VMARK Version 1.0.0 and Version 1.0.1
Search Engine for Amazon Food Reviews

Design Document

Academic Project for the Course CS F469
Information Retrieval

Name of the Student	ID Number
Aniketh Janardhan Reddy	2014A7PS096H
Kartik Sethi	2014A7PS130H
Monica Adusumilli	2014A7PS005H
Vinay Datta Renigunta	2014A7PS509H

BITS Pilani
Hyderabad Campus

ABSTRACT

Information retrieval is the activity of obtaining information resources relevant to an information need from a collection of information resources. The primary goal of most of the information retrieval systems is to enable users to retrieve the data that is relevant to their queries efficiently. The Vector Space Model(VSM) is a classical approach in Information Retrieval which models documents and queries as vectors in the term space. The components of the vectors are determined by the term weighting scheme, a function of the frequencies of the terms in the document or query as well as throughout the collection. VMARK is composed of two information retrieval systems – the syntactic VSM and the semantic VSM. The syntactic VSM is built using conventional term recognition, indexing and search optimisation techniques such as stemming using Porter's algorithm, standard inverted indexing and tf-idf weighting. It promises a high precision on queries which require raw term matching. But, it lacks the ability to recognise the semantics of the query. To alleviate this issue, the semantic VSM was built. It uses the WordNet to perform indexing based on part-of-speech (POS) tagging and synonyms based expansion. It promises high precision for queries whose terms do not exactly match the terms present in the documents. This document describes the design and implementation of the VMARK Search Engine 1.0.0 (Syntactic) and VMARK Search Engine 1.0.1(Semantic) and gives a comparative study of their precision performances.

LITERATURE REVIEW

An extensive study over the various implementations of Vector Space Models (VSM's) shows that a lot of studies in this domain are related to the classic tf-idf weighting and cosine normalisation. A review of the existing literature tells us that the efficiency of the retrieval depends on the efficiency of the term weighting schemes to a large extent. Limited amount of information exists on semantic VSMs and their implementations. As far as we know, there are almost no papers that compare and contrast the syntactic and semantic VSMs and the advantages and disadvantages of one over the other. Also, the knowledge base on analysing the context of the query is very limited.

VMARK uses weighting schemes proposed by the paper titled 'New Term Weighting Formulas For The Vector Space Method In Information Retrieval' by Erica Chisholm and Tamara G. Kolda. It also draws inspiration for semantic VSM design from the various literature in the area especially the paper titled 'A Systematic Study of Semantic Vector Space Model Parameters' by Douwe Kiela and Stephen Clark.

METHODOLOGY

An effort to overcome the problems that haven't been solved is made and the approach used is described below.

Vector Space Model

The Vector Space Model represents documents and queries as vectors in multidimensional space, whose dimensions are the terms used to build an index to represent the documents. The Vector Space Model procedure can be broadly divided into three stages. The **first stage** is the document indexing where content bearing terms are extracted from the document text. The **second stage** is the weighting of the indexed terms to enhance retrieval of document relevant to the user. The **last stage** ranks the document with respect to the query according to a similarity measure and retrieves the top 'n' ranked documents where 'n' is user-defined.

The major difference between the S*yntactic* and S*emantic* VSM lies in the Stage 1,where stemming is used in syntactic VSM and POS tagging and synonyms are used in the semantic VSM.

VMARK Version 1.0.0 : Syntactic VSM

The syntactic VSM is built using conventional term recognition, indexing and search optimisation techniques such as stemming using Porter's algorithm, standard inverted indexing and tf-idf weighting.

Stage 1:

A. Tokenization

Given a character sequence and a defined document unit, tokenization is the task of chopping it up into pieces, called tokens, perhaps at the same time throwing away certain characters, such as irrelevant punctuations.

B. Stemming

Stemming is the process in which a word is converted to its root based on a set of rules and the root is stored in the dictionary. A root word can represent many words that might have different meaning and root word may not carry any meaning. This helps in reducing the size of the dictionary as there is a possibility that a lot of words will be mapped to the same root.

C. Inverted Index

An inverted index (also referred to as postings file or inverted file) is an index data structure storing a mapping from content, such as words or numbers, to its locations in a database file, or in a document or a set of documents. The purpose of an inverted index is to allow fast full text searches, at a cost of increased processing when a document is added to the database.

VMARK Version 1.0.1 : Semantic VSM

The semantic VSM built uses the WordNet package to perform indexing based on part-of-speech (POS) tagging and synonyms based expansion in addition to the standard inverted indexing and tf-idf weighting.

Stage 1:

While the sub-stage A and C of stage 1 remain same for the semantic and syntactic VSM, the sub-stage B is replaced by POS Tagging and Synonym Expansion.

B. POS Tagging and Synonym Expansion

In the process of token processing, POS (Parts of Speech) tagging is done for each token and the synonyms of each token are also included in the inverted index. This helps in increasing the probability of finding a document of user interest even if he/she uses terms that are different from terms in the corpus.

Stage 2:

Term Weighting

It has been proved that proper term weighting can greatly improve the performance of the Vector Space Model[1].
The weighting scheme used composes of three different types of weighting, namely,

1) Local
2) Global
3) Normalization

The term weight is given by $L_{ij}\ G_i\ N_j$

Where,
L_{ij} is the local weight for term i in document j ,
G_i is the global weight for term i , and
N_j is the normalization factor for document j .

Local weights are functions of how many times each term appears in a document.
Global weights are functions of how many times each term appears in the entire collection.
Normalization factor compensates for discrepancies in the lengths of the documents.

After extensive literature review and analysis of the exiting techniques, the seemingly best techniques were chosen for calculating the **Global Weights** and **Local Weights.** The **document vectors** and **query vectors** are weighted using separate

schemes. The local weight is computed according to the terms in the given document or the query. The global weight, however, is based on the document collection regardless of whether we are weighting documents or queries. The normalization is done after the local and global weighting. Normalizing the query vectors is not necessary because it does not affect the relative order of the ranked document list.

DATA STRUCTURES

VMARK uses five data structures to connect the indexing, normalization and querying modules:

1. Inverted Index – invindex.txt
This data structure is used to store the inverted index which is one of the outputs of the indexing phase. For each term(after stemming/WordNet expansion) in the corpus, it stores the document ID if that term occurs in that document, frequency of that term in that document and the SQRT weighting for that term in that document. It is used by the querying module to retrieve the relevant documents.

2. List of Terms – terms.txt
This data structure is used to store the dictionary i.e. the list of terms in the corpus. It also stores the line number of the posting list in the inverted index file for that particular term and also the collection and document frequencies for that term. It is generated by the indexing module and is used the normalization and querying modules.

3. Document Term index – tf.txt
This data structure stores the list of terms in the document and their SQRT values for that document. It is an output of the indexing module and is used by the normalizing and querying modules to build the document vector.

4. Normalization Values – norm.txt
This data structure stores the normalization value for each document. It is generated by the normalization module and is used by the querying module to rank the documents by relevance.

5. Document Line Number – docline.txt
This data structure stores the starting line number in the corpus for each document. It is output by the indexing module and is used by the querying module to retrieve the relevant documents.

QUERY WEIGHTING

LOGG : Augmented Log

This is used to compute the local weight of the queries

$$\mathbf{L_{ij}} = \begin{cases} 0.2 + 0.8 \log(f_{ij} + 1) & \text{if } f_{ij} > 0 \\ 0 & \text{if } f_{ij} = 0 \end{cases}$$

ENPY : Entropy

This is used to compute the global weight of the queries

$$\mathbf{G_j} = 1 + \sum^{N} \frac{\frac{f_{ij}}{F_i} \log \frac{f_{ij}}{F_i}}{\log N}$$

DOCUMENT WEIGHTING

SQRT :Square Root

This is used to compute the local weight of the documents.

$$\mathbf{L_{ij}} = \begin{cases} \sqrt{f_{ij} - 0.5} + 1 & \text{if } f_{ij} > 0 \\ 0 & \text{if } f_{ij} = 0 \end{cases}$$

IGFL : Log Global Frequency IDF

This is used to compute the global weight of the documents.

$$\mathbf{G_j} = \log\left(\frac{F_i}{n_i} + 1\right)$$

where,
F_i is the frequency of term i throughout the entire collection.
f_{ij} is the frequency of term i in document j
N is the total Number of Documents
n_i is the number of documents in which the term has occurred

NORMALIZATION

COSN : Cosine Normalization

Cosine Normalization is the most familiar form of normalization in the vector space model is cosine normalization (COSN)

$$\mathbf{N_j} = \frac{1}{\sqrt{\sum_{i=0}^{m} (G_i L_{ij})^2}}$$

Where,
L_{ij} is the local weight for term i in document j ,
G_i is the global weight for term i , and
N_j is the normalization factor for document j .
m is the number of terms in that particular document.

The magnitude of the weighted document vector multiplied by the normalization factor forces the magnitude of the weighted document vectors to be one. This allows us to compare the angle between the weighted vectors without bias towards longer documents.

NOVELTY IN THE APPROACH

Novelty 1:

The approach we have chosen is totally novel because it uses LOGG, ENPY, SQRT, IGFL, COSN to calculate the term-weighting instead of the traditional tf-idf approach. Moreover, it has been proven that using the above techniques for calculating the Global and Local weighting of the documents are efficient compared to any other traditional techniques.[1]

Novelty 2:

We chose to implement semantic VSM to overcome the difficulties faced by syntactic VSM which may not give appropriate results when the query terms do not closely match the document terms. But, with the help of POS tagging and synonym expansion, we have managed to bring the semantics of the query into the picture thereby expanding the scope of the VMARK search engine. WordNet package is used for the purpose of POS tagging and synonym expansion.

ANALYSIS AND RESULTS

Queries that are relevant to the corpus are given to VMARK and the precision results are as follows.It is found that the Syntactic VSM gives better results when the query terms closely match the terms in the corpus.The average precsion for the Syntactic VSM VMARK Version 1.0.0 is calculated as **0.76.**The average precision for the Semantic VSM VMARK Version 1.0.1 is found to be **0.59**. This is becuase the queries terms almost match the terms in the document.For the queries where the terms in the query do not match the terms in the document or if there is a huge gap between the vocabulary of the formulated query and the document,Semantic VSM was giving better results.

Query	Precision Values for each Version	
	1.0.1	1.0.0
Crispy chips	0.8	1
Cream and onion chips	0.7	1
Chicken Nuggets	0.4	0.6
Fresh and Tasty	0.8	1
Cheese and cream	0.6	0.6
Butter and bread	0.6	1
Dog food	1.0	1
Oily and junk	0.6	0.8
Edible color	0.5	0.6
Food is tasty	0.0	1
In love with chicken	0.6	1
Amazon is awesome	0.6	0.8
Amazon delivery sucks	0.3	0.4
Delicious Tuna	0.6	0.2
Scrumptious Bacon	0.8	0.4

CONCLUSION

The goal of Information Retrieval is to make it easier for the user to obtain data relevant to a given query in minimum time and cost.We have tried to achieve this through the different versions of VMARK. The success or failure of the Vector Space Model largely depends on the term weighting schemes. Based on the literature review we have done, we used the techniques that were proven to be efficient to compute the Local weights, Global weights, Normalisation values and hence the term weighting.

Moreover the semantic VSM is more promising when the user space is huge because the vocabulary varies from user to user in formulating a query and a mere syntactic search may not yield accurate results. VMARK Version 1.0.0 (Syntactic VSM) and VMARK Version 1.0.1 (Semantic VSM) retrieve the top ‘n’ documents that are relevant to the query where ‘n’ is user defined and almost all the documents retrieved are found to be relevant to the queries.

REFERENCES

[1] New Term Weighting Formulas for the Vector Space Method in Information Retrieval;
Erica Chisholm and Tamara G. Kolda
[2] Introduction to Information Retrieval. Christopher D. Manning; Prabhakar Raghavan. Hinrich Schütze. Cambridge University Press
[3] A Systematic Study of Semantic Vector Space Model Parameters by Douwe Kiela and Stephen Clark