Effects of VSM, LSI, RP and SOM on Information Retrieval

Santosh Tirunagari and Jayasimha Ramachandra Rao
santosh.tirunagari@aalto.fi, 245577
jayasimha.ramachandra.rao@aalto.fi, 245234

May 31, 2011

Abstract

The main aim of this project is to apply Salton's vector space model (VSM), and compare other methods suggested like Latent semantic indexing and Random projections on the document corpus and query corpus and to compare the the suitability of the methods to the task. This report also discusses the advantages and disadvantages of each method. In this report its shown that the highest similarity among the queries with the documents can be retrieved by using random projection and then followed by VSM , SOM and LSI.

Contents

0.1 Introduction 4

0.2 Data and Methods 4

0.2.1 Data 4

0.2.2 Vector Space Model (VSM) 5

0.2.3 Latent Semantic Indexing (LSI) 5

0.2.4 Random Projections (RP) 6

0.2.5 Self Organising Maps (SOM) 7

0.2.6 Cosine similarity 7

0.2.7 Term and Document Frequencies 8

0.3 Experiments 8

0.4 results 8

0.5 conclusion 9

0.6 contributions 9

0.7 References 10

List of Figures

1 Singular Value Decomposition 6

2 similarity comparision among LSI, VSM and RP and SOM . 10

List of Tables

1 distrubution of documents in categories 5

2 No of dimensions selected per method 9

0.1 Introduction

Information retrieval is the area of study dealing with retrieving interesting information from the large collections of data. It has its applications in text, image and multimedia applications. The main components of this project include: 1) A document collection set and queries, 2) Pre-processing approaches, parsing the documents sets and query sets which can be shown in experiments section 3) Representation of both documents and queries (dependent on information retrieval model chosen), 4) Comparison algorithm, cosine distance can be used which is popular to get the documents which are relevant to queries 5) Feedback module.

This report mainly concentrates on comparison of methods like LSI, VSM, RP and SOM with which the highest similarity is between query and document vectors. The importance of considering similarity measures becomes particularly important in the case of very sparse high-dimensional data, related to the curse of dimensionality. It has been shown that this phenomenon has an impact on various techniques for classification (including k-NN classifiers) and clustering [4]. It also effects information retrieval results [5].

Hence in our experiments apply LSI, SOM and RP as the dimensionality reduction methods analyze the performance of the cosine similarity metrics with each of them.

The next part of the report consists of the methods used in the project and experiments done, results and conclusion.

0.2 Data and Methods

This section describes the dataset and methods that have been applied in this term project.

0.2.1 Data

The document collection used in the experiment is cacm.all. There are 3204 documents in the document set and 64 queries in the query set.

Type	Number
Documents	3204
queries	64

Table 1: distrubution of documents in categories

0.2.2 Vector Space Model (VSM)

The VSM is one of the models in the information retrieval which is quite famous. In this technique the documents are modeled as the sets of keyword terms that can be individually weighted and also the queries are modeled with the same terms and perform queries by comparing the query model with each of the document representation in the space. the comparison is based on some similarity measure like cosine which is pretty good for the document mining. The queries doesnot require to have one of the terms in the document model.

Documents and queries are represented as vectors.

$d_j = (w1, j, w2, j, ..., wt, j)$

$q = (w1, q, w2, q, ..., wt, q)$

Each dimension corresponds to a separate term. If a term occurs in the document, its value in the vector is non-zero

Advantages of the VSM are : improved performance over the Boolean model due to weighting schemes and partial matching allowed which gives a natural ranking

Disadvantages of the Vector Space Model are terms are considered to be mutually independent.

0.2.3 Latent Semantic Indexing (LSI)

Using SVD for term-document matrices is better known in the IR community as **Latent semantic indexing** (LSI). The Latent Semantic Indexing information retrieval model builds upon the prior research in information retrieval and, using the singular value decomposition (SVD) [1] to reduce the dimensions of the term-document space. LSI explicitly represents terms and documents in a rich, high-dimensional space, allowing the underlying (latent), semantic relationships between terms and documents to be exploited during searching.

Singular Value Decomposition (SVD) of the data matrix $\mathbf{X}$ is defined as

$$\mathbf{X} = \mathbf{V}\mathbf{\Sigma}\mathbf{U}^\mathsf{T} \tag{1}$$

where $\mathbf{V} \in \mathbf{R}^{n \times n}$ and $\mathbf{U} \in \mathbf{R}^{d \times d}$ are orthonormal matrices, containing the left and right singular vectors of $\mathbf{X}$, respectively, and $\mathbf{\Sigma} \in \mathbf{R}^{n \times d}$ is a rectangular diagonal matrix containing the singular values of $\mathbf{X}$ on its main diagonal. In the same manner as with PCA, we can select the D left singular vectors corresponding to the largest singular values and perform the dimension reduction as

$$\mathbf{X}_{\text{SVD}} = \mathbf{V}^\mathsf{T}\mathbf{X} \tag{2}$$

It is worth noting that if the data vectors are centered, i. e. $\text{E}(\mathbf{X}) = 0$, then PCA and SVD produce the same projection.

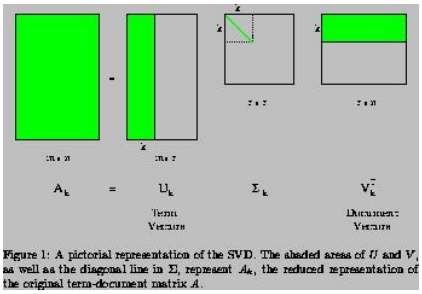

Figure 1: Singular Value Decomposition

some of the pros and cons of LSI are, Attempts to overcome term independance assumption of the classical vector space model. Approach involves mapping the term document matrix to a reduced space using singular value decomposition. The resulting matrix is then used for in comparisons to queries. Better performance obtained but difficulty exists in determining the optimal reduction in matrix size.

0.2.4 Random Projections (RP)

The random projection method is designed to approximate the cosine distance between vectors. The basic idea of this technique is to choose a random hyperplane (defined by a normal unit vector) at the outset and use the hyperplane to hash input vectors.

Random projection method is a heuristic and it also gives better results.

0.2.5 Self Organising Maps (SOM)

SOM is an unsupervised, competitive learning, clustering network. In this, only one neuron or prototype vector is active or is the winner at a particular point of time. The prototype which is closest to a document vector is moved closer by a learning parameter. and then the neighbours are moved. SOM can be used for dimensionality reduction as no of prototype vectors is less that no of input document vectors.

0.2.6 Cosine similarity

In the implementation of the comparison algorithm, the similarity or the distance measures must be predetermined. This measure shows the degree of closeness or separation of the data objects [4] . The performance of the compariosn algorithm depends on choosing a good similarity metric over the input data. Universally there is no solid similarity metric. But cosine is chosen most. Hence we also chose cosine measure.

Not all distance measures is a metric, a measure d is said to be a distance metric [1] if it satisfies all the four properties namely non-negativity, isolation, symmetry and triangular-inequality.

- non-negativity: The distance between any two points must be non negative. $d(i, j) \geq 0$
- isolation: The distance between two points is zero if and only if the points are identical. $d(i, j) = 0$ iff $x = y$.
- symmetry: The distance between the points . i and j must be equal to the distance between j and i . that is $d(i, j) = d(j, i)$
- triangular-Inequality: Sum of distances from two distinct points must be greater than the distance from third point. That is $d(i, j) \leq d(i, h) + d(h, j)$

similarity can be calculated as one minus distance.

Relevance rankings of documents in a keyword search can be calculated, using the assumptions of document similarities theory, by comparing the deviation of angles between each document vector and the original query vector where the query is represented as same kind of vector as the documents. In practice,

[1] `http://kochanski.org/gpk`

it is easier to calculate the cosine of the angle between the vectors, instead of the angle itself: A cosine value of zero means that the query and document vector are orthogonal and have no match (i.e. the query term does not exist in the document being considered).

cosine similarity can be calculated as follows $d_{ij} = \left(\frac{x_i.x_j}{|x_i|\times|x_j|}\right)$

0.2.7 Term and Document Frequencies

Term Frequency (TF) is the total count of the particular word repeated in a document and is calculated as $tf_{ij} = \frac{n_{i,j}}{\sum\limits_{k} n_{k,j}}$ where $n_{i,j}$ is the number of times the term t_i occurs in document d_j and the denominator is the sum of number of times all terms occur in document d_j [5].

0.3 Experiments

As the preprocessing step, the document collection is parsed to extract .T and .W fields from the document collection and .W field from the query collection. In the next step puntuation symbols are removed and the text is converted to lower case. we proceed with removing the stopwords. After this process all the unique terms are collected. document term maps is generated and weight normalized based on the unique terms for both document collection and the query collection.

Each row represents a document vector hence the total documents in the document collectoin are 3204 and query collection are 64. for every query vector the cosine similarity is calculated with every document vector in the document collection, and the first best documents are retieved based on the highest similarity.

In the experiment we used random projections, svd (matlab function) and som (somtoolbox from hut) to reduce the dimensionality of the data and donducted the experiments for them aswell.

0.4 results

Red represents RP, Green - VSM, Yellow-SOM and Cyan - LSI.

Method	Dimensions selected
VSM	429
SOM	200
LSI	199
RP	199

Table 2: No of dimensions selected per method

The results show that the highest similarities can be achieved through random projects, followed by VSM, SOM and LSI. we conducted experiments on taking different dimensions of documentmap using svd. but the similarities achieved is the same.

0.5 conclusion

The results show that the highest similarities can be achieved through random projects, followed by VSM, SOM and LSI. we conducted experiments on taking different dimensions of documentmap using svd. but the similarities achieved is the same.

In future this work can be carried out by actually calculating the precision and recall values using these methods and comparing how best is the similarity maching criteria suits the practical document retrieval.

0.6 contributions

Both of them tried to analyse the contents of the course on information retrieval and have found the work very interesting. we divided the tasks equally to each other in both report writing and conducting experiments aswell. This project is the combined effort of us. During this course we learnt new techniques like random projections, hence wanted to use it in the experiments and found its very good in small applications both computationally and efficiently aswell, though it is heuristic. We thank the course co-ordinator and lecturer for making this course a well interactive session.

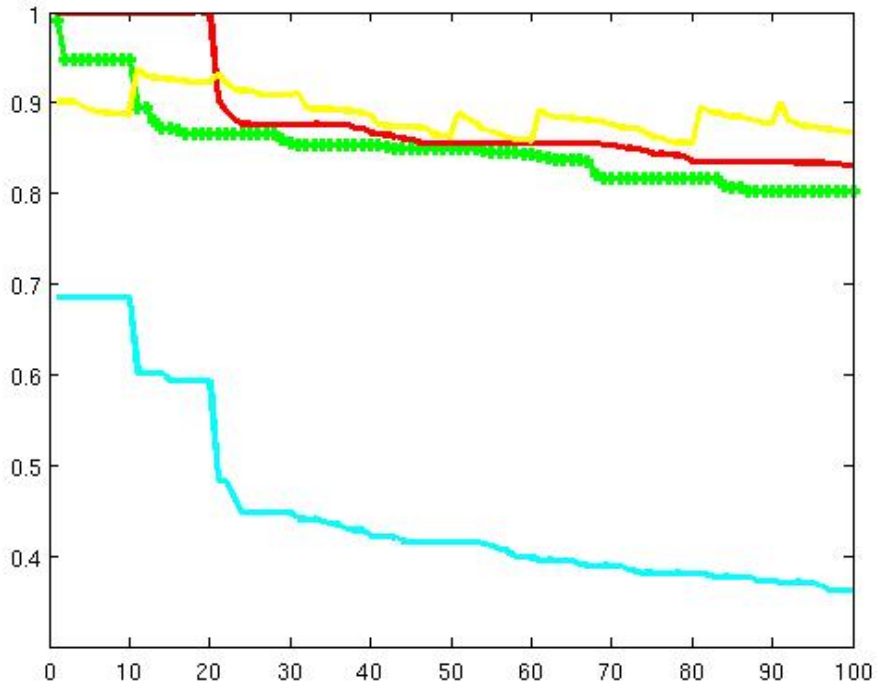

Figure 2: similarity comparision among LSI, VSM and RP and SOM

0.7 References

[1] G. Golub and C. Van Loan. Matrix Computations. Johns-Hopkins, Baltimore, Maryland, second edition, 1989.

[2] S. Dumais. Improving the retrieval of information from external sources. Behavior Research Methods, Instruments, & Computers, 23(2):229–236, 1991.

[3] Radovanović, Miloš; Nanopoulos, Alexandros; Ivanović, Mirjana (2010), "Hubs in space: Popular nearest neighbors in high-dimensional data", Journal of Machine Learning Research 11: 2487–2531

[4] Radovanović, Miloš; Nanopoulos, Alexandros; Ivanović, Mirjana (2010), "On the existence of obstinate results in vector space models", Proceedings of the 33rd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval: 186–193.

[5] Taghva Kazem and Veni Rushikesh, Effects of Similarity Metrics on Document Clustering, Proceedings of the 2010 Seventh International Conference on Information Technology: New Generations, IEEE Computer Society 2010.

Appendix

```
% code written by Santosh Tirunagari
% retrieves top 10 matching documents with the similarities.

function [qr,in] = proj(map,querymap,qrels)
% calculate similarity
t = 1;
[m,n] = size(map);

for i = 1:1:64
   for j = 1:1:3204

       if(sum(map(j,1:n-1))> 0 )
       temp(t,1) = 1-pdist2(querymap(i,1:n-1),map(j,1:n-1),'cosine');
      if(temp)
          qdid(t,:) = [querymap(i,n) map(j,n) temp(t,1)];
          t = t+1;
      end
       end
   end
end

sortedA = sortrows(qdid,1);  %# Sort the rows by the first column
 [~,~,uniqueIndex] = unique(sortedA(:,1));  %# Find indices of unique values
                                    %#   in the first column
 cellA = mat2cell(sortedA,...                    %# Break matrix up by rows
                 accumarray(uniqueIndex(:),1),3);  %#   into a cell array

 % sort rows and find max similarity documents
for i= 1:1:64
cellA{i,1} = sortrows(cellA{i,1},-3);
for j = 1:1:64
   queryrel{j,:} = (cellA{j,1}(1:10,:));
end
end

qr = cell2mat(queryrel);

% code for splitting data set to parse .T and .W fields
% http://www.dataminingresearch.com/?s=awk&x=0&y=0

BEGIN { docNo = 0 }
$0 == ".T" { docNo++  #starting a new doc when we encounter a .T line
                              textStarted = 1
                              printThisLine = 0
                              print "Processing Doc #" docNo
                              docName = sprintf("%s%d", "cacm.", docNo) # doc name is like cisi.1 cisi.2 etc.
                             }
```

```
$0 == ".A" || $0 == ".X" || $0 == ".I" || $0 == ".B" { textStarted = 0 }  # when a new field separator
encountered stop printing until next .T or .W
$0 == ".W" { textStarted = 1  # body part starts with .W
                        printThisLine = 0
                        }
        { if (textStarted == 1)
                {
                        if (printThisLine)
                                print $0 > docName

                        printThisLine = 1 #consequent lines after .T or .W will be printed
                }
        }

 %% plot graph

load qrvsm.mat
load qrrp.mat
load qrsvd.mat
load qrsom.mat

qrrp = sortrows(qrrp,-3);
qrvsm = sortrows(qrvsm,-3);
qrsvd = sortrows(qrsvd,-3);

plot(qrrp(1:100,3), 'r','linewidth',3);
hold on;

plot(qrvsm(1:100,3), 'g','linewidth',3);
hold on;
plot(qrsvd(1:100,3), 'c','linewidth',3);
hold on;


plot(qrsom(1:100,3), 'y','linewidth',3);

%  som code  by Jaya Simha
% LSI and RP by Santosh Tirunagari

[u v z] = svd(map(:,1:429));
svd_map = map(:,1:429)*z(:,1:199);
docids = map(:,430);
svd_map = [svd_map docids];
save svd_map.dat svd_map -ascii

[u v z] = svd(querymap(:,1:429));
svd_qmap = querymap(:,1:429)*z(:,1:199);
qids = querymap(:,430);
svd_qmap = [svd_qmap qids];
```

```
save svd_qmap.dat svd_qmap -ascii

rp = randn(199);
rp = zscore(rp);
rpmap = map(:,1:199) * rp; % project the data on some random space to remove orthogonality.

qrp = randn(199);
qrp = zscore(rp);
qrpmap = querymap(:,1:199) * qrp;


docids = map(:,430);
rpmap = [rpmap docids];

qids = querymap(:,430);
qrpqmap = [qrpmap qids];


save rp_map.dat rpmap -ascii
save rp_qmap.dat qrpqmap -ascii


sMap=som_randinit(map(:,1:429)' ,'msize',[20 10]);
sMap=som_seqtrain(sMap,map(:,1:429)');
som_map = (sMap.codebook');
docids = map(:,430);
som_map = [som_map docids];
save som_map.dat som_map -ascii

sMap1=som_randinit(querymap(:,1:429)' ,'msize',[20 10]);
sMap1=som_seqtrain(sMap,querymap(:,1:429)');
som_qmap = (sMap1.codebook');
qids = querymap(:,430);
som_qmap = [som_qmap qids];
```