A Comparison of Information Retrieval Models

Mandeep Pannu
Kwantlen polytechnic university
Faculty of CSIT
12666 – 72 Ave
Surrey, B.C. CANADA V3W 2M8
mandeep.pannu@kpu.ca

Anne James
Coventry University
Faculty of Engineering & Computing
3 Gulson Road, CV1 2JH
Coventry, United Kingdom
a.james@coventry.ac.uk

Robert Bird
Coventry University
Faculty of Engineering & Computing
3 Gulson Road, CV1 2JH
Coventry, United Kingdom
Robert.bird@coventry.ac.uk

ABSTRACT
The personalization process can either be focused on individuals and their interaction with documents, or on the identification of shared patterns of behavior and the segmentation of the user population into groups of common interests. One issue in the personalization and the filtering processes is the selection of an appropriate model for the efficient representation and manipulation of user profiles and documents. It should be capable of facilitating the determination of relevant documents in terms of similarity between users and documents. The aim of this paper is to introduce three models for representing documents and profiles in the search process, and to examine their computational processes. The volume of document databases, the large number of users and their different interests creates the need for precise and efficient filtering techniques. This paper investigates different information retrieval models, which can be used to determine the similarity between documents and user profiles.

Categories and Subject Descriptors
H.3.3 [Information Search and Retrieval]: Information Models, Information filtering

General Terms: Personalization and user profiles.

Keywords
Information Retrieval, Boolean Model, Vector Space Model, Probabilistic Model.

1. INTRODUCTION
Information models are significant because they are representative of three different mathematical models, with their own methods for representing documents and calculating similarity between documents and users' profiles [4]. It will focus on three models: the Boolean Model, the Vector Space Model, and the Probabilistic Model. The Boolean Model is an instance of the set-theoretic models, where documents are represented as sets of words, on which operations are performed in order to determine similarities. The Vector Space Model is an algebraic model in which documents and users' profiles are represented as vectors. Operations, such as the dot product of two vectors, are used to determine similarities as a scalar values. Finally, in the Probabilistic Model probabilistic inference is used to retrieve documents. This model relies on probabilistic theorems, such as Bayes' theorem, to compute similarities as probabilities of relevance. Three models supporting information retrieval were covered, with a particular emphasis on their mode of representation of the documents and their processing algorithms.

WCCCE'14, May 2-3, 2014, Richmond, British Columbia, Canada.

http://dx.doi.org/10.1145/2597959.2597978

2. DOCUMENT REPRESENTATION AND PROCESSING
The filtering or retrieval process requires a specific representation of web documents and user profiles.

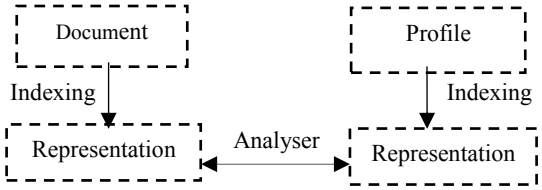

Figure 1.1: Information Retrieval

Figure 1.1 illustrates the basics of an information retrieval process. The analysis is based on simplified representation of documents created during the indexing process and on the representation of the targeted user profile. The analysis algorithm calculates the similarity based on these representations and determines how well each of the documents satisfies the user information requirements (how similar it is to the user profile). As a simplified representation can be less precise and more ambiguous that the original document (or profile), the search results can be less accurate than if a full original document had been compared with full profile, however the computational and storage requirements for such comparison would be higher.

As web a document can be complex, it is required that its content is represented in a form that can be analysed efficiently. The exact representation of the same document can vary from system to system, however in general there is an indexing process that actually converts documents into a simplified form. The basic simplified form of a document can be, for example, a list containing all the distinct keywords used within the document. In a more advanced system, it can be a vector containing keywords-value pairs, where the value can be for instance the number of times a word occurs in the document or the distance between the first occurrence of that word and the start of the document. If the document representation contains some additional rating values (like number of occurrences or position in the document), then a system that is analysing the similarity can be more advanced and has the possibility to provide more accurate results. The indexer used by Google is storing the information about the position of keywords and the distance between them as well as the kind of

HTML tag that is used to enclose it; for example, whether it is H1 tag, which is used for titles or section names or H2 tag which is rather used for subsections and therefore can be considered as less important [5]. There is no reason why a system could not use more than one technique for storing the representation of documents, one basic presentation for easily filtering out most of the documents and a detailed one, for predicting the relevance of the remaining documents with a higher accuracy.

3. BOOLEAN INFORMATION RETRIEVAL (BIR)

The Boolean Information Retrieval model is based on classical Set theory. Documents are represented as a set of terms it contains (not all words have to be used), while queries are represented as logical expressions. The keywords in the query can be linked together with Boolean operators AND, OR and NOT [8]. Each term can have one of two logic states – it can be either present (logical 1) or absent (logical 0) [8].

The relevance of a document to the query of a user is calculated by evaluating the logical value of the query as either 1 or 0. A value of 1 is given to every term in the query that exists in the set representing document, and 0 for every term that does not exist in the representation of the document.

3.1 Document representation in BIR

For the purpose of Boolean Information Retrieval each document in the database has to be presented as set of terms. In order to limit the size of each representation, not all words have to be stored. Instead a dictionary (set) of interesting words is created. Depending on the purpose of the database the dictionary can be small and contains only words for one specific domain or large, containing e.g. all nouns. During the indexing process, each document is compared to the set of interesting terms to create the vector representation. If the terms dictionary is created as a vector containing words, then each document can be represented by a vector of ones and zeros. The vector size should be the same as the size of the dictionary vector and for every word in the dictionary; if it is relevant to the document then the document representation vector will contain 1 on the same position as the word otherwise it will contain 0 for that position.

	Term 1	**Term 2**	**Term 3**	**…**	**Term M**
Dictionary	KPU	University	Course	…	Library
Doc. 1	1	0	0	…	1
Doc. 2	1	1	0	…	0
Doc. 3	1	1	0	…	1
…	…	…	…	…	…
Doc. N	0	0	0	…	1

Figure 1.2: Example of documents representation for BIR

In the example in Figure 1.2 the first document can be related to a library in KPU but it is most likely not the university library because the term ‘university’ does not occur in it. Document 2 can be related to KPU University but not to the library while the third document is related to ‘KPU’, ‘University’ and ‘Library’.

The exact method of storing the documents representations can vary from system to system, but the Boolean Information Retrieval requires a method to verify whether a term is relevant to a document or not (e.g. whether it occurs in the document or – for possible implementation – whether synonym of the word occurs in the document).

3.2 Query representation in BIR

The user query is a logical statement whose value has to be evaluated for each of the documents in the database in order to filter the relevant documents. Each keyword in the query is a single word or conjunctions of words.

Queries are specified as Boolean expressions and terms combined with operators. For example, a query that should return all documents that contains Term1 and documents that contain Term2 but not Term3 can be expressed as:

Query1 = Term1 OR (Term2 AND NOT Term3)

3.3 Determination of document relevance in BIR

In order to determinate the relevance of a document to the query, the logical value of the query has to be evaluated. Each term in the query has a logical value 1 if the word exists in the document (or its representation) and logical value 0 if it does not. After all terms in the query are replaced by logical values, the query can be evaluated as any logic sentence. If the sentence is true then the document is considered relevant.

In the example the dictionary has five terms: ‘KPU’, ‘University’, ‘Course’, ‘Cost’, and ‘Library’.

If a user wants to find the cost of the course and information about the university library, the following query can be used:

Query = KPU AND University AND ((Course AND Cost) OR Library)

This, after replacing words with values from the terms in each document will produce following sentences:

Document 1 = 1 AND 0 AND ((0 AND 0) OR 1) = 0

Document 2 = 1 AND 1 AND ((1 AND 1) OR 0) = 1

Document 3 = 1 AND 1 AND ((0 AND 0) OR 1) = 1

Document 4 = 1 AND 0 AND ((0 AND 1) OR 0) = 0

Document 5 = 0 AND 1 AND ((1 AND 0) OR 1) = 0

The logical value of query is 1 for Document 2 and Document 3 therefore these two documents would be returned.

4. VECTOR SPACE MODEL (VSM)

Vector Space Model (VSM) is an algebraic model used for information filtering, information retrieval, indexing and relevance ranking [2]. The Vector Space Model is a way of representing and comparing documents and queries based on words (keywords) with values [2]. This model can be used to rank the similarity between documents – not just to answer if document contains required words or not. Each component of a vector represents one term/keyword, and has a value. The value is a real number that indicates how relevant a term is to the document or query being described [2]. VSM processing can be divided into two stages: Document Indexing with Term Weighting and Documents Relevancy Ranking.

4.1 Document Indexing

The first stage of information retrieval is document indexing. Each indexed document is represented as a vector of terms

contained by the document and weights of each term. Weight of a term describes how important that term is to the document, e.g. terms from documents' title will be more important than terms from the footer. The process of creating the vector includes stop words removal and stemming. Stop words like 'of', 'an', 'the', and etc are removed as there are not relevant to the document abstract [10]. Words suffixes – like 'ed', 'ion', 'ing', 'ions' can be removed to avoid recording different variants of a single word.

The indexing process can cover an entire document or only part of a document. Some systems for example only index words from the document title and the abstract only, while other index the entire document and then modify the relevance value of each term depending on the term position in the document.

Every term has to be evaluated to estimate its importance in the document. In the basic implementation the rating can be set according to the number of times that a term occurs in a document. In general, VSM relies on two main factors for term weighting: Term Frequency vector (TF), and Inverse Document Frequency vector (IDF) [1]. In a term frequency vector created for a document, the rating of a term depends on the number of occurrences of that term in the document. However, some words are very common (e.g. 'a', 'the', 'in') and therefore the rating for these terms would be very high – even if they are not important to the content of the document. To overcome this problem, an Inverse Document Frequency vector (IDF) is created. This vector stores the general importance of every term, in respect to all documents. It is generated by calculating the number of documents that contains each term. The weight for each term in the IDF vector is higher in the term is less popular and lower if it is more popular. The weight value for each term is calculated as the logarithm from the quotient of the total number of indexed documents divided by the number of documents in which the term appears. Once the term frequency vector for each document is created, and one inverse document frequency vector for all documents is ready, then the final document representation is created. The weight for each term in the vector representing each document is calculated by multiplying the weight from the term frequency vector for that document, with the rating of the term in the inverse document frequency vector. If that value would be used to calculate the relevance to a query then long documents would usually be considered more relevant, because each term can occur more times in a longer document. To resolve this issue, the generated is normalised, by dividing weight of each term in the vector by the vector length. The length of a vector is calculated as the square root from sum of squares of all weights in the vector. As the result of normalisation, the length of vectors for all documents is equal to one, and the length of each document does not affect the retrieval process [10].

4.2 Determination of document relevance in VSM

Once the documents are indexed, a search system can rank and order the documents according to the calculated similarity to a query. The query is represented in the same fashion as the documents – by term vector with ratings for each stored term – except that the normalization of the vector is not essential.

The similarity between a single document and the query is calculated as a cosine similarity between two vectors. If the two vectors are displayed in the N dimensional Cartesian coordinate system (where N is the total number of terms in both vector, and each axis is representing the value of one term) then the cosine similarity would be equal to the cosine of the angle between the two vectors.

To calculate the cosine similarity, the weight of each term from one of the vectors is multiplied with the weight of the same term from other vector (zero weight is assumed if term does not exists), and then all values have to be summarised. Finally that value should be divided by the length of the first vector and by the length of the second vector.

$$Sim\ (D,P) = \frac{D \cdot P}{\|D\|\ \|P\|} = \frac{\sum_{i=1}^{m} d_i p}{\sqrt{\sum_{i=1}^{m} d_i^2}\ \sqrt{\sum_{i=1}^{m} p_i^2}}$$

Where:

D – term vector for document

P – term vector for the query

As term vector for documents is normalized during the indexing, its length can be omitted as it is equal to 1 for all documents. The same applies to the query term vector – it can be normalized once.

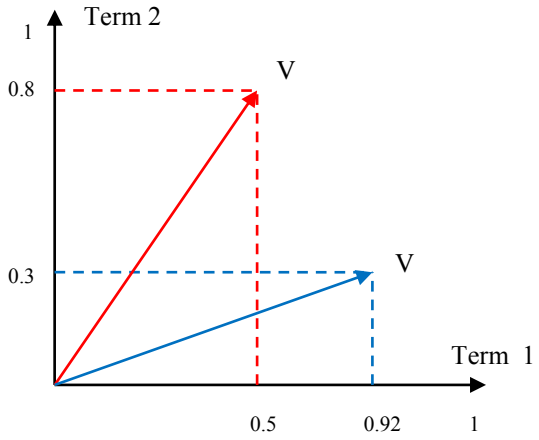

Figure 1.3: An example of two normalized vectors

The figure 1.3 shows an example of two normalized vectors and the cosine similarity between vectors V_1 and V_2 is calculated below.

$$V_1 = [0.53, 0.85]$$

$$V_2 = [0.92, 0.39]$$

$$simil = \cos(\alpha) = V_1 \cdot V_2 = [0.53, 0.85] \begin{bmatrix} 0.92 \\ 0.39 \end{bmatrix}$$
$$= [0.53 \cdot 0.85 + 0.92 \cdot 0.39] = 8.81$$

4.3 Example of VSM application

The example will consider four documents, and one query.

Documents:

Document	*Content*
Doc.1	KPU university Computer
Doc. 2	SFU university arts department
Doc. 3	SFU University Computer Science department
Doc. 4	KPU Computer department

Query:

Term:	KPU	University	Computer	Science
Importance:	1	1	0.5	0.3

During the indexing, all the terms were extracted from the documents to create representation of each of the document. In the process an Inverse Document Frequency vector has to be generated. To generate IDF vector the indexer first has to create Document Frequency vector (DF) that for every term counts the number of documents that contains the term. Subsequently, the total number of documents is divided by the number of document that contains a specific term, and the logarithm of that value is stored in the Inverse Document Frequency vector (IDF) for that term.

All generated vectors should be normalized to eliminate the advantage given to the longer documents, as even if a term is repeated multiple times in longer documents, it should not be considered relevant to that document if it is flooded by other terms. The normalization of a vector is simply a process of dividing weights of each term stored in that vector by the length of that vector.
After the indexing process is completed the system is ready to generate responses to queries. In order to retrieve the search results for a specific query, a similarity between the user query and each of the documents has to be calculated.

5. PROBABILISTIC INFORMATION RETRIEVAL (PIR)

Both Boolean Model and Vector Space Model provide similarity ratings without considering a level of certainty for the output relevance. There are several models based on probability theory that aim to determine the probability of a document being relevant to a query [8].

The Probabilistic retrieval was first proposed by Manor and Kuhns in 1960, [11] and many variants of the Probabilistic Model have been proposed. The user information is represents by a set of possibly weighted keywords given by the users or induced by the system.

5.1 Probabilistic Information Retrieval principles

The results retrieved by probabilistic information retrieval systems depend on estimations and probabilities. The first assumption is that terms are dispersed differently between relevant and non-relevant documents [3]. A PIR system ranks documents and sorts them in decreasing order of probability of relevance to the information need once the probability is calculated [3]. The results are as accurate as the calculated probability [9].

The classic probabilistic model returns documents in decreasing order of calculated probability of relevance to the information requirement. After the indexing process every term can have assigned a value that indicates the probability that a document containing this term is relevant to the concept described by the term. In the retrieval phase the documents have calculated a value which is the sum of probabilities from terms that exists in both a document and in the query. The documents are then retrieved in order according to this value (descending). The document representation for this version of Probabilistic Information Retrieval could be the same as in the Boolean model as it only need to store information if either document contains a term or not [9].

Similarly to the Inverse Document Vector in the VSM model, a vector has to be created that stores information about how important each term is. If ‘p’ is the probability that a document which contains a term and it is relevant to the query and ‘q’ is probability that the document contains the term but it is not relevant, then the weight of the term is calculated as:

$$w_i = log \frac{p_i(1-q_i)}{q_i(1-p_i)}$$

Where:

$$p = \frac{number\ of\ relevant\ documents\ containing\ term}{total\ number\ relevant\ of\ documents}$$

$$q = \frac{number\ of\ irrerevant\ documents\ containing\ term}{total\ number\ of\ not\ relevant\ documents}$$

If

n_i = Number of documents containing term i

r_i = Number of relevant documents containing term i

N = Total number of documents

R = Number of relevant document

Then p and q can be expressed as

$$p_i = \frac{r_i}{R}$$

$$q_i = \frac{n_i - r_i}{N - R}$$

Terms	D1	D2	D3	D4	W(term weight)
KPU	1	0	1	1	-1.322
University	1	1	1	0	0.26
Science	0	0	1	0	-0.26
Engineering	1	0	0	0	-0.26
SFU	0	1	0	0	1.32
Business	0	1	0	1	0.48
Department	0	1	1	1	0.26
Computing	0	0	1	0	-0.26

And w_i can be expressed as

$$w_i = \log \frac{p_i(1-q_i)}{q_i(1-p_i)} = \log \frac{r_i(N-R-n_i+r_i)}{(n_i-r_i)(R-r_i)}$$

As the number of relevant documents is unknown, some assumptions have to be made. Usually it is assumed that the probability p is constant (e.g. equal to 0.5), and that q can be estimated by the values from Inverse Document Frequency vector, created as in the Vector Space Model [8].

With the assumption that 50% of the documents containing a term are relevant, the number of relevant document containing the term and the number of irrelevant document containing the term will be equal and their sum will be zero in the denominator. To avoid infinity values when R-r = 0 or n-r=0, 0.5 can be added to each component as follows:

$$w_i = \log \frac{(r_i+0.5)(N-R-n_i+r_i+0.5)}{(R-r_i+0.5)(n_i-r_i+0.5)}$$

The equation indicates how to calculate the probability that a document containing a specific term is relevant to a query with that term. It also considered as the weight of the term.

This model is based on uncertain guess of whether a document has relevant content to match the query and the document representation. The Probabilistic Retrieval Model uses the estimation method to retrieve the results based on assumptions that are made explicitly. Relevance feedback can improve the ranking by providing better assumptions.

5.2 Probabilistic Retrieval Example

The documents set contain four documents with following contents:

Document	*Content*
Doc.1	KPU university Business
Doc. 2	SFU university arts department
Doc. 3	SFU University Computer Science department
Doc. 4	KPU Business department

And the user query is: 'KPU university computing science'.

The indexed documents have been presented in the Figure 1.4. The terms are extracted from the documents. Term weight is applied to each term. The weight of each term is calculated and it can be used in the same way as the IDF vector in the Vector Space Model.

Figure 1.4: Example of documents representation for PIR

After term weights are calculated, the relevance values can be calculated for each of the documents.

6. ADVANTAGES AND DRAWBACKS OF INFORMATION RETRIEVAL MODELS

The Boolean retrieval model enables users to formulate complex logical statements. However, the construction of Boolean queries can be difficult for an average user, and all the terms entered in a query are considered equally important. Due to the binary nature of the results the model does not provide a ranking of retrieved documents, only a set of retrieved document – without any particular order. Also, because an exact matching criterion is used the returned set of documents will be either almost empty (which is a low recall as many relevant document would not be retrieved) or will include many documents (therefore precision would be low as irrelevant document would be also in the set). An example of exact match query is science AND computer. In Boolean terms, the document has to contain both 'science' and 'computer' to satisfy the query. It means if one term is missing, it will not be considered relevant at all, while if it contains both terms it will be considered fully relevant [12].. As all terms are equally weighted, this model is more useful for data retrieval than information retrieval [13].

To eliminate the problem with different variants of the same words, each word in both dictionary and document can be represented without suffix. Also process of dictionary creation can be altered by representing synonymous as a single word in order to decrease the size of the database, and to eliminate the problem of exact words matching. During the dictionary creation, if a word has already a synonym in the dictionary then it does not have to be added to it. This requires that when a document is being indexed and a new word is detected in it, then any synonym of that word existing in the dictionary will be considered as existing in the document. This approach can decrease the database size and eliminate the problem with checking for exact match only. However the precision of retrieval can decrease with these optimisations.

In contrast with the Boolean Retrieval Model, in the VSM a range of values can applied to each term – both in documents representations and in the user query. In addition, the normalization of the vectors long documents are not favored over short ones, and because of the use of inverse document frequency vector, popular terms are not considered important while rare terms are promoted. In the Boolean Model a page containing a full dictionary would be considered relevant to most of the queries, unless they contain the NOT operator; in the VSM model each term on such page would be considered very irrelevant to the document and as such the document would not be considered highly similar to any short query. That apparent advantage can however be considered a drawback since long document that can contain terms specified in query only in the title and the abstract and yet be still very relevant to the query. The importance of these terms will be low in comparison to a short document that contains

the same terms in the footer. More advanced application can calculate the importance of terms differently, for example by preferring terms that appear at the beginning of the document.

Another drawback of the VSM document representation is that the order of the terms is lost and it is not possible to prefer documents that contain query terms that are close to each other, over documents that contain terms separated in different parts of the document.

For the problems with exact match or synonymous, the same techniques can be used that are used for the Boolean model. The user can also choose the minimum similarity of retrieved documents to increase the retrieval precision. However increasing the threshold will also decrease the recall.

The Probabilistic retrieval Model is based on assumptions that are made explicitly – like assuming that 50% of document containing a term are relevant to that term – however not all assumptions fit the reality. The total number of relevant documents has to be guessed and p is a constant which is not always true [6]. To achieve precise results the probabilistic retrieval model requires that terms are independent. The weight calculation ignores the term frequency and position within documents, and therefore longer documents are promoted.

7. CONCLUSION

Three models supporting information retrieval were covered, with a particular emphasis on their mode of representation of the documents and their processing algorithms. As the oldest model the Boolean Retrieval model has the advantage of simplicity and convenience. It is however restrictive in the formulation of similarities. Similarity can be in two states only: true or false. At the other extreme, the probabilistic approach is an attempt to model the subjective features of the information retrieval process over a range of probabilities. The calculation of probabilities requires the specification of assumptions that can be highly biased and inconsistent. The document representation in the probabilistic model is very simple and ignores terms frequency or position. The Vector Space Model on the other hand combines clarity with flexibility. The underlying algebraic model is well specified and well understood, and the documents are represented with more details. The VSM offers a viable compromise in information retrieval processing.

8. REFERENCES

[1] Abual-Rub, M. S., Abdullah, R., and Rashid, N. A. 2007. A Modified Vector Space Model for Protein Retrieval. *Journal of Computer Science and Network Security* 7 (9), 85-89

[2] Berry, M. W., Drmac, Z., and Elizabeth, J. R. 1999. 'Matrices, Vector Spaces, and Information Retrieval'. Journal on SIAM Review 41 (2), 335-362

[3] Fuhr, N. 1992. 'Probabilistic Models in Information Retrieval'. *Journal on Computer* 35 (3), 243-255

[4] Grossman, D. A., and Frieder, O. 2004. Information Retrieval-Algorthims and Heuristics. 2nd Edition edn. Netherlands: Springer.

[5] Google 2014. Google Help [online] available from http://www.google.com/support/websearch/bin/static.py?hl=en&page=guide.cs&guide=1186810&answer=106230&rd=1 [March 2014]

[6] Jones, K. S., Walker, S., and Robertson, S. E. 2000. A Probabilistic Model of Information Retrieval: Development and Comparative Experiments'. Journal on Information Processing and Management 36 (6), 779-808

[7] Polyvyanyy, A., and Kuropka, D. 2007.A Quantitative Evaluation of the Enhanced Topic-Based Vector Space Model : A Technical Report of the Hasso-Plattner-Institute, 19

[8] Manning, C. D., Raghavan, P., and Schütze, H. 2008. Introduction to Information Retrieval. United States: Cambridge University Press

[9] Robertson, S. E., van Rijsbergen, C. J., and Porter, M. F. 1981. *Probabilistic Models of Indexing and Searching.* 'Proceedings of the 3rd Annual ACM Conference on Research and Development in Information Retrieval'. Kent, UK: Butterworth

[10] Singhal, A., and Salton, G. 1995. Automatic Text Browsing using Vector Space Model. *in Proceedings of the Dual-use Technologies and Applications Conference. 318- 324*

[11] Singhal, A., and Google, I. 2001. Modern Information Retrieval: A Brief Overview. *Bulletin of the IEEE Computer Society Technical Committee on Data*, 24

[12] Shah, C. 2009. Retrieval Models-1. USA

[13] Salton, G., Fox, E. A., and Wu, H. 1983. Extended Boolean Information Retrieval. *Journal of Commun. ACM* 26 (11), 1022-1036