МАТЕМАТИКА И МАТЕМАТИЧЕСКО ОБРАЗОВАНИЕ, 2002
MATHEMATICS AND EDUCATION IN MATHEMATICS, 2002
Proceedings of Thirty First Spring Conference of
the Union of Bulgarian Mathematicians
Borovets, April 3–6, 2002

EXTENDED BOOLEAN OPERATIONS IN LATENT SEMANTIC INDEXING SEARCH

Jivko Steftchev Jeliazkov, Preslav Ivanov Nakov

The paper presents a method for the usage of Boolean expressions for information retrieval based on Latent Semantic Indexing (LSI). The basic binary Boolean expressions such as OR, AND and NOT (AND-NOT) and their combinations have been implemented. The proposed method adds a new functionality to the classic LSI method capabilities to process user queries typed in a natural language (such as English, Bulgarian or Russian) used in the "intelligent" search engines. This gives the user the opportunity of combining not only distinct words or phrases, but also whole texts (documents) using all kinds of Boolean expressions. An evaluation of the implementations has been performed using a text collection of religious and sacred texts.

Introduction. The classic search engines give the user the opportunity to use keywords and/or Boolean expressions containing keywords. The "intelligent" search engines can process queries in natural language but do not permit the usage of Boolean expressions. We focus on the design of appropriate functions and mechanisms that will give the user an opportunity to combine free-form queries with Boolean operations in order to get better search results. The goal is achieved by combining the classic LSI algorithm with sophisticated implementation of the appropriate Boolean operations.

Latent Semantic Indexing. The *Latent Semantic Indexing* (LSI) is a powerful statistical technique for information retrieval. It is a two-stage process that consists of (see [2, 3, 4] for details): *off-line* construction of document index, and *on-line* respond to user queries.

The off-line part is the training part when LSI creates its index. First a large word-to-document matrix X is constructed where the cell (i, j) contains the occurrence frequency of the i-th word into the j-th document. After that, a *singular value decomposition* (SVD) is performed, which gives as a result three matrices D, T (both orthogonal) and S (diagonal), such that $X = DST^t$. Then all the three matrices are truncated in such a way that if we multiply the truncated ones D', S' and T', we get a new matrix X' which is the least-squares best fit approximation of X. This results in the compression of the original space in a much smaller one where we have just a small number of significant factors (usually 50-400). Each document is then represented by a vector of low dimensionality (e.g. 100). It is possible to perform a sophisticated SVD, which speeds up the process by directly finding the truncated matrices D', S' and T' (see [1]).

The on-line part of our search engine (and of LSI) receives the query (pseudo-document) that the user typed and finds out its corresponding vector into the document space constructed by the off-line part using a standard LSI mechanism. Now we can measure the degree of similarity between the query and the indexed documents by simply calculating the cosine between their corresponding vectors. (see [5, 6])

Boolean operations. Consider an e-commerce portal tracking the users' purchases in order to offer them personalised advertisement: banners, etc. We can think of the purchases as query components and of the advertisement as a new document in the same space. We need some kind of a similarity function that will give us a measure of the similarity between our advertisements and the users "profile". Let us define d_1, d_2, $\ldots$, d_n as distances (in LSI sense) between the ad and the n components of the query. The classic LSI algorithm calculates the cosines between the vectors to find the degree of their similarity. Most of the similarity measures for the Boolean operations we propose below are based on Euclidean distances, although we can use some other distances (angle, Manhattan distance, Chebishov distance, power distance, etc.). It is important to note that we must first normalise the vectors before calculating Euclidean distances. All Boolean operations proposed return a value between 0 and 1. Almost the same results can be obtained using the classic cosines but for some functions it is difficult to fit the values returned in the interval $[0, 1]$. There are several similarity measures we have experimented with:

- **OR-similarity measure.** This measure depends only on the minimal distance between the document and the query components and has the following general representation: $S_{or} = f(\min(g(d_1), g(d_2), \ldots, g(d_n)))$, where $f(x)$ and $g(x)$ are some one-argument functions.

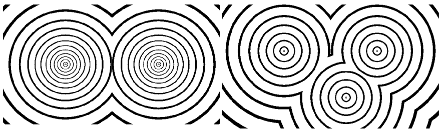

Fig. 1. OR similarity for two and three component query

In case we have more information for the query we can add weights to the query components and modify $g(x)$ to $g(x, w)$. So the formula is:

$$S_{or} = f(\min(g(d_1, w_1), g(d_2, w_2), \ldots, g(d_n, w_n))).$$

OR similarity measure has well separated picks at the query components vectors. The similarity measures for two- and three-component query, $f(x) = 1/(1 + x)$, $g(x) = x$ are shown on figure 1.

- **AND-similarity measure.** This measure depends only on the sum of distances between the document and the query components. This measure has the following general representation:

$$S_{or} = f(g(d_1) + g(d_2) + \cdots + g(d_n))),$$

where $f(x)$ and $g(x)$ are some one-argument functions. Usually this measure can be thought of a superposition of distinct similarity measures of the query components.

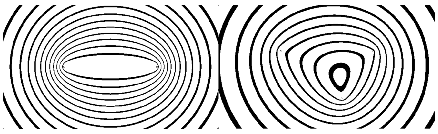

Fig. 2. AND similarity for two and three-component query

The similarity measure for two- and three-component query, $f(x) = 1/(1+x), g(x) = x$ are shown on figure 2.

• **Combination of the previous two (AND-OR)**. This similarity measure is a combination between the previous two. $S_{and-or} = f(S_{and}, S_{or})$. We can use linear combination between S_{or} and S_{and} measures. $S = k.S_{or} + (1-k).S_{and}$, where k is constant and $0 \leq k \leq 1$.

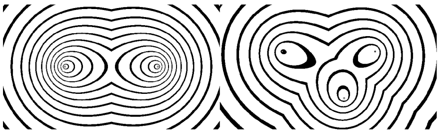

Fig. 3. Combined similarity for 2 and 3 comp. query, $k = 0.5$

Figure 3 shows the two- and three-component query results for $k = 0.5$. We still have two distinct parts like the OR-similarity function but higher values in the middle region between them just like the AND-similarity function. Figure 4 is an example of a weighed combined similarity measure.

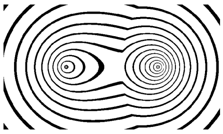

Fig. 4. Weighed combination for 2 component query, $k = 0.5$

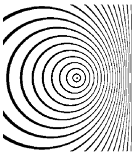

Fig. 5 NOT (AND-NOT) similarity measure

• **Binary NOT (AND-NOT)-similarity measure.** A common problem with the search engines is that they often return too much documents considered as "very similar" to the user query. In this case the user could specify what to exclude. So we have a composite query consisting of two natural language phrases: one saying what to include and the other one what to exclude. This leads to the creation of the binary NOT similarity measure. If a document is more similar to the exclude document text it will receive a similarity measure of 0 (see the second clause below). Otherwise, we return a similarity measure between 0 and 1, that takes in account the distances to both documents. We can define the NOT-measure $S_{not} = 1 - d_1/(1+d_2)$, when $d_1 < d_2$, and $S_{not} = 0$, else. The result is shown on figure 5.

Application to Religious and Sacred Texts. The Boolean operations we propose have been tested on a document collection of religious and sacred texts we found at `http://davidwiley.com/religion.html`. We selected 196 different religious and sacred texts from 14 categories: apocrypha (acts, apocalypses, gospels, writings), Buddhism, Confucianism, Dead Sea scripts, The Egyptian Book of the Dead, Sun Tzu: The Art of War, Zoroastrianism, The Bible (Old and New Testaments), The Quran and The Book of Mormons. The experiments were performed in a 30 dimensional space with a preliminary to SVD replacement of the frequencies in X (196 documents × 11451 words) with logarithms (see [6] for detailed explanation). Fig. 6 illustrates the inter-document similarities given by the correlation matrix (196 × 196), shown in 5 different colours for the five correlation intervals: 87,5–100%, black colour; 75–87,5%, dark grey; 62,5–75%, grey; 50–62,5%, light grey; 0–50%, white.

The dark rectangles in the main diagonal show the high correlation between texts belonging to the same religion. For example: the black rectangle from the bottom right corner contains texts from the Book of Mormons. To the left and up on the main diagonal can be found the Quran, then the Old Testament (The Bible), then come the Zoroastrian texts, The New Testament (The Bible), the Sun Tzu's Art of War, the Egyptian Book of the Dead and so forth. And the smooth rectangle in the upper left corner shows the relatively high similarity between all kinds of apocrypha present. We see for example that The Book of Mormons is more correlated to the New Testament than to The Old Testament.

The first class of experiments was "practical" and included composition of two or more different queries and their combination with Boolean operations we implemented.

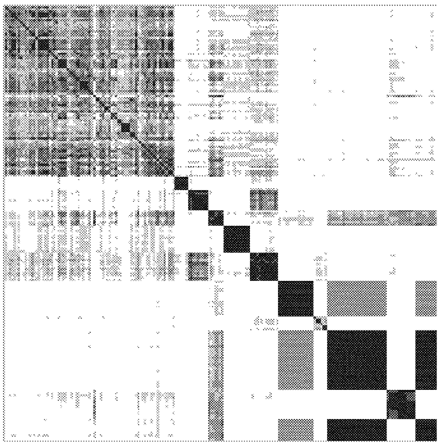

Fig. 6. Correlation between religious text (196 × 196)

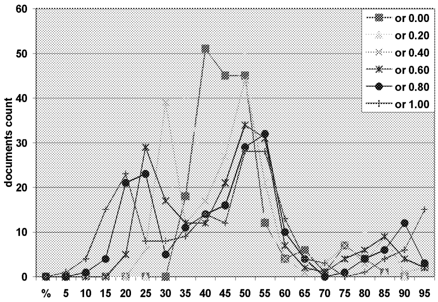

Fig. 7. Correlation between religious text (196 × 196)

The second class of "theoretical" experiments included the choice of two or more texts from the same space and performing queries using Boolean operations (OR, AND and NOT).

One of the most interesting experiments were those with the combined OR and AND similarity measure search using different values for the parameter k. Fig. 7 shows the distribution of the correlation coefficients returned by our search engine for the 196 documents using a text from the Sun Tzu's Art of War and another one from the Egyptian Book of the Dead. We can see that the results vary which suggests that we can obtain quite different results by just tuning the parameter k. The tables below show an example of the Boolean operations at work.

Conclusion. We consider the technique of LSI to be very important in the future and continue our experiments with new kinds of similarity functions and their behaviour on different types of texts. Further work concerns study of the dependence of the best similarity function upon the text collection parameters.

```
Z:\ >new_doc
suntzu1.txt

  SUNTZU1.TXT: 1.00000000
 SUNTZU10.TXT: 0.96812259
  SUNTZU8.TXT: 0.96652910
 SUNTZU11.TXT: 0.93972055
  SUNTZU3.TXT: 0.93858290
  SUNTZU9.TXT: 0.93604917
  SUNTZU5.TXT: 0.93365826
  SUNTZU2.TXT: 0.93192063
  SUNTZU6.TXT: 0.93054489
  SUNTZU4.TXT: 0.92828905
  SUNTZU7.TXT: 0.92593509
    CONF2.TXT: 0.91226262
 SUNTZU13.TXT: 0.91114359
 SUNTZU12.TXT: 0.85816521
    CONF1.TXT: 0.82958573
    CONF5.TXT: 0.78001097
    CONF3.TXT: 0.75975322
    CONF8.TXT: 0.75835731
    CONF9.TXT: 0.74499495
 PLNSENCA.HTM: 0.73306848
    CONF7.TXT: 0.71974991
    CONF4.TXT: 0.71070083
  COMRULE.HTM: 0.68268537
    CONF6.TXT: 0.68213084
  APCTHOM.HTM: 0.65737315
 TOMCNTND.HTM: 0.65597126
 REPORTPL.HTM: 0.64527600
 ACTPTNPL.HTM: 0.64414208
 REPTPILT.HTM: 0.63537118
      BKS.HTM: 0.63271447
 CONSTITU.HTM: 0.60308042
...........................
```

```
Z:\ >new_doc_bool
suntzu1.txt conf1.txt AND

    CONF2.TXT: 0.93506011
  SUNTZU1.TXT: 0.91479286
    CONF1.TXT: 0.91479286
 SUNTZU13.TXT: 0.90802675
  SUNTZU8.TXT: 0.90745685
 SUNTZU10.TXT: 0.89604427
  SUNTZU2.TXT: 0.88122315
  SUNTZU3.TXT: 0.87397853
 SUNTZU11.TXT: 0.86994911
    CONF5.TXT: 0.85956386
  SUNTZU6.TXT: 0.85553152
    CONF3.TXT: 0.85421266
    CONF8.TXT: 0.84673427
  SUNTZU5.TXT: 0.84201628
    CONF9.TXT: 0.83800404
  SUNTZU4.TXT: 0.83730812
  SUNTZU9.TXT: 0.83378707
  SUNTZU7.TXT: 0.82986823
    CONF7.TXT: 0.82701671
    CONF4.TXT: 0.80956085
    CONF6.TXT: 0.78754286
 SUNTZU12.TXT: 0.77238908
 PLNSENCA.HTM: 0.71858016
  COMRULE.HTM: 0.65251025
 TOMCNTND.HTM: 0.64263567
 SENTANCE.HTM: 0.60424740
      BKS.HTM: 0.58734007
  APCTHOM.HTM: 0.57461640
  FGAPCPT.HTM: 0.57040597
  GOSMARY.HTM: 0.56554976
  MYSTERY.HTM: 0.56087279
...........................
```

```
Z:\ >new_doc_bool
suntzu1.txt conf1.txt NOT

  SUNTZU1.TXT: 1.00000000
 SUNTZU10.TXT: 0.98252302
  SUNTZU8.TXT: 0.98189181
 SUNTZU11.TXT: 0.96651472
  SUNTZU3.TXT: 0.96605616
  SUNTZU9.TXT: 0.96306676
  SUNTZU2.TXT: 0.96280884
  SUNTZU5.TXT: 0.96209854
  SUNTZU6.TXT: 0.96099163
  SUNTZU4.TXT: 0.95893613
  SUNTZU7.TXT: 0.95728178
 SUNTZU13.TXT: 0.95335401
 SUNTZU12.TXT: 0.91590555
 PLNSENCA.HTM: 0.84335849
  COMRULE.HTM: 0.80440870
 TOMCNTND.HTM: 0.78884876
  APCTHOM.HTM: 0.77033574
      BKS.HTM: 0.76180693
 REPORTPL.HTM: 0.75937276
 ACTPTNPL.HTM: 0.75767365
 REPTPILT.HTM: 0.75380184
  FGAPCPT.HTM: 0.73594581
 MARTBART.HTM: 0.73030361
 CONSTITU.HTM: 0.72582505
  MYSTERY.HTM: 0.72447034
 ACTJNTHE.HTM: 0.71790911
  APCJMS1.HTM: 0.71604588
   ACTMAT.HTM: 0.71421530
  REVSTEV.HTM: 0.71192762
  NAGHAM6.HTM: 0.70218790
 DEATHPLT.HTM: 0.70177880
...........................
```

```
Z:\ >new_doc_bool
suntzu1.txt conf1.txt OR

  SUNTZU1.TXT: 0.99999997
    CONF1.TXT: 0.99999993
    CONF2.TXT: 0.82317039
  SUNTZU8.TXT: 0.72128584
 SUNTZU10.TXT: 0.69019913
 SUNTZU13.TXT: 0.68069817
  SUNTZU2.TXT: 0.61376306
  SUNTZU3.TXT: 0.60314637
 SUNTZU11.TXT: 0.59609037
    CONF5.TXT: 0.57673504
    CONF3.TXT: 0.57598572
  SUNTZU6.TXT: 0.56392683
    CONF8.TXT: 0.55314244
  SUNTZU5.TXT: 0.54518708
    CONF9.TXT: 0.53777290
  SUNTZU9.TXT: 0.53540358
  SUNTZU4.TXT: 0.53511372
    CONF7.TXT: 0.52538150
  SUNTZU7.TXT: 0.52381202
    CONF4.TXT: 0.49115954
    CONF6.TXT: 0.46332613
 SUNTZU12.TXT: 0.44084160
 PLNSENCA.HTM: 0.38921508
  COMRULE.HTM: 0.35047941
 TOMCNTND.HTM: 0.34533693
 SENTANCE.HTM: 0.32821505
      BKS.HTM: 0.32180010
  APCTHOM.HTM: 0.31799264
  FGAPCPT.HTM: 0.31500315
  GOSMARY.HTM: 0.31317246
 REPORTPL.HTM: 0.31256336
...........................
```

REFERENCES

[1] M. Berry, T. Do, G. O'Brien, V. Krishna, S. Varadhan. SVDPACKC (Version 1.0) User's Guide. April 1993.

[2] S. Deerwester, S. Dumais, G. Furnas, T. Laundauer, R. Harshman. *Indexing by Latent Semantic Analysis. Journal of the American Society for Information Sciences*, **41** (1990), 391–47.

[3] T. Laudauer, P. Foltz, D. Laham. Introduction to Latent Semantic Analysis. Discourse Processes, 25, 259–284.

[4] LSA 1990-99, see `http://lsa.colorado.edu`

[5] P. Nakov. Getting Better Results with Latent Semantic Indexing. In Proceedings of the Students Presentations at ESSLLI-2000, 156-166, Birmingham, UK, August 2000.

[6] P. Nakov. Latent Semantic Analysis of Textual Data. In Proceedings of CompSysTech'2000, Sofia, Bulgaria. June 2000.

Rila Solutions
Acad. G. Bontchev Str.
1113, Sofia, Bulgaria
e-mail: `Jivko.Jeliazkov@rila.com, Preslav.Nakov@rila.com`

РАЗШИРЕНИ БУЛЕВИ ОПЕРАЦИИ ЗА ТЪРСЕНЕ ПО МЕТОДА НА ЛАТЕНТНОТО СЕМАНТИЧНО ИНДЕКСИРАНЕ

Живко Стефчев Желязков, Преслав Иванов Наков

Представен е метод за използване на булеви функции при извличане на информация по метода на латентното семантично индексиране (ЛСИ). Реализирани се основните булеви операции ИЛИ, И и НЕ, както и техни комбинации. Предложеният метод добавя нова функционалност към класическите възможности на ЛСИ за обработка на потребителски заявки на естествен език (английски, български, руски), използвани от интелигентните търсещи машини. Това дава възможност на потребителя да комбинира не само отделни думи, но и цели текстове, използвайки всевъзможни булеви операции. Действието на операциите е демонстрирано върху колекция от религиозни текстове.