Unstructured Data Mining - Methods and Techniques for Information Retrieval

Prof. Leena Deshmukh, AP
JSPM's Jayawant Institute of Management Studies,
Tathawade, Pune
linadeshmukh@gmail.com
9960639644

***Abstract** - Textual data is unstructured. The term information retrieval generally refers to the querying of unstructured textual data. Unstructured data mining adopts different techniques to discover and retrieve information from large collection of documents. Data contained in documents are unstructured without any associated schema. The process of information retrieved consists of locating relevant documents. In this paper commonly used methods and techniques like Text Retrieval - the query is regarded as specifying constraints for selecting relevant documents (Document selection, Document ranking, Vector space model), Text Indexing (Inverted indices, Signature file), and Query Processing (Relevance feedback – when examples of relevant documents are available system can learn from such examples to improve retrieval performance, pseudo-feedback – when we don't have such examples, a system can assume a top few retrieved documents tobe relevant & extract more related keywords), for retrieval of unstructured data are discussed with their merits and demerits.*

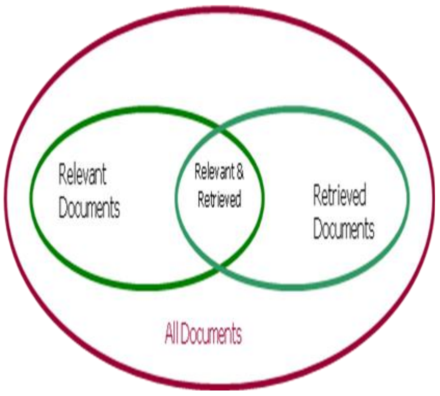
Keywords- Data mining, tools, precision, recall, F-score

INTRODUCTION

Now a day's most knowledge seekers are searching data through internet. This information is stored in the text databases or document databases, which consists of large collection of documents from various sources, such as news articles, research papers, books, digital libraries, e-mail messages and web pages. Text databases are rapidly growing due to increasing amount of information in each sector.
Data stored in text databases are semistructured data in that they are neither completely unstructured nor completely structured. There have been a great deal of studies on the modeling and implementation of unstructured data in recent data. Users need tools to compare different documents, rank the importance and relevance of the documents, or find patterns and trends across multiple documents.

BASIC MEASURES FOR TEXT RETRIEVAL: PRECISION AND RECALL

Let the set of documents relevant to a query be denoted as **{Relevant},** and the set of documents retrieved be denoted as **{Retrieved}**.
The set of documents that are both relevant
and retrieved is denoted as retrieved and denoted as **{Relevant}** ∩ **{Retrieved}**.



There are two basic measures for assessing quality of text retrieval:
Precision: This is the percentage of retrieval documents that are in fact relevant to the query (i.e. "correct" responses). It is formally defined as

$$precision = \frac{|\{Relevant\} \cap \{Retrieved\}|}{|\{Retrieved\}|}$$

Recall: This is the percentage of documents that are relevant to the query and were, in fact, retrieved. It is formally defined as

$$recall = \frac{|\{Relevant\} \cap \{Retrieved\}|}{|\{Relevant\}|}$$

An information system retrieval system often needs to trade off recall for precision or vice versa. One commonly used trade-off is the ***F-score***, which is defined as Harmonic mean of recall and precision:

$$F\text{-}score = \frac{recall \times precision}{(recall + precision) / 2}$$

TEXT RETRIEVAL METHODS

Retrieval methods fall into two categories: They generally either view the retrieval problem as ***document selection problem*** or as a ***document ranking problem***.

In ***document selection methods***, the query is regarded as specifying constraints for selecting relevant documents. A typical method of this category is the Boolean Retrieval model, in which a document is represented by a set of keywords and a user provides a Boolean expression of keywords, such as "car or repair shops", "tea or coffee", or "database systems but not Mysql". The retrieval system would take such a Boolean query & return documents that satisfy the Boolean expression. Because of difficulty in prescribing a users information need exactly with a Boolean query, the Boolean retrieval method generally only works well when the user knows a lot about the document collection and can formulate a good query in this way.

Document ranking methods use the query to rank all the documents in the order of relevance. For ordinary users and exploratory queries, these methods are more appropriate than document selection methods. Most modern information retrieval systems present a ranked list of documents in response to a users keyword query. There are many different ranking methods based on a large spectrum of mathematical foundations, including algebra, logic, probability and statistics. The common intuition behind all these methods is that matches the keywords in a query with those in the documents and score each document based on how well it matches the query. The goal is to approximate the degree of relevance of document with a score computed based on information such as the frequency of words in the document and the whole collection. It is inherently difficult to provide a precision measure of the degree of the relevance between a set of keywords. For example, it is difficult to quantify the distance between data mining and data analysis. Comprehensive empirical evaluation is thus essential for validating any retrieval method.

VECTOR SPACE MODEL

In this model we represent a document ad a query both as vectors in a high-dimensional space corresponding to all the keywords and use an appropriate similarity measure to compute the similarity between the query vector and the document vector. The similarity values can be then used for ranking documents.

To tokenize text, the first step in most retrieval systems is to identify keywords for representing documents, a preprocessing step often called tokenization. To avoid indexing useless words (a, an, the etc), a text retrieval system often associates a **stop list** with a set of documents. A stop list is a set of words that are deemed "irrelevant". It may vary per document set.

A group of different words may share the same word stem. A text retrieval system needs to identify groups of words where the words in group are small syntactic variants of one another and collect only the common word stem per group. For example the group of words drug, drugged and drugs share a common word stem, drug and can be viewed as different occurrences of the same word.

We can model a document to facilitate information retrieval as

Let, **d**=set of Documents, and

t=set of terms,

v=vector (each document) in the dimensional space $\mathbf{R}^{t}$.

Let, the **term frequency** be the number of occurrences of term t in the document **d, freq(d,t)**

The (weighted) **term-frequency matrix TF(d,t)** measures the association of a term **t** with respect to the given document **d**: it is generally defined as 0 if the document does not contain the term, nonzero otherwise. There are many ways to define term-weighting for the non zero entries in such a vector.

For example we can simply set **TF(d,t)=1** if the term **t** occurs in the document **d**, or use the term frequency **freq(d,t),** or relative term frequency, ie, the term frequency verses the total number of occurrences of all the terms in the document. There are also other ways to normalize the term frequency. For example, the Cornell SMART system uses the following formula to compute the (normalized) term frequency:

$$TF(d,t) = \begin{cases} 0 & if\ freq(d,t) = 0 \\ 1 + \log(1 + \log(freq(d,t))) & otherwise. \end{cases}$$

Inverse document frequency (IDF), which represents scaling factor or the importance of a term **t**. If a term **t** occurs in many documents, its importance will be scaled down due to its reduced descriminative power. For example the term database system may likely be less important if it occurs in many research papers in database system conference. According to the same Cornell SMART system, **IDF(t)** is defined by the following formula:

$$IDF(t) = \log \frac{1 + |d|}{|d_t|},$$

Where **d** is the document collection, and $\mathbf{d_t}$ is the set of documents containing term **t**. If |**dt**| << |**d**|, the term t will have a large IDF scaling factor and vice versa.

In a complete vector place model, **TF** and **IDF** are combined together, which forms the **TF-IDF** measure:

TF-IDF(d,t) = TF(d,t) x IDF(t)

TEXT INDEXING TECHNIQUES

An Inverted Index is an index structure that maintains two hash indexed or B+-tree indexed tables: document_table and term_table where

Document_table: consists of a set of document records, each containing two fields: doc_id and posting_list, where posting_list is a list of terms (pointer to terms) that occur in the document, sorted according to some relevance measure.

Term_table: consists of a set of term records, each containing two fields: term_id and posting_list, where posting_list specifies a list of document identifiers in which the term appears.

Using such techniques it is easy to answer queries like "Find all of the documents associated with a given set of terms" or "Find all of the terms associated with a given set of documents." For example, to find all of the documents associated with a set of terms, we can first find a list of document identifiers in term_table for each term, and then intersect them to obtain the set of relevant documents. Inverted indices are widely used in industry. They are easy to implement. The posting_lists could be rather long, making the storage requirement quite large. They are easy to implement, but are not satisfactory at handling ***synonymy*** (where two very different words can have the same meaning) and ***polysemy/homonyms***(where individual word may have many meanings)

A signature file is a file that stores a signature record for each document in the database. Each signature has a fixed size of b bits representing terms. Simple encoding scheme used is as follows. Each bit of document signature is initialized to 0. A bit is set to 1 if the term it represents appears in the document. A signature S1 matches another signature S2, if each bit that is set in signature S2 is also set in S1. Since there are usually more terms than available bits, multiple terms may be mapped into the same bit. Such multiple-to-one mapping make the search expensive because a document that matches the signature of a query does not necessarily contain the set of keywords of the query. The document has tobe retrieved, parsed stemmed and checked. Improvement can be made by first performing frequency analysis, stemming and by filtering stop words, and then using a hashing technique and superimposed coding technique to encode the list of terms into bit representation. The problem of multiple-to-one mappings still exists, which is the major disadvantage of this approach.

QUERY PROCESSING TECHNIQUES

Once inverted index is created for a document collection, a retrieval system can answer a keyword query quickly by looking up which documents contain the query keywords. Specifically it maintains a score accumulator for each document and update these accumulators as we go through each query term. For each query term, we will fetch all of the documents that For each query term, we will fetch all of the documents that match the term and increase their scores.

When examples of relevant documents are available, the system can learn from such examples to improve retrieval performance. This is relevance feedback and has proven tobe effective in improving retrieval performance. When we don't have relevant example, a system can assume the top few retrieved documents in some initial retrieval results tobe relevant and extract more related keywords to expand a query. Such feedback is pseudo-feedback or blind feedback and is essential process of mining useful keywords from the top retrieved documents. Pseudo-feedback also often leads to improved retrieval performance.

CONCLUSION

With similarity metrics introduced in this paper, we can construct similarity based indices on text documents. Text-based queries can then be represented as vectors, which can be used to search for their nearest neighbors in a document collection. In this paper High dimensionality metrics were not introduced.

One major limitation of many existing retrieval methods is that they are based on exact keyword matching. However due to complexity of natural languages, keyword-based retrieval can encounter two major difficulties. First is ***synonymy***: two words with identical or similar meanings may have very different surface forms. And second is ***polysemy (homonyms)***: same keywords may mean different things in different contexts.

REFERENCES

1. *R. A. Baeza-Yates and B.A. Ribeiro-Neto. Modern Information Retrieval. ACM Press/Addison-Wesley, 1999.*
2. I.H. Witten, A. Moffat, and T.C. Bell. Managing Gigabytes: Compressing and Indexing Documents and Images. Morgan Kaufmann, 1999.
3. Jaiwei Han, Micheline Kamber. Data Mining Concepts and Techniques. Elsevier, 2006-2009
4. G.K. Gupta. Introduction to Data Mining with case studies. PHI, 2013.
5. Michael J.A. Berry, Gordon S. Linoff. Data Mining Techniques. Wiley, 2010.
6. Abraham Silberschatz, Henry Korth, S. Sudarshan. Database System Concepts. McGraw Hill, 2011.
7. Margaret Dunham. Data Mining Techniques.
8. www.Google.com