Validating a Model Predicting Retrieval Ordering Performance with Statistically Dependent Binary Features

International Journal of Information Retrieval Research.
5 (1): 2015, 1-18

Robert Losee
losee@unc.edu
Manning Hall, CB#3360
U. of North Carolina
Chapel Hill, NC 27599-3360
USA

Validating a Model Predicting Retrieval Ordering Performance with Statistically Dependent Binary Features

Keywords: Simulation, Modeling, Information Retrieval, IR Performance, Feature Dependencies

Abstract

Information retrieval performance may be measured retrospectively, after experiments have been conducted or other data has been gathered, or performance may be predicted for systems before retrieval, given the system, document, and user parameters. While many retrieval models assume the statistical independence of features, such as natural language terms, retrieval systems may be implemented consistent with feature dependence. The Information Retrieval Dependence (IRD) model predicts retrieval performance, with some or all dependencies and where there are binary features. Simulations with the Information Retrieval Validation (IRV) software are described that have been used to validate the IRD predictive model, showing that the IRD model accurately or exactly predicts retrieval performance under a variety of conditions. Instead of using traditional research methods using a sample of realistic documents and realistic queries, we exhaustively examine all document, query, and relevance combinations within a certain size range. The performance impact of making incorrect dependence assumptions is also considered. These predictive methods may be used in a range of retrieval and media applications that examine the similarity between statements of needs such as "queries," and documents, media, genes, and realia.

Introduction

When improving a science, such as Information Retrieval, models that describe what occurs are developed and compared to data (McCullagh, 2002; Shiflet & Shiflet, 2014). Models are used in developing further capabilities of scientists to describe what is happening, predict future occurrences, and to understand why the results are occurring. Below, we describe performance models of information retrieval systems and show how a model can be validated, within certain limits, by generating all possible document orderings or term combinations of a certain size. Empirical results showing a range of binary feature dependencies are compared to what is predicted. This enables one to make a claim that a particular scientific model is correct, given the assumptions of the model and the limitations imposed by the time needed to generate result data. This is different than most experimental studies, which determine performance levels associated with certain assumptions or techniques with existing data sets, such as TREC studies, and try to obtain better performance results than certain other methods have achieved with this same data set. These empirical studies use a relatively small set of queries with a set of real documents and assigned relevance judgments. Our study below will examine exhaustively a large number of generated queries, which can range from the thousands to millions of queries, with similar numbers of generated documents, and all the possible permutations of relevance values. While these generated data sets are artificial, as compared to actual natural language queries and documents, the exhaustive generation of all possible query characteristics, document characteristics, and relevance judgments within certain ranges may provide a more rigorous study of all the possible document sets within certain ranges, while at the same time examining large numbers of generated query and document combinations.

There are several basic models of Information Retrieval Systems. Vector systems may treat both documents and queries as vectors, with retrieval decisions being determined from the angle between the query and document vectors. Probabilistic retrieval often emphasizes the probabilities of various factors used in making decisions, such as the probability a document is relevant. Documents with higher expected probabilities of relevance are ranked ahead of those with lower expected probabilities of relevance. Language models suggest that the ranking of documents may be based upon the probabilities that query features are produced given the ordered set of features in each document. Other models include Boolean retrieval, where statements of the characteristics of documents to be retrieved are combined with Boolean operators of *and*, *or*, and *not*. Support Vector Machines and dimensionality expansion techniques modify the system of features so that an improved separation of classes of documents will occur. These and other models all have a wide range of assumptions and parameters that can be studied in order to implement and to improve the performance of the systems.

In the work below, we examine a probabilistic performance model that predicts a particular measure of retrieval performance (Losee R. M., 1998). This model can predict performance under feature independence assumptions or under feature dependence conditions. Python 3 code has been written and made available at http://ils.unc.edu/~losee/irv that can predict retrieval performance given certain parameters. This Information Retrieval Validation (IRV) software can iterate through all the possible document and term sets of certain sizes and characteristics, and the performance with these synthesized documents and terms can be evaluated and then compared with the predicted results. The predictions produced by the model match the empirical performance, suggesting that the model is accurate, at least within the size constraints tested. Using this IRV software, the effects of making incorrect statistical dependence assumptions or statistical independence assumptions are studied, when trying to make accurate estimations of the degree of dependence between features.

Information Retrieval Performance Measures

The performance of retrieval systems can be measured with several different assumptions in mind. Earlier retrieval systems often dealt with hundreds of documents to hundreds of thousands of documents. These systems were most commonly high-recall systems, in which it was considered desirable to retrieve almost all the documents on a certain topic. Performance measures that were useful in these systems include precision, the percent of documents that were relevant that were retrieved up to a certain point in the retrieval process, and recall, the percent of the relevant documents that were retrieved up to the point in question in the retrieval process. These, and many other popular retrieval performance measures, assume binary relevance, so that a document is either relevant to a query or information need, or not relevant. Precision and recall retrieval results may be displayed as a graph with precision shown on the y axis and increasing recall on the x axis.

High-precisions systems attempt to retrieve a few documents, with these documents being of the highest quality. Measures of performance that are commonly used when evaluating high precisions system include Mean Average Precision, the precision at an arbitrarily chosen place within the ordered list of documents, and the Discounted Cumulative Gain. Unlike high-recall searches, which attempt to locate almost all the documents on a topic, high-precision searches attempt to find a few excellent documents. Providing very specific indexing terms or automatically selecting rare terms that occur in a document usually results in better performance with higher precision searches, while often sacrificing recall. Systems designed for higher-recall choose broader index terms, controlled vocabularies, or more common vocabulary terms, so that when one searches using these broad terms one locates most of the documents on the topic, at the cost of also retrieving a substantial number of non-relevant documents.

Measuring performance as the length of a search, the number of documents examined, is another means of measuring performance. Some search length measures have interpretive advantages over other measures. For example, Cooper's Expected Search Length (ESL) measures the average number of non-relevant documents retrieved (Cooper, 1968). If one assumes that there is a cost to examining a non-relevant document, the ESL may be interpreted as the average economic cost of searching. By minimizing the ESL, the economic cost of searching may be minimized.

These performance measures are used primarily for the retrospective examination of search results. When examining the results of systems designed to retrieve documents, given a query and relevance judgments, these measures are highly useful for allowing one to compare the performance of one retrieval method or algorithm with another method, enabling one to advance Information Retrieval. However, these methods have not been used in most cases in the exact prediction of *future* results. While methods may be developed that would allow performance using these measures to be accurately predicted, some other measures have been proposed that are easier to predict, given current methods. The author believes that performance with any measure may be predicted exactly, but that methods have not yet been developed. A measurement method that can be predicted exactly with supplied methods is examined below.

The Average Search Length (ASL) or the Expected Position of a Relevant Document (EPRD) is the average number of documents one examines when examining all documents up to the expected or average position of a relevant document, with EPRD used in predicting performance and ASL in retrospective measures of performance. We often use EPRD below to represent both predictive and retrospective measures of performance, since they are shown to be equivalent below. As a simple measure of retrospective performance, ASL and EPRD can be predicted analytically. As an example of ASL or EPRD, consider the ordering of relevant documents (R) and non-relevant documents (N) ordered from left to right as, for example, R N R N N, when they have been sorted by a system-generated document weight. The first position on the left is the location of a relevant document, as is position 3, the third from the left. The average position of these relevant documents is (1+3)/2 = 2. This is the expected position of a relevant document and the ASL in this ordered list. We also assume that documents with the same system-generated weights used in ordering have a "tied" position, and that the mean position for these identically weighted documents is used as the position for each document with this tied weight. Assume the set of ordered documents with relevance values R N R N N, with the first two documents having tied weights, and all the remaining documents in the ranked list having decreasing and different weights from each other. The average search length is thus the average of the first relevant document treated as though it were at position 1.5, and the third document, treated as being at position 3. ASL is thus (1.5 + 3)/2 = 2.25. As a second example, consider 6 documents with computed relevance or document weights (with associated relevance values) of 5 (R) 5(R) 5(N) 4(N) 3(R) 3 (N). There are two relevant documents with the position of 2 (since the first three documents, with positions 1, 2, and 3 all have a weight of 5) and one relevant document with a position of 5.5 (there are two documents with a weight of 3, positions 5 and 6). This ASL is thus computed as (2 + 2 + 5.5)/3 = 3.166.

Predicting Retrieval Performance Analytically

Many measures have been developed to retrospectively evaluate retrieval system performance, given document orderings and associated document relevance judgments. However, measures may be used to predict future performance. For example, the Expected Position of a Relevant Document (EPRD) is the prediction of the

ASL, predicting analytically from probabilistic parameters that describe the documents, given a query and the associated relevance parameters (Losee R. M., 1998). Assume at first, for the sake of simplicity, that we have optimal ranking, a single term in the query and we only consider the presence or absence of this term in relevant and non-relevant documents. Assume that there are n documents, p is the percent of relevant documents with the query term, and t the percent of all documents with the query term. The EPRD is computed in this situation with the Analytic Retrieval Model (ARM) as $EPRD = n ((1-p+t)/2) + 1/2$.

Consider 2 relevant documents, with the same weights, both having the query term, followed by a relevant and a non-relevant document, both lacking the query term and having the same discrimination weight. Conceptually, there are 2 relevant documents at position 1.5 and a single relevant document at position 3.5, with $ASL = (1.5 + 1.5 + 3.5)/3 = 2.1666$. In the case where there are 4 documents, with one half having the term in question, $t=1/2$, and where two thirds of the relevant documents have the term in question, $p = 2/3$, the $EPRD = 4 ((1- (2/3) + (1/2))/2) + 1/2 = 2.1666$.

If we do not assume optimal ranking and choose to continue with the single term model, there are two rankings: documents ordered from highest to lowest discriminators or weights, or documents ordered from lowest to highest discriminator values or weights. Assume that the optimal document ranking from highest to lowest feature discriminators occurs with probability Q and that the worst-case ranking, ranking is the reverse from the optimal ranking. The $EPRD$ becomes $n [Q ((1-p+t)/2) + (1-Q)(1-(1-p+t)/2)] + 1/2$. When this equation has a Q of 1, that is, the ranking is perfect, then this equation reduces to $EPRD = n ((1-p+t)/2) + 1/2$, the formula above that assumes optimal ranking. We refer to $A=(1-p+t)/2$, so that $EPRD$ becomes $n [Q A + (1-Q)(1-A)] + ½$ for a single term. The variable A may be interpreted as the relative position of a relevant document in an optimal ranking of documents, with A ranging from 0 to 1, with $A=0$ being best and $A=1$ being worst. The variable A is a function of the difficulty of matching the documents and the query, possibly because of a match or mismatch between the terms used for the query and the terms in the relevant documents. The quantity Q is the probability of achieving the optimal ranking. It is often a reasonable simplifying assumption to treat Q as 1, assuming that ranking system itself is optimal. Losee and Paris (1999) showed over a decade ago that some commercial systems had Q values in the range from *0.75* to *1*, and it is likely that modern search engines have been tweaked to produce much higher values, approaching Q=1.

A general view of the Analytic Retrieval Model predicting performance is based on the survival function (Losee R. M., 1998, p. 118),

$$A = \int_{\mu_r}^{\infty} \Pr(d)\,\mathrm{d}d \ ,$$

where *A* becomes the portion of all of the documents at, or above, the mean position for the relevant documents, μ_r, and where *Pr(d)* is the probability of a document profile or set of characteristics. Computed as *1* minus the cumulative distribution function, the survival function may be applied to multiple feature models, most easily with the assumption that *Q=1*. The model described in this equation has a discrete form, where, for each document profile or set of terms, the percent of relevant documents with this profile is multiplied by the percent of all documents having this profile. Computed as

$$A = \sum_i \Pr(d_i|Rel)\left[C_i(D) - \frac{\Pr(d_i)}{2}\right] \tag{1}$$

where Pr*(*d_i*|Rel)* is the percent of relevant documents that have profile or set of terms d_i and the remaining part of the expression $C_i(D)$ is the cumulation of all documents from profile d_i through all other *D* values above it, up to the top value, and subtracting half the Pr(d_i) value modifies the cumulation so that it is only from the middle of the d_i values up to the best (top) profile. This is a discrete form of the survival function. Given these forms of the survival function, one may compute *A* for continuous or discrete features, and for different distributions for features, such as the normal, Poisson, or Bernoulli distributions (Losee R. M., 1998, pp. 119, 121). Church has developed a combinatoric model of retrieval performance, consistent with this probabilistic analytic model, and includes a predictive model for precision that can be used for analytic modeling of high precision retrieval (Church, 2010).

Feature Dependence

Natural language terms, DNA components, and aspects of other information-carrying entities have a degree of occurrence dependence that is not consistent with the assumptions of many retrieval models that assume statistical independence of features. For example, natural language phrases such as *published article* occur far more often than phrases with randomly selected terms like *published armadillo*. There is a degree of dependence between the term *published* and the term *article*, with their co-occurrence being higher than random. Non-text, such as images, will similarly have a degree of dependence between features. For example, consider the likelihood that one pixel on an image will be the same or a very similar color to adjacent pixels. One can measure the amount of error due to incorrectly assuming independence in part by comparing the performance obtained by (incorrectly) assuming independence, when the true case is dependence, with the performance obtained assuming independence, when independence is in fact the case (Losee, Bookstein, & Yu, 1986).

One method of capturing term dependence is to directly model features with the actual degree of dependence. This approach will be taken in the remainder of this article. One may include all 2 way relationships, all 2 and 3 way relationships, and so forth to include all possible relationships. The Bahadur-Lazarsfeld Expansion provides a method for doing this (Bahadur, 1961; Losee R. M., 1994), while a Generalized Term Dependence model provides another method for capturing all desired degrees of dependence (Yu, Buckley, Lam, & Salton, 1983). Bayesian network models of dependencies have also been used to capture relationships between items (Manning, Raghavan, & Schutze, 2008; Sloman, 2013). Dependencies may address the sequencing of terms, so that ordering of terms can be captured and estimated, as well as unordered dependencies (Eguchi & Croft, 2009; Park, Croft, & Smith, 2011). Other models of dependence have been used to capture feature dependencies in retrieval and other applications, such as information filtering and natural language processing (Bendersky & Croft, 2012; Bisht, Srivastava, & Dhami, 2010; Cai, Bu, Chen, & Liu, 2007; Dang, Luk, & Allan, 2014; Nanas, Vavalis, & De Roeck, 2010). Many dependence techniques estimate a limited amount of dependence, while others measure all available or possible statistical dependence information. Many emphasize certain aspects of the data or certain characteristics of probabilistic estimation. For example, the method developed below addresses all possible degrees of dependence, but does not address ordering of features. Dependence computations that capture full dependence, such as 4 way dependencies when 4 terms are present, are needed for full accuracy of this predictive model.

One may choose to modify the data so that it meets other desirable characteristics, such as statistical independence. The dimensionality of a dataset is the number of different independent features used in the dataset. Dimensionality reduction can reduce the complexity of a dataset by trying to isolate possibly less useful information in the dataset, in the form of dependencies, and then keeping only the more important information (dimensions) left in the remaining data. One can reduce the dimensionality by finding the most important underlying factors, such as may be performed with factor analysis (Borko, 1985). Latent Semantic Indexing uses singular value decomposition to produce a modified data set which can be manipulated using the vector retrieval model (Deerwester, Dumais, Furnas, Landauer, & Harshman, 1990).

Dimensionality expansion is a broad family of methods that moves in the opposite direction from dimensionality reduction. Consider every possible combination of terms that might exist in a dataset. If each of these combinations had its own dimension or an axis on a graph, then there would be a larger number of dimensions, given this one-axis-per-combination approach, and the number of features would increase. This would be a form of dimensionality expansion. Each dimension would stand on its own, providing statistical independence between axes. Support Vector Machines represent a set of algorithms that provide dimensionality expansion that can be useful in retrieval (Manning, Raghavan, & Schutze, 2008) .

When features such as natural language terms are considered present or absent, the probabilities of these may be precisely modeled as binary terms with dependence. Below, we consider how binary features may be processed with dependence. The model used below allows one to view dependencies in terms of covariance measures, making the interpretation of this model relatively simple and attractive. Other models such as those above can be used for this, but the model used below is explicitly designed to allow one to directly convert to and from covariance values.

Binary Dependent Features

Feature independence occurs when there are n features $f_1, f_2, \ldots, f_n$ with probabilities $p_1, p_2, \ldots., p_n$. The joint probability a document has features $f_1, f_2, \ldots, f_n$ is denoted as $\Pr(f_1, f_2, \ldots., f_n)$. Independence of features occurs when the joint probability of the features equals the product of the individual probabilities, or $\Pr(f_1, f_2, \ldots., f_n) = Pr(f_1) \bullet Pr(f_2) \bullet\bullet\bullet Pr(f_n) = p_1 \bullet p_2 \bullet p_3 \bullet\bullet\bullet p_n$. We note here, with p_{00} and p_{11} representing the joint probability with joint absences and joint presence, respectively, that the following hold (Teugels, 1990):

$p_{00} = (1 - p_1)(1 - p_2) + \sigma_{12} = 1 - p_1 - p_2 + \mu_{12}$

$p_{10} = p_1 (1 - p_2) - \sigma_{12} = p_1 - \mu_{12}$

$p_{01} = (1 - p_1) p_2 - \sigma_{12} = p_2 - \mu_{12}$

$p_{11} = p_1 p_2 + \sigma_{12} = \mu_{12}$,

where the variable σ_{ij} is the covariance between features f_i and f_j, and μ_{ij} is an element of the $\boldsymbol{\mu}$ vector of second order moments. Note that the notation used here for p_{10} and p_1 and related probabilities and variables is that of Teugels (1990), and differs from that used in many information retrieval studies.

There is a relationship between the vector of probabilities **p** and the $\boldsymbol{\mu}$ vector that may be computed as follows:

$$\boldsymbol{p}^{(n)} = \begin{bmatrix} 1 & -1 \\ 0 & 1 \end{bmatrix}^{\otimes n} \boldsymbol{\mu}^{(n)}. \tag{2}$$

Here the exponents for the **p** vectors and the $\boldsymbol{\mu}$ vectors represent the number of features, with the length of each vector being 2^n items long. Here the **p** and $\boldsymbol{\mu}$ vectors are transposed for the matrix manipulation. The middle matrix is multiplied by itself n times using the Kronecker product, which is denoted by the $\otimes$ symbol. For example,

$$\begin{bmatrix}1 & -1\\0 & 1\end{bmatrix}^{\otimes 2}=\begin{bmatrix}1 & -1\\0 & 1\end{bmatrix}\otimes\begin{bmatrix}1 & -1\\0 & 1\end{bmatrix}=\begin{bmatrix}1 & -1 & -1 & 1\\0 & 1 & 0 & -1\\0 & 0 & 1 & -1\\0 & 0 & 0 & 1\end{bmatrix}.$$

While Equation 2 shows how to convert **μ** vectors into **p** vectors, Equation 3 below converts from **p** into **μ**.

$$\boldsymbol{\mu}^{(n)}=\begin{bmatrix}1 & 1\\0 & 1\end{bmatrix}^{\otimes n}\boldsymbol{p}^{(n)} \tag{3}$$

By using both Equations 2 and 3, one can compute **μ** from **p** and compute **p** from **μ**. The $\mathbf{p}^{(2)}$ component for $(p_{00}, p_{01}, p_{10}, p_{11})^T$ produces $\boldsymbol{\mu}^{(2)} = (1, 0, 0, \sigma_{12})^T$. If one wishes to modify one of the vectors, the consequent change in the second vector can be computed. This can be useful below when trying to modify the degree of covariance between features and then observing the consequences.

There is a relationship between the vector of probabilities **p** and the covariance **σ** vector that may be computed as follows:

$$\boldsymbol{p}^{(n)}=\begin{bmatrix}1-p_i & -1\\p_i & 1\end{bmatrix}^{\otimes n}\boldsymbol{\sigma}^{(n)}, \tag{4}$$

where p_i represents the probability of the ith item in vector **p** and where the Kronecker product is computed over the *n* different values so that *i* moves from *n* down to *1* (Teugels, 1990).

While Equation 4 shows how to convert **σ** vectors into **p** vectors, Equation 5 below converts from **p** into **σ**.

$$\boldsymbol{\sigma}^{(n)}=\begin{bmatrix}1 & 1\\-p_i & 1-p_i\end{bmatrix}^{\otimes n}\boldsymbol{p}^{(n)} \tag{5}$$

Interestingly, there are two transformations that may be performed upon the **p** vector to produce a second vector, using Equations 2 or 4. This second vector may, in both cases, be transformed back into the original **p** vector, using Equations 3 or 5, or the **p** vector may be modified to produce a new **p** vector using the same Equations, as is experimentally studied below.

Validating the Performance Model with Simulations

A model may be shown to be true a number of different ways. Mathematical proofs are often used, as are the application of gathered data to show that models are consistent with natural observations (McCullagh, 2002; Nickerson, 2010). Simulations may serve to generate some or all possible data that might exist, allowing one to demonstrate the correctness of a model. The retrieval model using Equations 1 and 2, with the binary term dependence computed consistent with Equations 3 through 6, can be validated by showing that the analytically predicted performance is consistent with empirically determined performance. Many empirical retrieval studies show how to improve measured retrieval performance using a particular retrieval model so that performance exceeds what some other systems achieve. The TREC studies certainly support this form of study, providing a common set of measurements and a common dataset for everyone to use, facilitating the comparison of empirical results.

The study being described here has a different goal. Unlike studies that emphasize the qualitative aspects of the human searcher, the empirical quantitative analysis of document ranking algorithms applied to real documents and queries, or mathematical proofs, we iteratively move through all sets of possible documents, queries, and relevance judgments within a certain size range. We have proposed a model that predicts Information Retrieval Validation system performance, but is this model correct? This question is the focus of this paper, and this research question is answered with this iterative method.

One method for validating an information retrieval model that predicts performance is to exhaustively evaluate the predicted performance with all the possible term permutations of the inputs to the model, in all the possible document orderings, given a defined set of parameters. For example, one might ask whether this model predicts the same performance as is empirically found when we generate all the possible document orderings of a particular size with a given number of features per documents? While this approach does not illuminate some of the problems discovered with empirical testing of real-world retrieval systems, it does show that an exhaustive examination of all possible orderings that meet certain requirements will produce the same performance orderings as what is predicted by the model, suggesting the validity of the model.

Validation by exhaustively iterating through sets of possibilities has been used in mathematics to provide support for certain arguments or theorems (Nickerson, 2010). The four-color problem in mathematics suggests that any map on a flat surface can have countries colored in all cases with 4 colors or less. It has been shown that the four-color problem could be changed into a finite set of graph theoretic relationships (Appel & Haken, 1977). All the possibilities, that is, all the possible members of a set of graph-theoretic relationships, were tried using computer software, showing that each one could be colored satisfactorily with four colors or with graph theoretic equivalents. The mathematics community debated whether this form of argument using computer software constituted a "proof" of the theorem. If the Information Retrieval Validation software is correct and it exhaustively iterates through all the possible locations in the problem space, showing that the model is correct in each case, then this can be considered a form of argument. However, this is problematic, as in the real world the number of documents (and variations within documents) is infinite, making direct exhaustive evaluation virtually impossible. Similarly, most empirical Information Retrieval studies use a small number of queries and a small fraction of the existing documents.

Other forms of mathematical arguments make predictions about numbers of certain sorts and then tests these predictions by iterating through very large numbers of test cases (Shiflet & Shiflet, 2014). One might ask if there is an approximately even distribution of the digits *0* through *9* when writing out the constant π=*3.14159….* to a certain precision, or whether there is a pattern to the digit occurrences. Software might iterate through the first thousand digits of π, or perhaps the first million, or the first hundred million, and so forth, counting the number of digit occurrences for each of the ten digits. Studies have examined this problem and found no pattern to the occurrence of digits. Unlike the four-color problem, however, where Appel and Haken (1977) went through *all* the possibilities, examining a finite set of digits in an expansion of the value of π gives us a great deal of evidence that there is no pattern to the digits in π, but it is not a complete set of data showing that for *all* the digits in the infinitely long π, there is exactly (or approximately) an equal number of the digits *0* through *9*. We can only investigate a large number of cases, with the suggestion that the results hold for all possible cases.

Applying this technique to this form of the Information Retrieval Validation model used in this article does not give us results from actual author-produced documents, or user-produced queries or relevance judgments. Using a dataset with real queries and relevance judgments serves as a form of engineering where we can infer that a particular algorithm performs at a certain performance level in a certain environment. This traditional form of experimentation has a *practical utility*, but is not necessarily the *best way to study whether predictive models in a science predict accurately*. Using a TREC database for our predictive study would let us know that for certain, non-randomly selected query and document combinations, the predictions are accurate. However, some databases were developed in such a way that documents are easier to retrieve, and experiments using these databases are not generalizable to all retrieval systems (Shaw, Burgin, & Howell, 1997). Most empirical databases are developed for differing, specific purposes. Studying a range of documents exhaustively has the advantage that it may provide stronger arguments about the effectiveness of methods.

Iterating exhaustively through all possibilities of documents and queries of a certain size range can allow consumers of research to understand the quality of the predictions; more precisely, whether the predictions are correct or inaccurate. For example, by studying all permutations of documents up to 10 documents, one might become convinced that if the models accurately predict performance for any set of documents up to size *10*, then one would conclude the model to be accurate for larger sizes, too. As the number of documents used grows, the time necessary for processing grows exponentially. For this reason, it is desirable to choose a small number of documents that is still likely to capture most or all of the relationship phenomena that occur naturally in documents.

What is a reasonable upper-limit for the needed numbers of documents to show that a model is valid? Showing that all combinations of up to *10* documents likely captures most or all the kinds of differences that might occur from among a set of realistic documents. The number of dependent terms in this set may be varied from *1* term to *4* terms, with the assumption that all oddities that might occur will occur with *4*- (or fewer) way dependencies between features.

We assume here that showing that the document ordering predictions are accurate for small numbers of documents also applies in most situations to larger numbers of documents. Showing that the ordering predictions are accurate for 2 documents, for example, may apply to a set of 3 documents, by comparing the ordering of the first and second documents, the second and third documents, and the first and the third documents. Clearly this applies in the case of a metric space but may be problematic in some other spaces. If the model is applied to 10 documents, it probably applies to 11 documents, when we consider that it would apply to each and every set of 10 documents from among the 11. It is harder to consider the extrapolation of considering 4 terms in a query and 4 terms being considered in a document with the consideration of 5 or 6 terms in a query or 5 or 6 terms being considered in documents when retrieving the documents, although there seems to be no reason why matrix methods that apply for the analysis of covariance for 4 way dependencies would not also apply to 5 or 6 way dependencies.

The Information Retrieval Validation (IRV) software compares the EPRD or ASL performance computed both empirically (retrospectively) and analytically (predictively.) The empirical or retrospective measures of EPRD or ASL is, as was discussed earlier, where we compute the position of each relevant document and arithmetically determine the average position. The analytic measures of EPRD used in the IRV software estimate the Expected Position of a Relevant Document probabilistically, as in Equation 2. Using this method, the probabilities may be estimated so as to be consistent with the actual data, or so as to be consistent with statistical independence.

There are several IRV software options that may be set to execute several different tests. One test is whether using the methods described earlier can move from the original probabilities to covariance $\boldsymbol{\sigma}$ or $\boldsymbol{\mu}$ arrays and then back to the correct probabilities. To do this, the software may be executed with two terms with either the covariance or the mu options set, moving from the probabilities to the covariance $\boldsymbol{\sigma}$ values and then back to the probabilities, as in Equations 5 and 6. Similarly, probabilities are converted to $\boldsymbol{\mu}$ values and then back to probabilities, using an implementation consistent with Equations 3 and 4. In both cases, the EPRD performance values are computed with both the initial and the final probabilities and the EPRD values are compared to see if the predictive and retrospective models produce the same results.

Another test is to examine whether the methods described above can be extrapolated beyond term pairs to n way dependencies. This may be accomplished by comparing the retrieval performance values, EPRD, using empirically determined probabilities and using probabilities computed by converting from the initial probabilities to the $\boldsymbol{\mu}$ or covariance $\boldsymbol{\sigma}$ values and then back to a probability vector. With the number of terms being varied from *1* to *4*, the EPRD values are then compared, showing that the method works over a range of term relationships, from independence to *4* way dependencies.

A third form of test, used to produce the Figures below, deliberately produces errors in the covariance $\boldsymbol{\sigma}$ and the $\boldsymbol{\mu}$ values and then produces new (incorrect) probabilities and new (incorrect) EPRD values. Three kinds of errors are available. One adds a constant of 0.01 to the covariance σ or the μ value before the probabilities are recomputed and the EPRD values recomputed. Obviously, one expects most of the performance values to be wrong in this case. A second kind of error is produced by substituting one of a range of values in place of the actual μ or covariance σ value. A third kind of error is produced by adding one of the same range of values in to the actual μ or covariance σ values. In both cases, the probabilities are recomputed from the modified covariance $\boldsymbol{\sigma}$ or $\boldsymbol{\mu}$ values and then the EPRD is computed.

Results of Validation Procedure

Using the Information Retrieval Validation (IRV) software (http://ils.unc.edu/~losee/irv) shows that the Information Retrieval Dependence (IRD) performance model is consistent with empirical results. For this analysis, sets of all possible documents were generated with binary features, with sets of documents ranging from *1* document to *10* documents, with all the possible orderings of the documents in each set generated. The empirical and analytic (predictive) results were compared. All possible relevance values are generated for all the document sets. Using the IRV software over all these ranges of possibilities, the analytic model predicts the performance results, and these are compared to empirical results based on this data, with equality of numbers being treated as one performance value being within plus or minus 0.00001 of the other. This addresses rounding issues that occurs during numeric computations.

Executing the IRV software for all numbers of documents, iterating from *1* to *10* documents, and iterating through all possible *1*, *2*, *3,* and 4 way term relationships, we find that there are *551370* correct estimates of performance for *551370* attempts. The performance estimation accuracy here is 100%, suggesting that the IRD model accurately predicts actual information retrieval ordering performance. This number of correct values was obtained both when converting to and from the probabilities via the $\boldsymbol{\mu}$ values and to and from the probabilities via the covariance, $\boldsymbol{\sigma}$.

The Information Retrieval Dependence model suggests predicted performance results that are found using the Information Retrieval Validation software to be consistent with empirical results. Use of this probabilistic predictive model is thus justified. The implication of this is that the analytic model predicting retrieval performance is accurate, at least for the range of values tested above. We believe that if the model has been determined to accurately predict empirical results from the analytic model for all possible ranges of *1* to *10* documents and all the different term possibilities and relevance possibilities, the model is highly likely to be accurate for document sets of more than *10* documents, and for larger degrees of term dependencies.

This method of generating all term and document combinations up to a certain size does not show how a real system might perform, especially in regards to human computer interface aspects. TREC and other methodologies and techniques provide a prediction of how realistic systems might perform. Using this IRV software is appropriate for showing the validity of the Information Retrieval Dependence performance model.

A variable MAKEERRORS in the Information Retrieval Validation software can be set to True (the default is False) to force the system to make errors by adding *0.01* to the covariance σ or μ values between *2* terms in a *2* term model. Executing the program with this variable set to True results in many of the performance estimates being incorrect, as one would expect.

Erroneous Statistical Independence Assumptions

Some information retrieval models are correct, some wrong, and some are approximations of the truth, the latter because modelers and system implementers making simplifying assumptions that are not consistent with data for theoretical or practical purposes. For example, what happens when assumptions of statistical independence are made but we know that there are statistical dependencies between features? Here we compute the differences between performance results with the correct assumptions about the amount of dependence, with incorrect levels of statistical dependence. We note that making errors in the assumptions of a predictive model might predict inferior performance or superior performance to what empirically occurs.

Because we have shown above that the empirical and the analytic predictive model produce the same performance results, we may use these two models in different ways in our study of errors. For our study of erroneous dependence and independence assumptions here, we will treat the true, dependence model as what is produced by the empirical measure of performance and the analytic model as the indicator of what would happen if erroneous dependence or independence was assumed. We do this by using the analytic model and treating the μ and covariance σ between features as any of a range of values, with values near *0* for the covariance occurring when variables approach statistical independence.

To determine these performance data, we use the Information Retrieval Validation (IRV) software to compare the performance difference between the actual ranking and what occurs with a single statistical dependence between two features being modified. We only use documents with two features for this study of erroneous statistical assumptions. The data comes from applying the above algorithms and empirically computing the average EPRD value for the actual performance. Performance is computed for both the analytic model of performance and the empirical measure of performance of the generated documents. The results shown in Figures 1 through 4 each show the analytic performance level for the empirical performance with the *EPRD = 4.0*.

Using a deterministic algorithm, the first set of documents and the first set of relevance values that exceeded arbitrarily chosen characteristics were used where the empirical computation of the *EPRD* was *4.0*, with a range of modifying values for the μ and the covariance σ. This produces an arbitrarily chosen set of documents and relevance values. The documents chosen produced a μ between the two features for relevant documents of 0.25 and a covariance σ between the two features in relevant documents of 0.125. Figure 1 shows a linear relationship between the μ for relevant documents modifier and the resulting EPRD. For this data, the value indicated is added to the correct μ for relevant documents, so that we can examine how a certain degree of error in the estimation of the μ in relevant documents results in increases or decreases in performance. We can understand the values that are being added (the value on the x axis) to the correct μ value as modifying the performance, after all the other parameters are established from the empirical *EPRD = 4.0* values, with the increase in the resulting μ then decreasing EPRD (improving performance,) and with a decrease in μ then increasing EPRD and thus decreasing performance. Note that in Figure 1 the EPRD value is *4* when the modifying value is *0*.

Note that modifying the μ values (and later the covariance σ values) is being performed without modifying any other parameters. After modifying the μ or covariance σ values, the $\boldsymbol{p}$ values are computed, with no other changes being made in the $\boldsymbol{\mu}$ or covariance $\boldsymbol{\sigma}$ array. For this reason, the values produced for the $\boldsymbol{p}$ array may be beyond the range of normally acceptable values.

Figure 2 shows the performance that occurs for the performance parameters shown in Figure 1 but with the modifiers shown on the x axis replacing the μ value in the estimates. As with Figure 1, the relationship between the μ and the EPRD in Figure 2 is linear.

Figure 3 shows that as the value added to the true covariance σ increases, the EPRD decreases (improves). Figure 4 similarly shows that as the covariance σ is replaced with the value on the x axis, then the EPRD performance decreases (improves.) In Figure 3, the EPRD of 4 is obtained when nothing is added to the original σ value. In Figure 4, however, the EPRD of 4 is obtained when 0.125 is substituted for the σ value.

Figures 3 and 4 show that as the value added to the covariance increases, the EPRD decreases and the performance improves. Using Equation 5 above, one can see for the two term case, which is used in these Figures, the p values can be expected to increase when the $\boldsymbol{\sigma}$ array is held constant and only the covariance σ value between terms is increased.

When actual covariance σ and μ values are increased, the overestimates of the μ and covariance σ values result in performance estimates of EPRD that are too low, that is, too optimistic about the EPRD. Conversely, underestimating the covariance σ and μ values results in an estimate of the EPRD that is too conservative, and that the actual performance is better. The relationship between the amount of over or under-estimation and the EPRD is linear, suggesting that a large over- or under-estimation of the covariance σ or μ value does not result in an exponentially or RMS relationship between the parameter estimate and the performance estimate. Risk avoiders usually do not need to worry that reasonable but larger errors will be catastrophic.

Numeric Example

Consider a retrieval situation where there are two documents, one of them relevant, with the first feature and not the second, and a set of three documents, one of them relevant, with only the second feature and not the first. The EPRD for this may be computed empirically as 1 times 1.5 (the position in the middle of the first two documents) plus 1 times 4 (the position in the middle of the last three documents), which, divided by 2 (the number of relevant documents,) produces 5.5/2 = 2.75. One can see the output from this example by using the IRV software and setting the variable ARTICLEEXAMPLE to True. Executing the software with this variable set results in the software using a data set representing the problem described here, displaying various matrices associated with this set of 5 documents, and then exiting.

For an analytic analysis using covariance methods, Equations 5 and 6, the original **p** vector for relevant documents is [0, 0.5, 0.5, 0] and the **p** vector for all documents is [0, 0.6, 0.4, 0].

The matrices produced as a result of the Kronecker product are as follows: for converting from ***p*** to covariance $\boldsymbol{\sigma}$ vectors for relevant documents,

$$\begin{bmatrix} 1 & 1 & 1 & 1 \\ -0.5 & 0.5 & -0.5 & 0.5 \\ -0.5 & -0.5 & 0.5 & 0.5 \\ 0.25 & -0.25 & -0.25 & 0.25 \end{bmatrix}.$$

For the ***p*** to covariance $\boldsymbol{\sigma}$ vector for all documents,

$$\begin{bmatrix} 1 & 1 & 1 & 1 \\ -0.6 & 0.4 & -0.6 & 0.4 \\ -0.4 & -0.4 & 0.6 & 0.6 \\ 0.24 & -0.16 & -0.36 & 0.24 \end{bmatrix}.$$

Converting the covariance $\boldsymbol{\sigma}$ vector to the ***p*** vector for relevant documents is accomplished by multiplying the following by the covariance vector,

$$\begin{bmatrix} 0.25 & -0.5 & -0.5 & 1 \\ 0.25 & 0.5 & -0.5 & -1 \\ 0.25 & -0.5 & 0.5 & -1 \\ 0.25 & 0.5 & 0.5 & 1 \end{bmatrix}.$$

Converting the covariance $\boldsymbol{\sigma}$ vector to the ***p*** vector for all documents is accomplished by multiplying the following by the covariance vector,

$$\begin{bmatrix} 0.24 & -0.6 & -0.4 & 1 \\ 0.36 & 0.6 & -0.6 & -1 \\ 0.16 & -0.4 & 0.4 & -1 \\ 0.24 & 0.4 & 0.6 & 1 \end{bmatrix}.$$

The covariance $\boldsymbol{\sigma}$ array for relevant documents is computed as [1, 0, 0, -.25] while the covariance $\boldsymbol{\sigma}$ array for all documents is [1, 0, 0, -0.24]. The original retrospective

EPRD is 2.75, while the estimated EPRD, produced from the covariance $\boldsymbol{\sigma}$ matrix conversion, as in Equation 5, is 2.75. Studying this data may allow the reader to more fully understand the relationships between the various data items used in analytically predicting retrieval performance from $\boldsymbol{\mu}$ and covariance $\boldsymbol{\sigma}$ arrays.

Practical Applications

The Information Retrieval Dependence (IRD) probabilistic model has a number of applications. Most importantly, the model can be used to describe what occurs in the performance of retrieval system operations that are consistent with the assumptions of the model. In this case, the assumptions include that the features are binary and that the features are often statistically dependent. The relationships between different parameters are examined, comparing the empirical measure of performance and the analytically predicted performance, given the empirical degree of feature dependence or a variant of this value.

When designing a retrieval system, predicting the performance of the system, given the parameters associated with the documents, queries, relevance values, and model characteristics, can be highly useful. Design of a system so that performance is increased or possibly maximized will result in greater satisfaction with the system, possibly increased use, and possibly increased profits.

Predicting what happens when errors are made can be useful in several instances. Many retrieval models assume statistical independence of features. Describing what happens when scientific errors are made is difficult in many circumstances. Here the performance is obtained where the empirical performance (x) is known to be one value but estimating performance (y) consistent with a different degree of dependence produces another, specific value, and the difference between these two performance values x and y may be computed in order to describe the effect of estimating dependence errors. There is clearly a degree of dependence between features in natural languages (Manning & Schutze, 1999). By examining performance given the actual dependences and the performance that would be obtained with simplifying assumptions and estimates about term relationships, we are able to show the nature of empirical and estimated performance differences due to modeling errors.

The understanding of retrieval performance allows one to describe and predict retrieval performance when a range of characteristics are varied. Understanding the relationships that exist between system parameters and system performance is one of the foundations of the sciences of computer performance and information retrieval, and understanding information retrieval system parameters and their relationship with retrieval performance is at the core of the science of information retrieval.

Summary and Conclusions

Information retrieval has continued to expand over the past several decades, but most of the research involves experimental testing using real-world databases. Analytic models can be used to relate the performance of information retrieval systems to the parameters of the users' queries, users' relevance values, and system databases. While models assuming a single term or statistical independence between terms can simplify the performance models, they make assumptions that are widely realized to be unrealistic. We discuss means of estimating term dependencies and these, when incorporated into an analytic retrieval performance model, provide the Information Retrieval Dependence (IRD) model that predicts retrieval performance given query, relevance values, and databases.

The validity of the IRD model is tested by comparing the performance obtained empirically with the performance predicted by the IRD model. The Information Retrieval Validation software generates all the possible sets of document, term, and relevance characteristics, within specified ranges, and shows that the empirical results are experimentally equivalent to the analytic results predicted by the IRD, except when errors are deliberately introduced.

Many retrieval models assume feature independence. The work shows how this assumption can be studied, given that terms are usually dependent upon each other. Data shows here the linear relationship between performance decreases or increases when errors in the dependence variables are used in models.

The primary weakness of this approach to studying performance is that while it tests a large number of possible document, relevance, and query possibilities, it does not examine all the possible combinations that might exist. To examine all possible combinations would require an infinite amount of time. Instead, we chose a small number of possibilities that are exhaustively evaluated, and we hope that the number is large enough so that most people would believe that if the results hold for all combinations at or below the size studied, then the same results would very likely be obtained for combinations beyond the constraints examined.

These results can be useful in the analysis of choices made in the design of future retrieval systems. For example, is the incorporation of dependencies of a certain order worth the added processing costs? What degree of natural language processing provides a cost-effective level of term dependencies? What will be the performance obtained with varying characteristics for documents, queries, and relevance values. Information retrieval will continue to improve, and the understanding of the foundational relationships between terms, queries, and documents is essential for continued improvements in performance. These models and results may also be applied to other applications where similarities are used, such as gene matching in bioinformatics and matching people in online dating services.

Future work also may apply these techniques to specific realistic query, document, and relevance combaintions, such as those in the TREC and related databases. However, the techniques described above show that this prediction method applies for all sets of any 10 documents and all queries of up to 4 terms, with full dependence and all possible relevance judgments. Note that all "real" existing sets of 10 or fewer documents with a query of 4 or fewer terms, with the document terms used in document ordering being limited to those terms in the query, produce the same performance results produced above from generated data. Note that a realistic query with 4 terms in it and one million documents can be studied as many sets of 10 ordered documents. The examination of larger queries, using the software described above, may allow many more terms to be studied, albeit likely with fewer documents, due to the exponential increase in processing time needed when increasing the numbers of terms and documents. The direct study of individual queries and documents from real databases will allow further confirmation of the above model, as well as of further models consistent with other types of performance measures and dependencies.

FIGURE 1: Estimates of the Expected Position of a Relevant Document (EPRD) predict improvements in retrieval performance, or decreases in EPRD, as the actual μ is increased by adding the value given on the x axis. Note that when no modification occurs, the EPRD is 4.

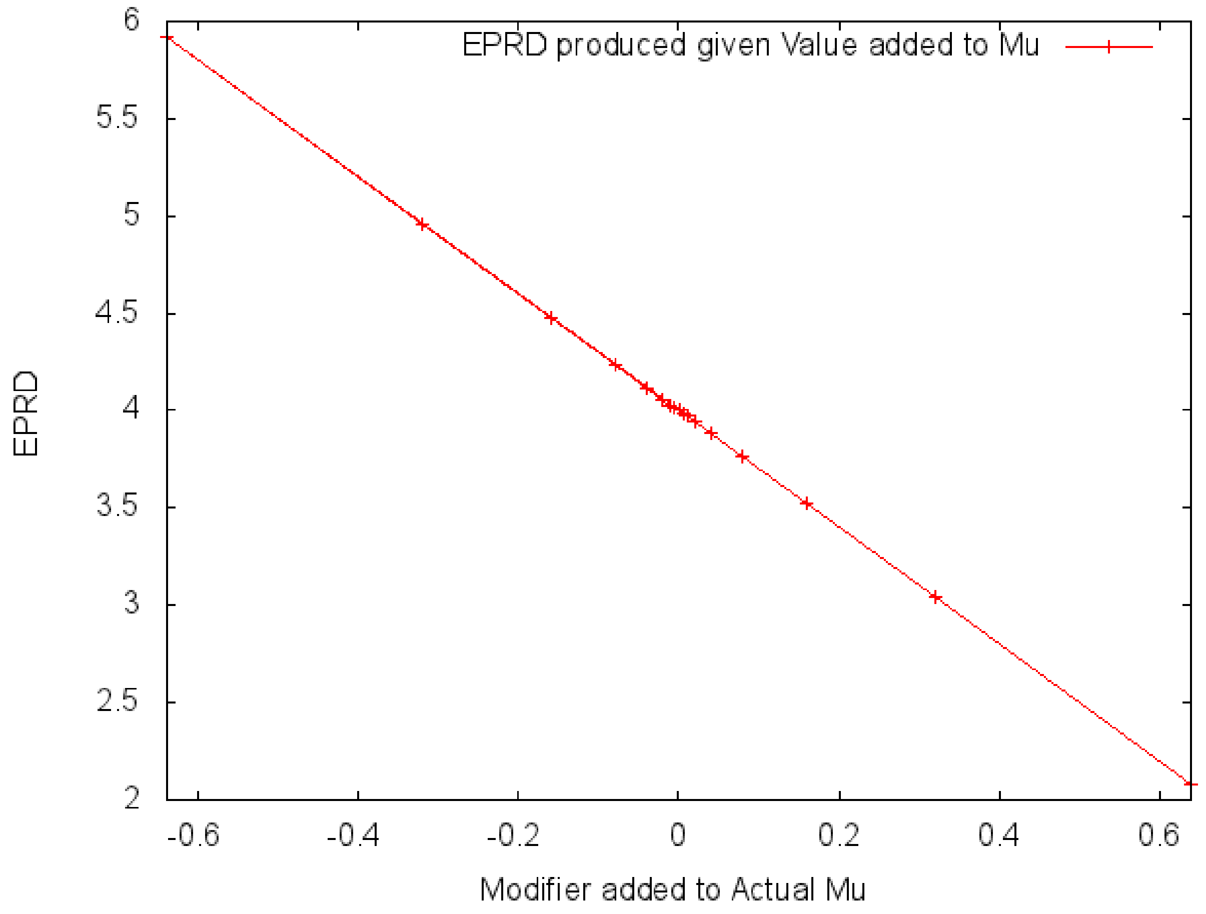


Figure 2: Estimates of the Expected Position of a Relevant Document (EPRD) changes as the actual μ is replaced with the μ provided on the x axis. When the value $\mu = 0.25$ is used, the EPRD is *4*.

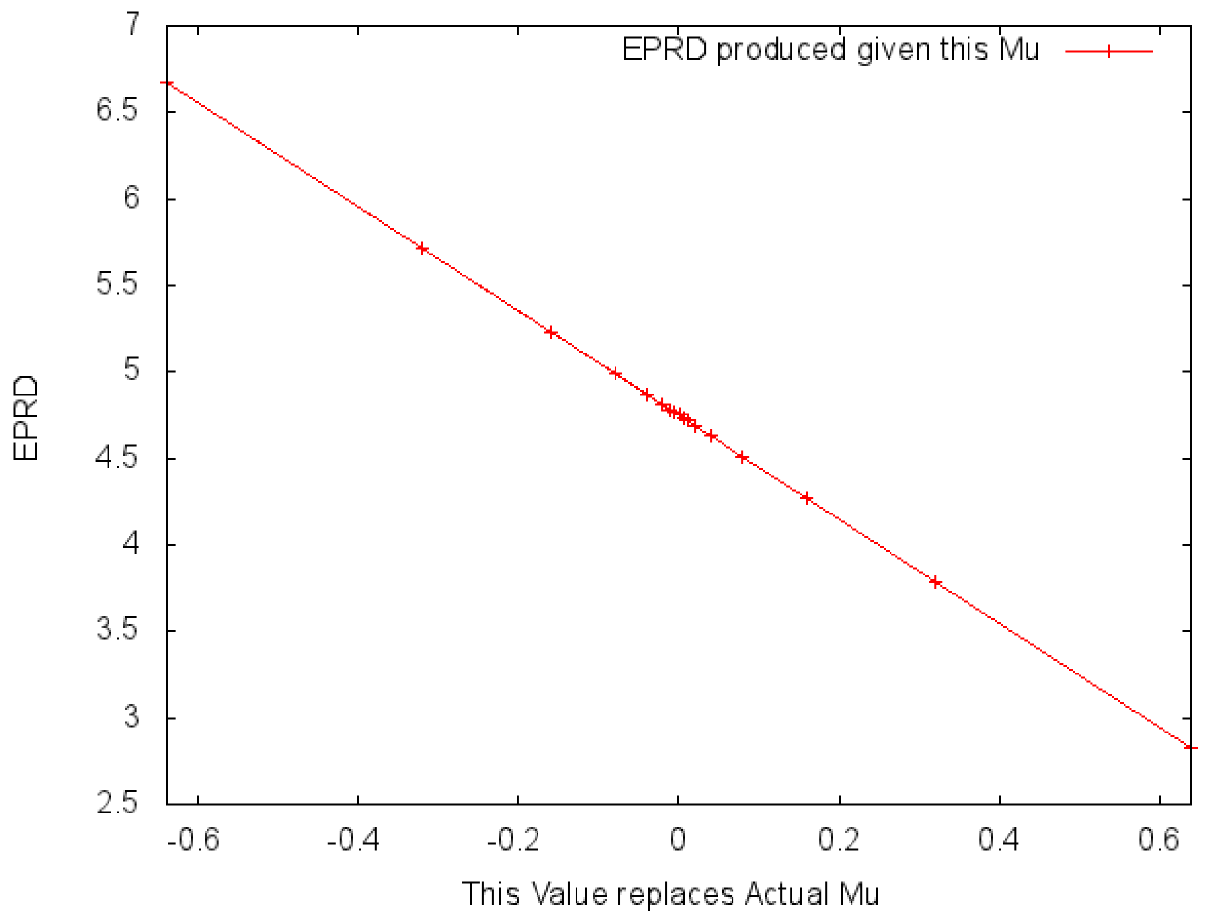


FIGURE 3: Estimate of the Expected Position of a Relevant Document (EPRD) changes as the actual covariance σ is increased by the value provided on the x axis. When no value is added to the covariance σ, the EPRD is 4.

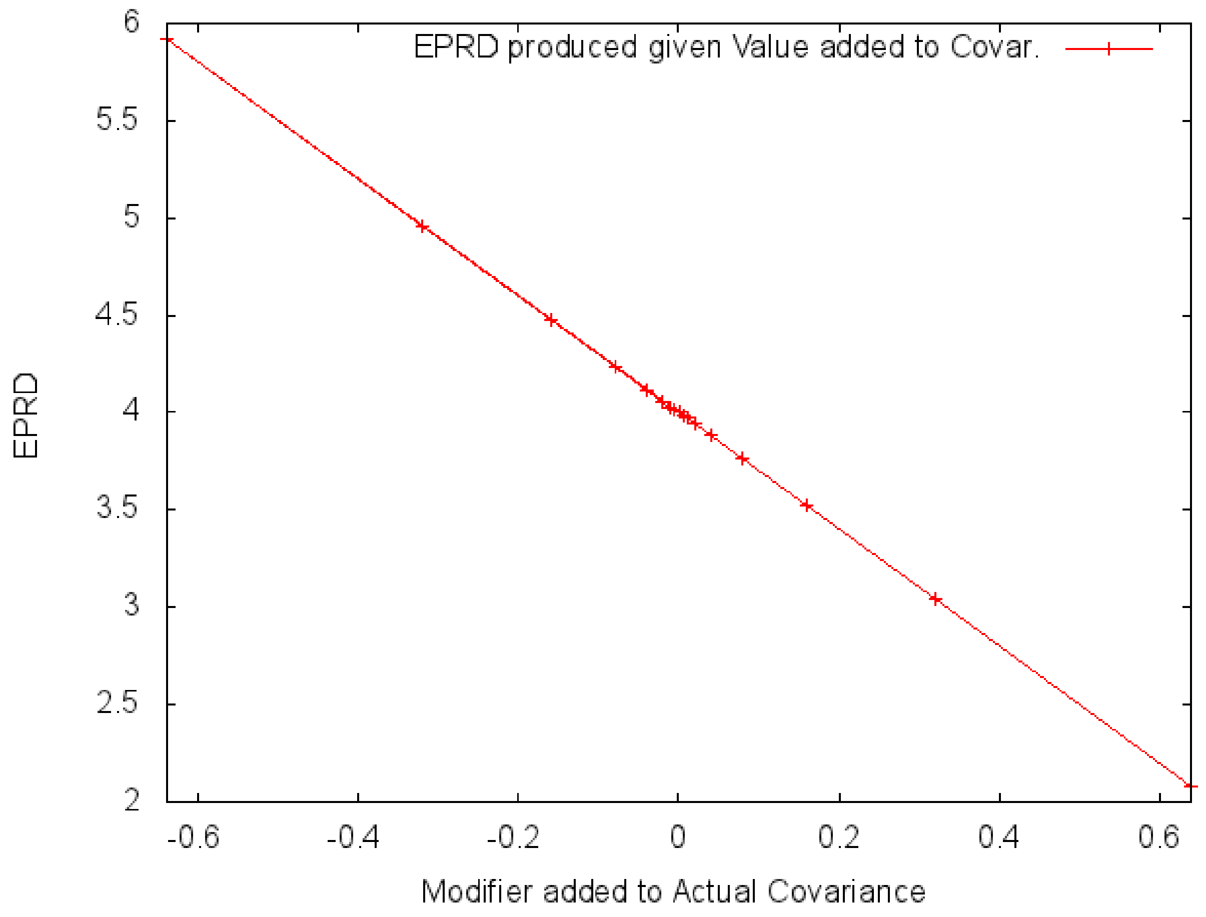
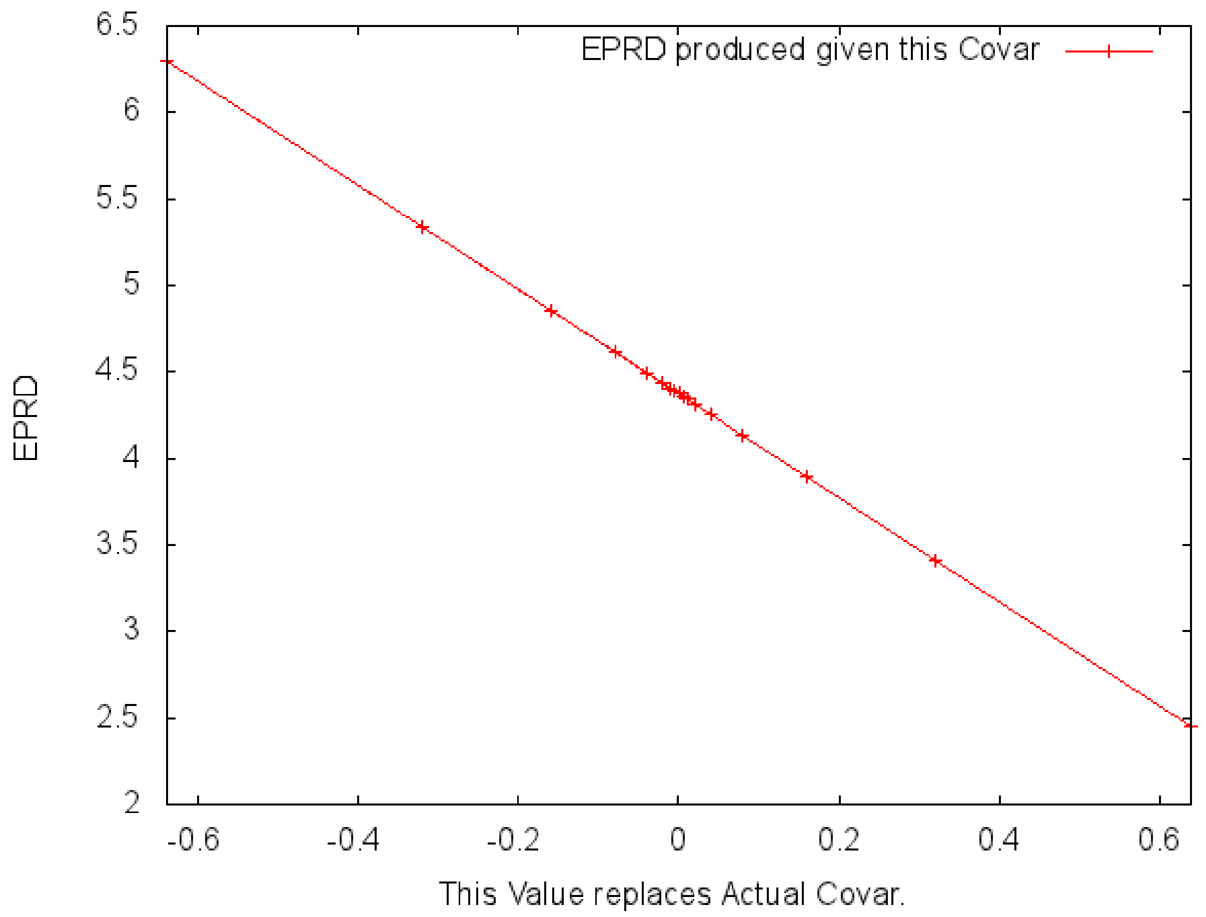


Figure 4. As the covariance σ value is replaced with increasing values on the x axis, the estimate of the EPRD decreases (improves). When the estimate of the covariance σ value is 0.125, then the EPRD is 4.



Bibliography

Appel, K., & Haken, W. (1977, October). Solution of the Four Color Map Problem. *Scientific American, 237*(4), 108-121.

Bahadur, R. R. (1961). A Representation of the Joint Distribution of Response of n Dichomotous Items. In H. Solomon, *Studies in Item Analysis and Prediction* (pp. 158-168). Stanford, CA: Stanford University Press.

Bendersky, M., & Croft, W. B. (2012). Modeling Higher-Order Term Dependencies in Information Retrieval using Query Hypergraphs. *ACM SIGIR Proceedings* (pp. 941-950). Portland, Oregon: ACM.

Bisht, R. K., Srivastava, G., & Dhami, H. S. (2010, July). Term Weighting Using Term Dependence. *International Journal of Computer Applications, 3*(11), 1-3.

Borko, H. (1985). Research in Computer Based Classifiication Systems. In L. M. Chan, P. A. Richmond, & E. Svenonius, *Theory of Subject Analysis: A Sourcebook* (pp. 287-305). Littleton, CO: Libraries Unlimited.

Cai, K., Bu, J., Chen, C., & Liu, K. (2007). Exploration of Term Dependence in Sentence Retrieval. *Proceedings of the ACL 2007 Demo and Poster Sessions* (pp. 97-100). Prague: Association for Computational Linguistics.

Church, L. (2010). *Combinatoric Models of Information Retrieval Ranking Methods and Performance Measures for Weakly Ordered Document Collections.* Chapel Hill, NC: Ph. D. Dissertation, U. of North Carolina.

Cooper, W. S. (1968). Expected Search Length: A Single Measure of Retrieval Effectiveness Based on Weak Ordering Action of Retrieval Systems. *Journal of the American Society for Information Science, 19*(1), 30-41.

Dang, E. K., Luk, R. W., & Allan, J. (2014). Beyond Bag-of-Words: Bigram-Enhanced Context-Dependent Term Weights. *Journal of the American Society for Information Science and Technology*, In Press.

Deerwester, S., Dumais, S. T., Furnas, G. W., Landauer, T. K., & Harshman, R. (1990, September). Indexing by Latent Semantic Analysis. *Journal of the American Society for Information Science, 41*, 391-407.

Eguchi, K., & Croft, W. B. (2009). Query Structuring and Expansion with Two-Stage Term Dependence for Japanese Web Retrieval. *Information Retrieval, 12*(3), 251-274.

Losee, R. M. (1994). Term Dependence: Truncating the Bahadur Lazarsfeld Expansion. *Information Processing & Mangement, 30*(2), 293-303.

Losee, R. M. (1998). *Text Retrieval and Filtering: Analytic Models of Performance.* Boston: Kluwer.

Losee, R. M., & Paris, L. A. (1999). Measuring Search Engine Quality and Query Difficulty. *Journal of the American Society for Information Science and Technology, 50*(10), 882-889.

Losee, R. M., Bookstein, A., & Yu, C. T. (1986). Probabilistic Models for Document Retrieval: A Comparison of Performance on Experimental and Synthetic Databases. *ACM Annual Conference on Research and Development in Information Retrieval* (pp. 258-264). Pisa, Italy: ACM.

Manning, C. D., Raghavan, P., & Schutze, H. (2008). *Introduction to Information Retrieval.* Cambridge: Cambrdige U. Press.

Manning, C., & Schutze, H. (1999). *Foundations of Statisitcal Natural Language Processing.* Cambridge, MA: MIT Press.

McCullagh, P. (2002). What is a Statistical Model? *The Annals of Statistics, 30*(5), 1225-1310.

Nanas, N., Vavalis, M., & De Roeck, A. (2010). A Network-Based Model for High-Dimensional Information Filtering. *SIGIR Proceedings* (pp. 202-209). Geneva, Switzerland: ACM.

Nickerson, R. S. (2010). *Mathematical Reasoning: Patterns, Problems, Conjectures, and Proofs.* New York, NY: Psychology Press, Taylor and Francis Group.

Park, J. H., Croft, W. B., & Smith, D. A. (2011). A Quasi-Synchronous Dependence Model for Information Retrieval. *ACM Conference on Knowledge and Information Management* (pp. 17-26). Glasgow, Scotland: ACM.

Shaw, W. M., Burgin, R., & Howell, P. (1997). Performance Standards and Evalution in IR Test Collections: Vector-space and Other Retrieval Models. *Information Processing & Management, 33*(1), 15-36.

Shiflet, A. B., & Shiflet, G. W. (2014). *Introduction to Computational Science: Modeling and Simulation for the Sciences* (Second ed.). Princeton, NJ: Princeton University Press.

Sloman, S. (2013). Counterfactuals and Causal Models. *Cognitive Science, 37*, 969-976. doi:10.1111/cogs/12064

Teugels, J. L. (1990). Some Represenations of the Multivariate Bernoulli and Binomial Distributions. *Journal of Multivariate Analysis, 32*, 256-268.

Yu, C. T., Buckley, C., Lam, K., & Salton, G. (1983). A Generalized Term Dependence Model in Information Retrieval. *Information Technology: Research and Development, 4*(2), 129-154.