge) in text documents. In recent work of effective pattern generation the term distribution calculation is not effective still it have misinterpretation and low frequency problem.

DRAWBACK

- In a large number of patterns generated how to effectively use and
- Update patterns is still an open research issue.
- Polysemy and synonymy
- Phrases have inferior statistical properties to terms
- They have low frequency of occurrence
- There are large numbers of redundant and noisy phrases among them
- Low frequency and misinterpretation.

2.2 PROBLEM DEFINITION

Text mining is the process of deriving high-quality information from text. High-quality information is typically derived through the devising of patterns and trends through means such as statistical pattern learning. With a large number of patterns generated by using data mining approaches, how to effectively use and update these patterns is still an open research issue. Most existing text mining methods adopted term-based approaches they all suffer from the problems of polysemy and synonymy. Polysemy means a word has multiple meanings, and synonymy is multiple words having the same meaning.

This approach provide an innovative and effective pattern discovery technique which includes the processes of pattern deploying and pattern evolving, to improve the effectiveness of using and updating discovered patterns for finding relevant and

interesting information. To update pattern effectively it uses effective threshold value in evolution process to reduce noisy pattern and also tune performance value for accuracy. In recent work of effective pattern generation the term distribution calculation is not effective still it have misinterpretation and low frequency problem.

2.3 PROPOSED SYSTEM

Pattern deploying and pattern progressing, calculate discovered specificities of patterns and then evaluates term weights according to the distribution of terms in the discovered patterns. It also considers the influence of patterns from the negative training examples. The process of updating ambiguous patterns can be referred as pattern progressing. The proposed methodology can improve the accuracy of patterns generated .

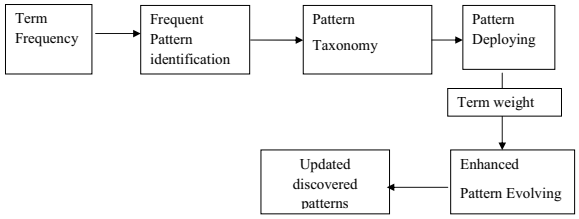


Fig.2.1 Overview of Proposed Work

The overview of the proposed work is shown in the Fig.2.1 in which term frequency is calculated which is given as input to next module to detect the frequent term. Next pattern taxonomy module involves the document splitting. In Pattern Deploying the weight is calculated and average value is given as threshold value to the next module based on that the updation of pattern generation is carried.

2.4 MODULE DESCRIPTION

2.4.1.Frequent and Closed Patterns

Frequent patterns is probably one of the most important concepts in data mining. Given a termset X in document d, to denote the covering set of X for d, which includes all paragraphs. Its absolute support is the number of occurrences of termset in all paragraphs. Its relative support is the fraction of the paragraphs that contain the pattern. For example a set of paragraphs for a given document PS(D)={dp1dp2..,dp3}, and duplicate terms were removed. Ten frequent patterns using the definition min_sup=50% is obtained .In order to provide repeated expressions that are exact repetitions as well as repetitions with slight variations. X2 and X3 are also frequent itemsets but they seem redundant because X1 carries the same information in one single itemset.

2.4.2.2Pattern Taxonomy

Patterns can be structured into a taxonomy by using the is-a (or subset) relation. For instance,in the Fig.2.2 for a 2 Terms pattern <t2,t3>,if there exist a 3t erm frequent pattern <t1,t2,t3> having the same frequency as pattern <t1,t2>,then the shorter one is treated as non closed pattern and therefore pruned which is shown in the Fig.2.2.After pruning, some direct “is-a” retaliations may be changed, for example, pattern <t1>g would become a direct subpattern of <t1,t2,t3> . Smaller patterns in the taxonomy, for

example pattern {t1}, are usually more general because they could be used frequently in both positive and negative documents

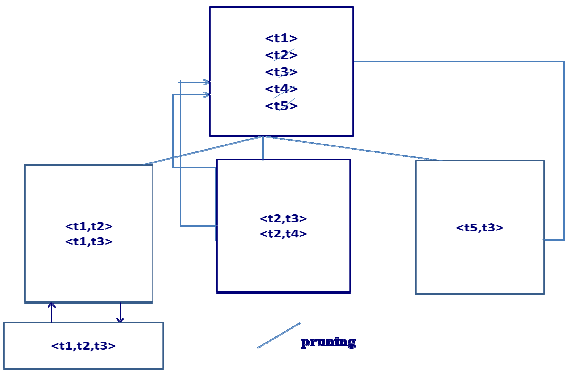


Fig. 2.2 Pruning of meaningless terms

5.2.3 Closed Sequential Patterns

The only one downside of a maximal frequent item set is that, actual support of those sub-item sets is not known. It is important to find association rules within the item

sets. Keep that in mind for now and let's think of how to get all the frequent item sets that have the same support with all their subsets. An item set is closed if none of its immediate supersets has the same support as the item set..A sequence A is a subsequence of another sequence B , denoted by A,B. A is a sub pattern of B, and B is a super pattern of A which is shown in the Fi.2.3.

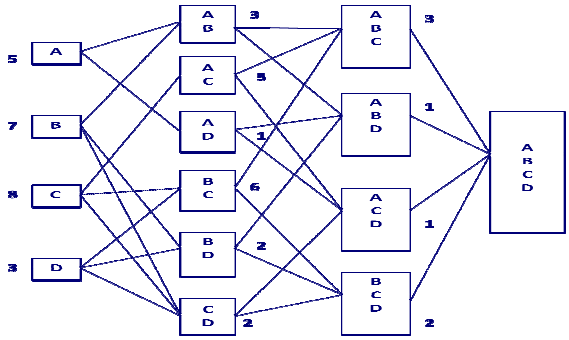


Fig.2.3 Representation of Closed, Maximal Frequent Itemsets

A sequential pattern X is called frequent pattern if its relative support greater than or equal to a minimum support. The property of closed patterns can be used to define closed

sequential patterns. A frequent sequential pattern X is called closed if not any super pattern A of (A,B) such that sup(A) = sup(A,B).

5.2.4 Pattern deploying method

To use the semantic information in the pattern taxonomy to improve the performance of closed patterns in text mining there is need to interpret discovered patterns by summarizing them as d-patterns in order to accurately evaluate term weights. The rationale behind this motivation is that d-patterns include more semantic meaning than terms that are selected based on a term-based technique.

D-Pattern Mining Algorithm is to improve the efficiency of the pattern taxonomy mining, an algorithm, SP Mining, was proposed to find all closed sequential patterns, which used the well-known Apriori property in order to reduce the searching space. A method is needed to deploy sequential patterns into a feature space. In this module the term weight for every document is calculated using the following parameters TF=Term Frequency, k= 1.5,b= 0.5,DL= Document Length ,AVDL= Average Document Length .

5.2.5 Inward Pattern Evolution

The technique will be useful to reduce the side effects of noisy patterns because of the low-frequency problem. In the enhanced approach the pattern generation the k value and m value is used in the algorithm to generate the relevant pattern based on the term weight since in the case of inner pattern evolving approach term weight is calculated only for the document where it is exactly present but in case of enhanced approach the term

weight is average term weight thus large number of pattern is generated with high accuracy indicates more information are provided.

CHAPTER 3

RESULTS AND ANALYSIS

3.1 EXPERIMENTATION

The proposed work is implemented using .Net Framework. The form for term frequency, term weight are created using VB.Net. The language used for the calculation is c#. The patterns generated are stored in SQL server database.

3.2 SYSTEM SPECIFICATION

3.2.1 Hardware requirements

Processor	: Pentium IV
Speed	:Above 500 MHz
RAM capacity	: 2 GB
Hard disk drive	: 80 GB
Key Board	: Lenovo 108 keys
Mouse	: Logitech Optical Mouse
Motherboard	: Intel
Monitor	: 17” Lenovo

3.2.2 Software requirements

Operating System	:Windows 7
Front end used	:DotNet
Back End	:Sql server 2000

3.1.3 SOFTWARE DESCRIPTION

The language specification does not state the code generation requirements of the compiler: that is, it does not state that a C# compiler must target a Common Language Runtime, or generate Common Intermediate Language (CIL), or generate any other specific format. Theoretically, a C# compiler could generate machine code like traditional compilers of C++ or Fortran.

Some notable features of C# that distinguish it from C and C++ (and Java, where noted) are:

- C# supports strongly typed implicit variable declarations with the keyword var, and implicitly typed arrays with the keyword new[] followed by a collection initialize.
- Meta programming via C# attributes is part of the language. Many of these attributes duplicate the functionality of GCC's and Visual C++'s platform-dependent preprocessor directives.
- Like C++, and unlike Java, C# programmers must use the keyword virtual to allow methods to be overridden by subclasses.
- Extension methods in C# allow programmers to use static methods as if they were methods from a class's method table, allowing programmers to add methods to an object that they feel should exist on that object and its derivatives.

3.1.3.1 Design goals of c#

- The C# language is intended to be a simple, modern, general-purpose, object oriented programming language.

- The language, and implementations thereof, should provide support for software engineering principles such as strong type checking, array bounds checking, detection of attempts to use uninitialized variables, and automatic garbage collection. Software robustness, durability, and programmer productivity are important.
- The language is intended for use in developing software components suitable for deployment in distributed environments.
- Source code portability is very important, as is programmer portability, especially for those programmers already familiar with C and C++.

3.3 IMPLEMENTATION RESULTS

This chapter is devoted to explain the simulation results because results of the project promise the effective operation of the model that has been designed .The effective pattern generation is carried using the methods frequent term distribution, pattern taxonomy, pattern deploying and enhanced inner pattern evolving approach are analysed in this chapter. The platform used for this is dotnet (c#).

For testing the validity of the proposed model patterns were selected from the xml file. The data sets were preprocessed using start index and end index parameters. The generated pattern may have a significant effect on the results of the conventional methods like pattern taxonomy pattern deploying, inner pattern deploying approach. The application of enhanced inner pattern evolving in the proposed model creates a significant improvement in the accuracy level and increase in pattern generation.

3.3.1DATA PREPROCESSING

This process involves the individual term generation that is present in the xml file. The terms are splitted and placed in ten documents and this document is used as input in order to calculate the terms along with its overall counts they presented. To avoid complexity in manual calculation instead of taking all terms here the author names and some books titles were chosen. The sample xml file is shown the Fig.3.1.Data preprocessing is carried out by removing the start tag ant end tag. The sample document with patterns is also shown in the Fig.3.2.

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<!DOCTYPEdblp SYSTEM "dblp.dtd">
<dblp>
<incollectionmdate="2002-01-03"
key="books/acm/kim95/AnnevelinkACFHK95">
<author>JurgenAnnevelink</author>
<author>RafiulAhad</author>
<author>Amelia Carlson</author>
<author>Daniel H. Fishman</author>
<author>Michael L. Heytens</author>
<author>William Kent</author>
```

Fig.3.1 XML File

Document1:

JurgenAnnevelink

RafiulAhad

Amelia Carlson

Daniel H. Fishman

Michael L. Heytens

William Kent

Object SQL - A Language for the Design and Implementation of Object Databases.

42-68

1995

Modern Database Systems

db/books/collections/kim95.html#AnnevelinkACFHK95

José A. Blakeley

OQL[C++]: Extending C++ with an Object Query Capability.

69-88

Modern Database Systems

db/books/collections/kim95.html#Blakeley95

1995

Yuri Breitbart

Hector Garcia-Molina

Abraham Silberschatz

Transaction Management in Multidatabase Systems.

573-591

Fig 3.2 Sample Document

3.2.1.1 Existing System

Terms and the term distribution in documents is specified and based on the distribution the term weight is calculated using the Eqn 3.1 is shown in the Table3.1. Consider the term Aatsushi Nakazava its total count is two and it is present in the document 9 based on the distribution the term weight is calculated.

3.2.1.2 Proposed System

In the proposed system the term weight is based on the calculation of term weight according to the distribution of the term in every document. Take the Aatsushi Nakazava its total count is two and it is exactly present in nineth document and its related information is distributed in all the documents. Thus based on the distribution of the term the term weight is calculated and using that the average value is calculated. In the Table 3.2 the distribution based term weight is calculated and they are shown.

$$W(t) = \frac{TF(k+1)}{k*((1-b)+b*\left(\frac{DL}{AVDL}\right)+TF)} \qquad 3.1$$

TF=Term Frequency

k= 1.5

b= 0.5

DL= Document Length

AVDL= Average Document Length

Table 3.1.Term weight for Inner pattern Evolving Approach

S.no	Term	Termcount	Document frequency	Document term count	Term weight
1.	Atsushi Nakazava	2	1	9	2.826
2.	Daniel Thalman	3	3	2,7,9	0.474 2.906 2.826
3	Joachim P.Walser	1	1	5	0.718
4	Richard T.Shodgrass	33	2	3,5	1.033 1.025
5	Knowledge Discovery in databases	31	1	5	1.572
6	The Computer Science Engineering Handbook	114	1	2	1.587
7	R.G.G.Catter	10	3	2,5,6	0.592 0.718 2.442
8	Object Modelling	1	1	7	2.83
9	Modelling of Business application	1	1	8	2.20
10	Object Oriented Databases	24	1	1	1.542
11	Introduction To modern retrieval	2	2	5,6	2.452 0.715
12	Topics in Information Systems	2	1	7	3.663
13	Nickine	3	1	5	1.137
14	Jean H.galler	1	1	2	1.006
15	Antino C. Gome	3	2	2,1	0.806 0.762

Table 3.2 Term Weight For Enhanced Inner Pattern Evolving Approach

S.no	Term	Term count	Doc Freq	Doc term count	weight1	weight 2	weight 3	weight 4	weight 5	weight 6	weight 7	weight 8	weight 9	weight10
1.	Atsushi Nakazava	2	1	9	1.085	1.00	1.085	1.08	0.99	3.28	3.22	3.16	3.14	3.37
2.	Daniel Thalman	3	3	2,7,9	0.502	0.47	0.502	0.50	0.47	2.94	2.90	3.16	2.82	3.06
3	Joachim P.Walser	1	1	5	0.819	0.73	0.819	0.82	0.07	2.44	2.33	2.83	2.25	3.06
4	Richard T.Shodgrass	33	2	3,5	1.033	1.02	1.033	1.03	1.02	4.25	4.84	4.83	4.88	4.82
5	Knowledge Discovery in databases	31	1	5	1.55	1.54	1.55	1.86	1.5	4.96	4.82	4.97	4.95	4.96
6	The Computer Science Engineering Handbook	114	1	2	1.59	1.58	1.59	1.58	4.96	3.68	2.25	3.06	3.54	4.968
7	R.G.G.Catter	10	3	2,5,6	0.604	0.59	0.604	0.6	0.58	3.59	3.57	3.68	3.54	3.62
8	Object Modelling	1	1	7	0.819	0.73	0.819	0.82	0.07	2.44	2.33	2.83	2.25	3.06
9	Modelling of Business application	1	1	8	0.819	0.73	0.819	0.82	0.07	2.44	2.33	2.83	2.25	3.06
10	Object Oriented Databases	24	1	1	1.55	1.54	1.55	1.86	1.5	4.96	4.82	4.97	4.95	4.96
11	Introduction To modern retrieval	2	2	5,6	0.819	0.73	0.819	0.83	0.09	2.32	2.33	2.92	2.22	3.06
12	Topics in Information Systems	2	1	7	1.085	1.00	1.085	1.08	0.99	3.28	3.22	3.16	3.14	3.37
13	Nickine	3	1	5	1.216	1.14	1.21	1.13	1.13	3.22	3.98	3.56	3.71	3.79
14	Jean H.galler	1	1	2	1.05	1.02	1.089	1.09	0.7	3.23	3.33	3.26	3.34	3.37
15	Antino C. Gome	3	2	2,1	0.806	0.76	0.80	0.75	3.24	3.20	3.49	3.61	3.12	3.39

3.2.1.3.Accuracy

The accuracy of the pattern generation process is calculated using Eqn 3.2

$$A = \frac{p}{T} \qquad 3.2$$

Where

p = pattern generated

T = total number of documents

3.2.1.4.Pattern generation

The pattern generation is based on the k value and m value which is used to decide the threshold value range .Thus in the Table 3.4 the exact needed document and corresponding generation of pattern in the Inner pattern evolving approach and enhanced inner pattern evolving approach are compared

Table 3.3. Accuracy comparison of Inner Pattern Evolving Approach and Enhanced Inner Pattern Evolving Approach

S.no	Term	Classification Accuracy(%)	
		Inner Pattern Evolving Method	Enhanced Inner Pattern Evolving Approach
1	Atsushi Nakazava	10	70
2	Daniel Thalman	30	50
3	Joachim P.Walser	10	60
4	Richard T.Shodgrass	20	70
5	Knowledge Discovery in databases	10	80
6	The Computer Science Engineering Handbook	10	50
7	R.G.G.Catter	30	60
8	Object Modelling	10	50
9	Modelling of Business application	10	60
10	Object Oriented Databases	10	50
11	Introduction To modern retrieval	20	70
12	Topics in Information Systems	10	80
13	Nickine	10	60
14	Jean H.galler	10	70
15	Antino C. Gome	20	80

Table 3.4 .Pattern generation comparison of Inner Pattern Evolving Approach and Enhanced Inner Pattern Evolving Approach

S.no	k value	m value	Number of Relevant patterns in the documents	Patterns retrieved in Inner Pattern Evolving Approach	Patterns retrieved in Enhanced Inner Pattern Evolving Approach
1	34	6	7	3	5
2	15	3	5	2	3
3	78	20	22	14	18
4	112	34	35	24	30
5	160	26	28	23	25
6	55	18	20	12	16
7	80	24	26	20	23
8	114	46	48	38	40
9	93	22	24	15	20
10	220	45	40	32	38
11	25	8	12	5	7
12	40	12	13	9	11
13	52	16	18	10	14
14	34	12	14	8	10
15	20	5	8	3	5

The accuracy comparison is shown in the Fig.3.3 in which the existing system accuracy reached up to 40% but proposed system accuracy extend up to 80%.In the proposed Enhanced approach the term weight distribution is used to classify the relevant documents but in the case of existing approach the classification based on presence of exact pattern in the particular document

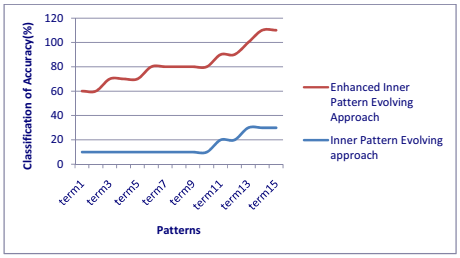


Fig. 3.3 Accuracy Comparison Inner Pattern Evolving Approach and Enhanced Inner Pattern Evolving Approach

The pattern generation of existing and proposed system are shown in Fig.3.4 in which the pattern generation for proposed system is high when compared to existing system .The k value and m value is used in the algorithm to generate the relevant pattern based on the term weight since in the case of inner pattern evolving approach term weight is calculated only for the document where it is exactly present but in the case of enhanced

approach the term weight is average term weight thus large number of pattern is generated with high accuracy indicates more information will be extract by the users.

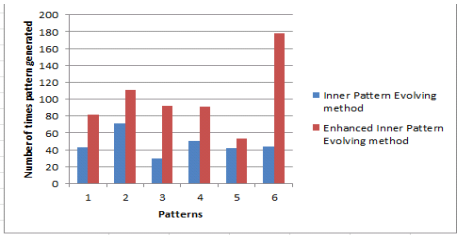


Fig.3.4 Comparison of patterns generated by standard and proposed methods

CONCLUSION AND FUTURE WORK

Many data mining techniques have been proposed in the last decade. These techniques include association rule mining, frequent item set mining, sequential pattern mining, maximum pattern mining, and closed pattern mining. However, using these discovered knowledge (or patterns) in the field of text mining is difficult and ineffective. The reason is that some useful long patterns with high specificity lack in support (i.e., the low-frequency problem). Hence, misinterpretations of patterns derived from data mining techniques lead to the ineffective performance. The proposed technique uses two processes, Enhanced pattern deploying and pattern progressing, to refine the discovered patterns in text documents with reduced influence of noise patterns. Future work may focus to balance between low and high threshold values.

APPENDIX

SAMPLE SOURCE CODE

TEXT DATA WEIGHT CALCULATION

```
using System;
usingSystem.Collections.Generic;
usingSystem.ComponentModel;
usingSystem.Data;
usingSystem.Drawing;
usingSystem.Text;
usingSystem.Windows.Forms;
usingSystem.Xml;
using System.IO;
usingSystem.Data.SqlClient;
usingSystem.Runtime.InteropServices;
namespace WindowsApplication9{
publicpartialclassFormTextData : Form
{SqlConnectioncon=newSqlConnection("DataSource=.;Initial
Catalog=DeterminingObjects;Integrated Security=True");
publicFormTextData()
{InitializeComponent();
```



```
}
privatevoid button1_Click(object sender, EventArgs e)
{
toolTip1.Show("processing", button1, 0, 0);
StreamReader re = File.OpenText("dblp.xml");
System.IO.TextWriterwr = newStreamWriter("output.txt");
string input = null;
stringouput = "";
intstrindex = 0;
intendindex = 0;
while ((input = re.ReadLine()) != null)
{
output = input.Substring(input.IndexOf(">"), input.LastIndexOf("</"));
strindex = 0;
endindex = 0;
strindex = input.IndexOf(">");
endindex = input.IndexOf("</");
MessageBox.Show(endindex.ToString ());
if (endindex != -1 &&endindex!= 0 &&strindex<endindex )
{
ouput = input.Substring(strindex+1, endindex - strindex-1);
```



```
listBox1.Items.Add(ouput);
}
}
re.Close();
wr.Flush();
wr.Close();
Hide(button1);}
privatevoid timer1_Tick(object sender, EventArgs e)
{
int b;
for (int i = 0; i < 1000; i++)
{
b= i + 1;

label2.Text = b.ToString()
}}
privatevoid button2_Click(object sender, EventArgs e)
{
toolTip1.Show("processing.....", button2, 0, 0);
con.Open();
SqlDataAdapter da = newSqlDataAdapter("select term from output1", con);
```

```
DataTabledt = newDataTable();
da.Fill(dt);
for (int i = 0; i <dt.Rows .Count ; i++) ////
{try
{
inttermcount = 0;
int termcount_doc1=0;
int termcount_doc2=0;
int termcount_doc3=0;
int termcount_doc4=0;
SqlCommandcmd = newSqlCommand("update output1 set termcount=" + termcount + " where term='" + dt.Rows[i][0].ToString() + "' ", con);
cmd.ExecuteNonQuery();
da2.Dispose();
dt2.Clear();
}
}
catch
{stringtname = "";
stringcolname = "";
tname = string.Concat("doc", cnt);
```



```
colname = string.Concat("termcount_doc", cnt);
inttermcount = 0;
string b = "";
b=dt.Rows[i][0].ToString();
string c = "";
c = b.Replace("'", "''");
SqlDataAdapter da2 = new SqlDataAdapter("select count(*) from " + tname + "  where term='" + dt.Rows[i][0] + "' ", con);
SqlDataAdapter da2 = new SqlDataAdapter("select count(*) from output  where term='" + dt.Rows[i][0] + "' ", con);
DataTable dt2 = new DataTable();
da2.Fill(dt2);
termcount = Convert.ToInt32(dt2.Rows[0][0].ToString());
SqlCommandcmd = new SqlCommand("update output1 set " + colname + "=" + termcount + " where term='" + dt.Rows[i][0].ToString() + "' ", con);
SqlCommandcmd = new SqlCommand("update output1 set termcount=" + termcount + " where term='" + dt.Rows[i][0].ToString() + "' ", con);
cmd.ExecuteNonQuery();
da2.Dispose();
dt2.Clear();}
}
con.Close();
```



```
toolTip1.Hide(button2);}
privatevoid button3_Click(object sender, EventArgs e)
{progressBar1.Visible = true ;
toolTip1.Show("Processing...", button3, 0, 0);
con.Open();
SqlDataAdapter da=newSqlDataAdapter ("select * from output1",con );
DataTabledt = newDataTable();
da.Fill(dt);
double wght1 = 0.0f;
double wght2 = 0.0f;
doublefinalwght = 0.0f;
double finalwght1 = 0.0f;
double k1 = 0.0f;
doubledj =0.0f;
doubledfj = 0.0f;
double dl = 0.0f;
double b=0.0f;
double N = 0.0f;
int doc_count1;
string doc_count1;
int doc_count2;
```



```
int doc_count3 = 0;
int doc_count4 = 0;
int doc_count5 = 0;
int doc_count6 = 0;
int doc_count7 = 0;
int doc_count8 = 0;
int doc_count9 = 0;
int doc_count10 = 0;
stringtbldoccount = "";
string[] doc_count = newstring[10];
for (intdoci = 1; doci< 11; doci++)
{
string tbldocname = ""
tbldoccount = "";
tbldocname = string.Concat("doc", doci);
tbldoccount = string.Concat("doc_count", doci);
SqlDataAdaptertbldoc_da = newSqlDataAdapter("select count(*) from " + tbldocname + "", con);
DataTabletbldoc_dt = newDataTable();
tbldoc_da.Fill(tbldoc_dt);
tbldoccount = tbldoc_dt.Rows[0][0].ToString();
```

```
doc_count[doci - 1] = tbldoc_dt.Rows[0][0].ToString();
}
k1 = 1;
b = 0.75;
N = 10;
progressBar1.Minimum = 0;
progressBar1.Maximum = dt.Rows.Count;
intpvalue = 0;
for (int i = 0; i <dt.Rows .Count  ; i++)
{pvalue = 0;
progressBar1.Value = i;
if (i ==dt.Rows .Count -1 )
{
progressBar1.Visible = false;
}
wght1 = 0.0f;
wght2 = 0.0f;
finalwght = 0.0f;
dj = 0.0f;
dfj = 0.0f;
stringtermname = "";
```



```
stringtermwgtname = "";
stringtermcolname="";
try
{
for (int j = 6; j < 11; j++)
{
termname = "";
termwgtname = "";
termcolname = "";
wght1 = 0;
wght2 = 0;
finalwght = 0;
dfj = 0;
dl = 0
termname = string.Concat("termcount_doc", j);
termwgtname = string.Concat("termweight", j);
dj = Convert.ToDouble(dt.Rows[i]["termcount"].ToString());
termcolname = dt.Rows[i]["term"].ToString();
fj = Convert.ToDouble(dt.Rows[i]["docfrequency"].ToString());
dl = Convert.ToDouble(doc_count[j - 1].ToString());
wght1 = ((k1 + 1) * dj) / ((k1 * ((1 - b) + (b * (dl  / 8000)))) + dj);
```



```
wght2 = Math.Log10 (((N - dfj) + 0.5) / (dfj + 0.5));
finalwght = wght1 * wght2;
finalwght1 = System.Math.Round(finalwght, 3);
SqlCommandcmd = newSqlCommand("update output1 set " + termwgtname + " = " + finalwght1
+ " where term='" + termcolname + "'", con);
cmd.ExecuteNonQuery();}}
MessageBox.Show("completed.....");
con.Close();

toolTip1.Hide(button3);
}
privatevoidFormTextData_Load(object sender, EventArgs e)
{progressBar1.ForeColor = Color.Red;}
privatevoid progressBar1_Click(object sender, EventArgs e)
privatevoid button4_Click(object sender, EventArgs e)
{toolTip1.Show("Processing", button4, 0, 0);
con.Open();
SqlDataAdapter da = newSqlDataAdapter("select * from output1", con);
DataTabledt = newDataTable();
da.Fill(dt);
for (int i = 0; i <dt.Rows.Count ; i++)
```



```
{
stringtermname = "";
termname = dt.Rows[i][0].ToString();
intdocfrequency = 0;
inttermdoc_value = 0;
for (int j = 3; j < 13; j++)
{
termdoc_value = Convert.ToInt32(dt.Rows[i][j].ToString());
if (termdoc_value> 0)
{docfrequency++;}
}try
{SqlCommandcmd = newSqlCommand("update output1 set docfrequency=" + docfrequency + "
where term='" + termname + "' ", con);
cmd.ExecuteNonQuery();
}catch
con.Close();
toolTip1.Hide(button4);
}
privatevoid button5_Click(object sender, EventArgs e)
{
toolTip1.Show("Processing", button5, 0, 0);
```

```
con.Open();

SqlDataAdapter da = newSqlDataAdapter("select * from output1", con);

DataTabledt = newDataTable();

da.Fill(dt);

for (int i = 0; i <dt.Rows .Count ; i++)

{

try

{

double wgt1 = 0.0f;

double wgt2 = 0.0f;

wgt1=Convert.ToDouble(dt.Rows[i]["termweight1"].ToString())+
Convert.ToDouble(dt.Rows[i]["termweight2"].ToString())+
Convert.ToDouble(dt.Rows[i]["termweight3"].ToString())+
Convert.ToDouble(dt.Rows[i]["termweight4"].ToString())+
Convert.ToDouble(dt.Rows[i]["termweight5"].ToString())+
Convert.ToDouble(dt.Rows[i]["termweight6"].ToString())+
Convert.ToDouble(dt.Rows[i]["termweight7"].ToString())+
Convert.ToDouble(dt.Rows[i]["termweight8"].ToString())+
Convert.ToDouble(dt.Rows[i]["termweight9"].ToString())+
Convert.ToDouble(dt.Rows[i]["termweight10"].ToString());

wgt2 = wgt1 / 10;

SqlCommandcmd = newSqlCommand("update output1 set avgtermweight=" + wgt2 + " where term='" + dt.Rows[i][0].ToString() + "'", con);

cmd.ExecuteNonQuery();
```



```
}catch

}

con.Close();

toolTip1.Hide(button5 );

frmDisplayTerms f1 = newfrmDisplayTerms();

f1.Show();

}}}
```


SCREEN SHOTS

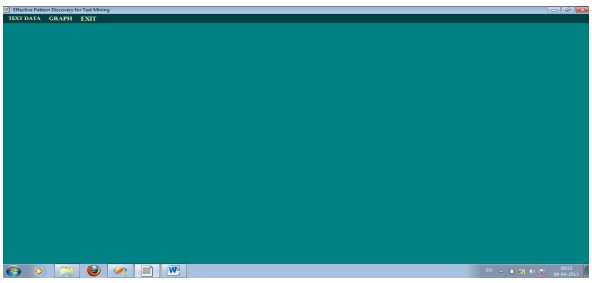


Fig. A1 Text data weight calculation

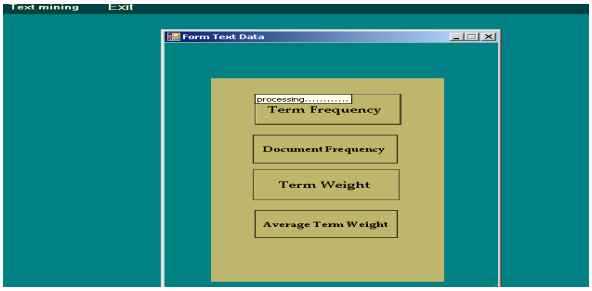


Fig. A2. Term frequency is processed

term	termcount
Atsushi Nakazawa	2
Daniel Thalmann	3
73-92	1
Joachim P. Walser	1
db/conf/3dim/3d...	1
Richard T. Snod...	33
0-9660746-0-2	1
3-540-41904-7	1
Minorities and the...	1
Mathematical ind...	1
Understanding S...	1
47	3
Christos Berberidis	1
journals/tods/Ad...	1
Arjan Durresi	1
db/conf/3dim/3d...	1
books/mk/Cattell...	2
db/books/collect...	1
A Model for Com...	1

Fig.A3. Term with term count

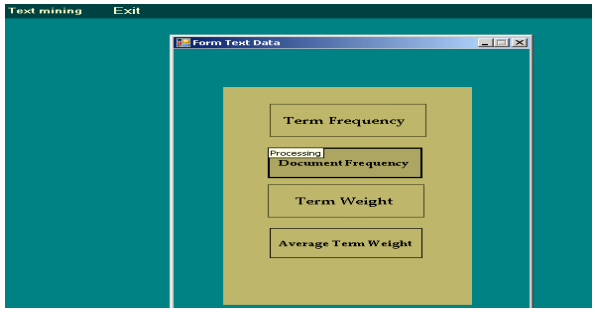
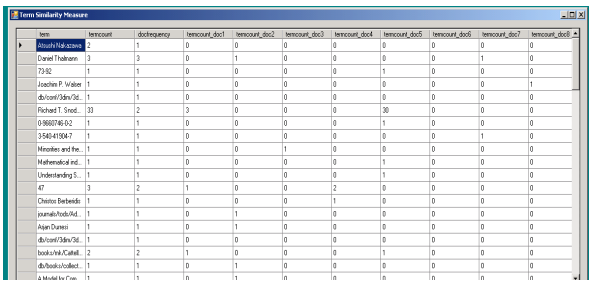
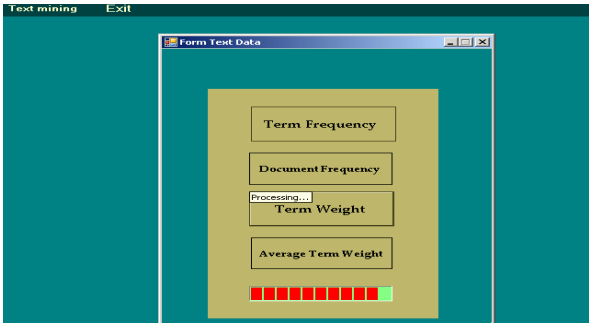


Fig. A4 Document frequency



FigureA5 Term count in documents



FigA6Term weight calculation

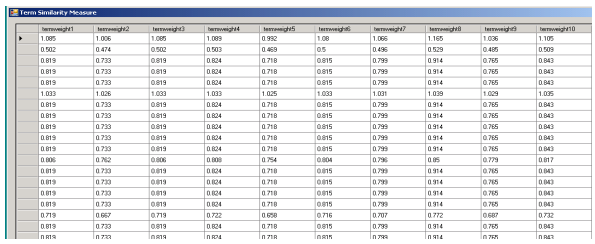


Fig.A7 Term weight in documents

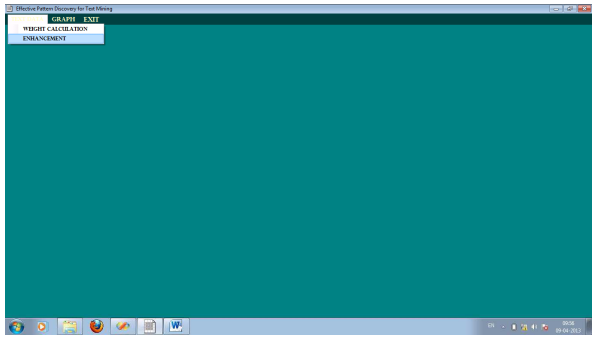


Fig.A8 Enhancement processing

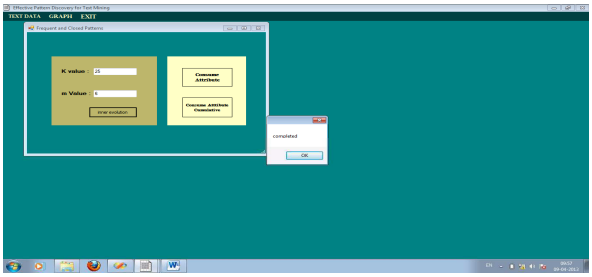


Fig.A9 Pattern generation

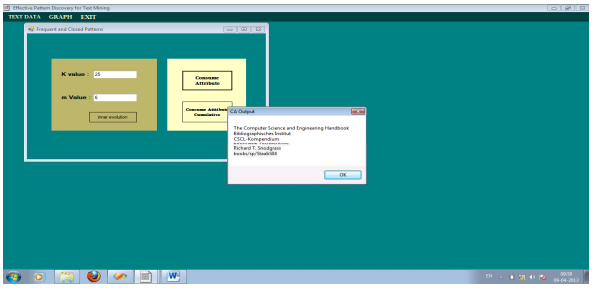


Fig.A10 Pattern generation for Inner Pattern evolving Method

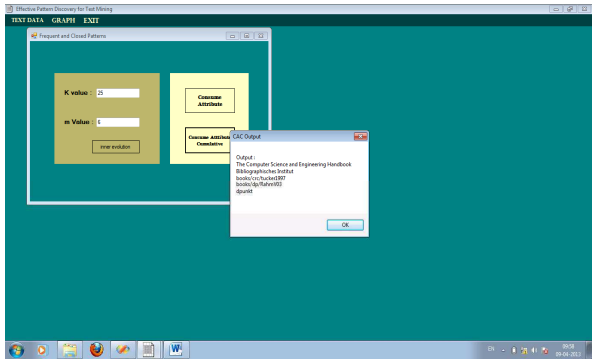


Fig.A11 Pattern generation for Enhanced Inner Pattern evolving Method

REFERENCES

[1] Han, J., Pei, J. and Yin, Y. “Mining Frequent Patterns without Candidate Generation,” Proc.ACM SIGMOD Int’l Conf. Management of Data (SIGMOD ’00), pp. 1-12, 2000.

[2] Lewis, D.D. “Feature Selection and Feature Extraction for Text Categorization,” Proc. Workshop Speech and Natural Language, pp. 212-217, 1992.

[3] Li X. and Liu B. “Learning to Classify Texts Using Positive and Unlabeled Data,” Proc. Int’l Joint Conf. Artificial Intelligence (IJCAI ’03), pp. 587-594, 2003.

[4] 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.

[5] Pei, J., Han J, Mortazavi-Asl, B., Pinto, H., Chen, Q., Dayal, U., and Hsu, M. “Prefixspan: Mining Sequential Patterns Efficiently by Prefix-Projected Pattern Growth,” Proc. 17th Int’l Conf. Data Eng. (ICDE ’01), pp. 215-224, 2001.

[6] Shehata, S., Karray, F. and Kamel, M. “A Concept-Based Model for Enhancing Text Categorization,” Proc. 13th Int’l Conf. Knowledge Discovery and Data Mining (KDD ’07), pp. 629-637, 2007.

[7] Sebastiani, F.“Machine Learning in Automated Text Categorization,” ACM Computing Surveys, vol. 34, no. 1, pp. 1-47, 2002.

[8] Wu, S T., Li, Y. and Xu, Y. “Deploying Approaches for Pattern Refinement in Text Mining,” Proc. IEEE Sixth Int’l Conf. Data Mining (ICDM ’06), pp. 1157-1161, 2006.

[9] Wu, S T., Li,Y., Xu,Y., Pham, B. and Chen, P.“Automatic Pattern- Taxonomy Extraction for Web Mining,” Proc. IEEE/WIC/ACM Int’l Conf.Web Intelligence (WI ’04), pp. 242-248, 2004.

LIST OF PUBLICATIONS

1. C.Sree Devi, D.Chandrakala **“**Enhanced Inner Pattern Evolving Approach for Discovered Patterns”**,** International Conference, Advance in Computer and Communication, Sun College of Engineering and Technology Nagercoil, 18th February 2013.

2. C.Sree Devi, D.Chandrakala “Enriched Inward Pattern Progressing Methodology for Discovered Patterns”**,** National Conference, Innovation on Information Technology, Bannari Amman Institute of Technology, Coimbatore, 22nd February 2013.