AN EFFICIENT KEYWORD BASED SEARCHING FOR MINING FREQUENT DATASETS

N. Sivashanmugam
Assistant Professor
School Of Computer Science
Tamil Nadu Open University, Chennai-15.

Dr. C. Jothi Venkateswaran
Associate Professor & Head,
Dept. Of Computer Science
Presidency College, Chennai-5.

ABSTRACT: Data Mining extracts knowledge from large databases to discover existing and newer patterns. Data mining is the technique of automatic finding of hidden valuable patterns and relationships from huge volume of data stored in databases in order to help make better business decisions. Discovering useful patterns hidden in a database plays an essential role in several data mining tasks. Frequent patterns are patterns (such as itemsets, subsequences, or substructures) that appear in a data set frequently. A substructure can refer to different structural forms, such as subgraphs, subtrees, or sublattices, which may be combined with itemsets or subsequences. If a substructure occurs frequently, it is called a (frequent) structured pattern. Finding such frequent patterns plays an essential role in mining associations, correlations, and many other interesting relationships among data. Moreover, it helps in data classification, clustering, and other data mining tasks as well. Thus, frequent pattern mining has become an important data mining task and a focused theme in data mining research. Frequent itemsets find application in a number of real- life contexts. The proposed system retrieves the optimized dataset in an efficient manner thereby mining the efficient data from the wide range of datasets.

INTRODUCTION:

Data mining, the extraction of hidden predictive information from large databases, is a powerful technology with great potential to help companies focus on the most important information in their data warehouses. Data mining tools predict future trends and behaviors, allowing businesses to make proactive, knowledge-driven decisions. The automated, prospective analyses offered by data mining move beyond the analyses of past events provided by retrospective tools typical of decision support systems. Data mining tools can answer business questions that traditionally were too time consuming to resolve. They scour databases for hidden patterns, finding predictive information that experts may miss because it lies outside their expectations.

Outlier detection is an important branch in data mining, which is the discovery of data that deviate a lot from other data patterns. In other related area dealing with detecting outliers is clustering algorithms where outliers are objects not located in clusters of a dataset, and these algorithms generate outliers as by product. Outliers can occur by chance in any distribution, but they are often indicative either of measurement error or that the population has a heavy-tailed distribution. A frequent cause of outliers is a mixture of two distributions, which may be two distinct sub-populations, or may indicate 'correct trial' versus 'measurement error'; this is modelled by a mixture model. Outliers, being the most extreme observations, may include the sample maximum or sample minimum, or both, depending on whether they are extremely high or low. However, the sample maximum and minimum are not always outliers because they may not be unusually far from other observations.

LITERATURE SURVEY:

Conventional Data Management Strategies

In conventional information management principles, the stored records are normally identified by sets of key words or index terms, and requests for information are expressed by using Boolean combinations of index terms. The retrieval strategy is normally based on an auxiliary inverted-term index that lists the corresponding set of document references for each allowable index term. The Boolean retrieval system is designed to retrieve all stored records exhibiting the precise combination of key words included in the query: when two query terms are related by an and connective, both terms must be present in order to retrieve a particular stored record; when an or connective is used, at least one of the query terms must be present to retrieve a particular item.

In some systems where the natural language text of the documents or the document excerpts is stored, the user queries may be formulated as combinations of text words. In that case, the queries may include location restrictions for the query terms- for example, a requirement that the query terms occur in the same sentence of any retrieved document or within some specified number of words of each other.

Boolean data management systems have become popular in operational situations because high standards of performance are achievable. The retrieval technology which is based on list intersections and list unions to implement Boolean conjunction *("A* and B") and Boolean disjunction *("A* **or** B"), respectively, is now well understood. The conventional Boolean retrieval technology is however also saddled with various disadvantages:

1. The size of the output obtained in response to a given query is difficult to control; depending on the assignment frequency of the query terms and the actual term combinations used in a query formulation, a great deal of output can be

obtained or, alternatively, no output might be retrieved at all.

2. The output obtained in response to a query is not ranked in any order of presumed importance to the user; thus, each retrieved item is assumed to be as important as any other retrieved item.

3. No provisions are made for assigning importance factors or weights to the terms attached either to the documents or dataset.

Data Evaluation

Lee gives an overview of the different models that have been proposed, and shows that only the p-norm model has two key properties that, if not present, are detrimental to retrieval effectiveness.

Smith proposed to recursively aggregate inverted lists, calculating and storing intermediate scores for every document that is encountered in any of the lists, in what is referred to as being the term-at-a-time approach. In effect, not all nodes in the query tree are visited for every document, but all of the inverted lists are fully inspected, and temporary memory proportional to the total size of the relevant inverted lists is required. Smith's Infinity-One method gives an approximation to the p-norm model, with the aim of reducing computational cost by reducing the volume of floating point operations. As is demonstrated below, the number of score calculations can be greatly reduced via an exact lossless pruning approach.

Turtle and Flood describe the max-score ranking mechanism, to accelerate keyword query evaluation when sum-score aggregation functions are used and only the top-k documents are required. Using document-at-a-time evaluation, the algorithm commences by fully scoring the first k documents in the OR-set of the query terms. Thereafter, the k^{th} largest document score is tracked, as an entry threshold that candidate documents must exceed before they can enter the (partial) ranking. The max-score algorithm uses the information conveyed by the entry threshold to reduce two cost factors: 1) the number of candidate documents that are scored; and 2) the cost associated with scoring each candidate document.

Content-based Methods

In content-based recommendation methods, the utility $u(c, s)$ of item s for user c is estimated based on the utilities $u(c, si)$ assigned by user c to items $si \in S$ that are “similar” to item s. For example, in a movie recommendation application, in order to recommend movies to user c, the content-based recommender system tries to understand the commonalities among the movies user c has rated highly in the past (specific actors, directors, genres, subject matter, etc.). Then, only the movies that have a high degree of similarity to whatever user’s preferences are would get recommended. The content-based approach to recommendation has its roots in information retrieval and information filtering research.

Because of the significant and early advancements made by the information retrieval and filtering communities and because of the importance of several text-based applications, many current content-based systems focus on recommending items containing textual information, such as documents, Web sites (URLs), and Usenet news messages. The improvement over the traditional information retrieval approaches comes from the use of user *profiles* that contain information about users’ tastes, preferences, and needs. The profiling information can be elicited from users explicitly, e.g., through questionnaires, or implicitly – learned from their transactional behavior over time.

More formally, let *Content*(*s*) be an *item profile*, i.e., a set of attributes characterizing item *s*. It is usually computed by extracting a set of features from item *s* (its content) and is used to determine appropriateness of the item for recommendation purposes. Since, as mentioned earlier, content-based systems are designed mostly to recommend text-based items, the content in these systems is usually described with *keywords*.

For example, a content-based component of the Fab system which recommends Web pages to users represents Web page content with the 100 most important words. Similarly, the Syskill & Webert system represents documents with the 128 most informative words. The “importance” (or “informativeness”) of word *ki* in document *dj* is determined with some *weighting* measure *wij* that can be defined in several different ways.

One of the best-known measures for specifying keyword weights in Information Retrieval is the *term frequency/inverse document frequency (TF-IDF)* measure that is defined as follows. Assume that *N* is the total number of documents that can be recommended to users and that keyword *ki* appears in *ni* of them. Moreover, assume that *i, j f* is the number of times keyword *ki* appears in document *dj*. Then *i, j TF*, the term frequency (or normalized frequency) of keyword *ki* in document *dj*, is defined as

$$TF_{i,j} = \frac{f_{i,j}}{\max_z f_{z,j}}$$

Limited content analysis:

Content-based techniques are limited by the features that are explicitly associated with the objects that these systems recommend. Therefore, in order to have a sufficient set of features, the content must either be in a form that can be parsed automatically by a computer (e.g., text), or the features should be assigned to items manually.

While information retrieval techniques work well in extracting features from text documents, some other domains have an inherent problem with automatic feature extraction. For example, automatic feature extraction methods are much harder to apply to the multimedia data, e.g., graphical images, audio and video streams. Moreover, it is often not practical to assign attributes by hand due to limitations of resource Another problem with limited content analysis is that, if two different items are represented by the same set of features, they are indistinguishable.

Therefore, since text-based documents are usually represented by their most important keywords, content-based systems cannot distinguish between a well-written article and a badly written one, if they happen to use the same terms.

Over-specialization. When the system can *only* recommend items that score highly against a user's profile, the user is limited to being recommended items similar to those already rated. For example, a person with no experience with Greek cuisine would never receive a recommendation for even the greatest Greek restaurant in town. This problem, which has also been studied in other domains, is often addressed by introducing some randomness. For example, the use of genetic algorithms has been proposed as a possible solution in the context of information filtering. In addition, the problem with over-specialization is not only that the content-based systems cannot recommend items that are different from anything the user has seen before. In certain cases, items should not be recommended if they are *too similar* to something the user has already seen, such as different news article describing the same event. Therefore, some content based recommender systems, such as DailyLearner, filter out items not only if they are too different from user's preferences, but also if they are too similar to something the user has seen before.

RELATED WORK:

About some mining concepts,

1. Association:

This technique is useful for finding relationship between data in a dataset. It comprises of antecedent and a consequent part. Association Rule Mining is used for finding frequent data using the terminologies support and the confidence value. Support value is the one related to how many times the data get repeated in the database and the confidence value is the one counts for the combination of if then statements become true. For example if a person buys strawberry he likely to purchase cream for it and the vice versa

2. Classification:

This technique involves classifying data based on its attributes or features. For example car can be classified based on its features like wheels, seat type, colour etc. , people can be classified based on Age, Gender, work etc.

3. Clustering:

Clustering is based on grouping similar data or values based on the attributes or features. Based on classified data, similar datasets are clustered.

5. Prediction:

Prediction is about predicting the result of particular event. It is based on mining similar patterns that occur in the past.

PROPOSED WORK:

This paper involves computation of keyword based searching for mining frequent data sets efficiently. The first step involves computation of lexicons for the user given query. Then they are mapped with the each word of a dataset. Based on it the word count will be computed. The minimum threshold value is set. The threshold differs based on the context of the application. The datasets where word count matches with the threshold value are taken. Then they are matched with datasets from some other resource by computing similarity between the datasets present in two different resources. The similar datasets are extracted and its hitcount are captured. The datasets are given weight based on the word count and the hitcount. The matched datasets are sorted in descending order for presenting it. Since the growth of data becomes large in this internet era, the mining of frequent pattern becomes essential. The proposed work is expected to give frequent datasets on demand.

ALGORITHM:

Step 1: input query

Step 2: formation of lexicons for the user given query.

Step 3: Matching the input query with the datasets from the given resource.

Implementing simple word count program in java for getting count for the presence of computed lexicons in the dataset.

Step 4: Computation of cosine similarity between the matched datasets and the datasets from different resources.

Formula for computing cosine similarity,

theta = crossprod(a, b) / (sqrt(crossprod(a, a)) * sqrt(crossprod(b, b)))

Step 5: from similarity values the resultant datasets are retrieved and hitcounts are captured.

The datasets are weighted based on it and are sorted in descending order.

CONCLUSION:

Thus this system has been developed to face the issues of identifying outliers in large datasets. By Identifying outliers, it can be removed and the resources occupied by those outliers are freed-up. As a result the resources management is done in an efficient manner. The proposed system retrieves the optimized dataset in an efficient manner thereby detecting the outliers

FUTURE ENHANCEMENTS:

This system can be further extended to apply in detection of frequent item sets in multimedia data sets. In Multimedia data sets it is highly challenging to mine frequent datasets.

REFERENCES:

[1] A Review on Infrequent Weighted Itemset Mining using Frequent Pattern Growth- Shipra Khare , Prof. Vivek Jain

[2] A Method for Mining Infrequent Causal Associations and Its Application in Finding Adverse Drug Reaction Signal Pairs

[3] A Review on Efficient Mining Approach of Infrequent Weighted Itemset -IEEE Journal by Sonia Jadhav,G. M. Bhandari

[4] An Algorithm for Mining Maximum Frequent Itemsets Using Data-sets Condensing and Intersection PruningShui Wang, Ying Zhan, and Le Wang

[5] Minimally Infrequent Itemset Mining using Pattern-Growth Paradigm and Residual Trees -Ashish Gupta,Akshay Mittal,Arnab Bhattachary

[6] DataMining algorithms Reference http://www.cs.umd.edu/~samir/498/10Algorithms-08.pdf

[7] About DataMining Techniques- http://www.ibm.com/developerworks/library/ba-data-mining-techniques/

[8] Frequent Pattern Mining- www.charuaggarwal.net/freqbook.pdf

[9] Analysis of Frequent pattern www.analysis-of-patterns.net/files/bgoethals.pdf

[10] DataMining Algorithms for frequent pattern mining- https://en.wikibooks.org/wiki/Data_Mining.../Frequent_Pattern_Mining