ISSN 2229-6956(ONLINE)
DOI: 10.21917/ijsc.2010.0001

A DYNAMIC FEATURE SELECTION METHOD FOR DOCUMENT RANKING WITH RELEVANCE FEEDBACK APPROACH

K.Latha[1], B.Bhargavi[1], C.Dharani[1] and R.Rajaram[2]

[1]*Department of Computer Science and Engineering, Anna University of Technology, Tiruchirappalli, Tamil Nadu, India*
E-mail: erklatha@gmail.com

[2]*Department of Information Technology, Thiagarajar College of Engineering, Madurai, Tamil Nadu, India*
E-mail: rrajaram@tce.edu

Abstract

Ranking search results is essential for information retrieval and Web search. Search engines need to not only return highly relevant results, but also be fast to satisfy users. As a result, not all available features can be used for ranking, and in fact only a small percentage of these features can be used. Thus, it is crucial to have a feature selection mechanism that can find a subset of features that both meets latency requirements and achieves high relevance. In this paper we describe a 0/1 knapsack procedure for automatically selecting features to use within Generalization model for Document Ranking. We propose an approach for Relevance Feedback using Expectation Maximization method and evaluate the algorithm on the TREC Collection for describing classes of feedback textual information retrieval features. Experimental results, evaluated on standard TREC-9 part of the OHSUMED collections, show that our feature selection algorithm produces models that are either significantly more effective than, or equally effective as, models such as Markov Random Field model, Correlation Co-efficient and Count Difference method.

Keywords:

Feature Selection, Expectation Maximization, Markov Random Field, Generalization, Document Ranking

1. INTRODUCTION

In information retrieval, a ranking is used by search engines to rank matching documents according to their relevance to a given search query. One of the most important problems in information retrieval is determining the order of documents in the answer returned to the user. Once a search engine has identified a set of potentially relevant documents, it faces the task of determining which documents are most relevant, so that they may be listed first. This is typically done by assigning a numerical score to each document based on a ranking function, which incorporates features of the document, the query, and the overall document collection. Many methods and algorithms for document ordering have been proposed.

Feature selectors are algorithms that are applied to the data before it reaches Document Ranking program. Most feature selection [13] algorithms perform a search through the space of feature subsets. Since the size of this space is exponential in the number of features, an exhaustive search is usually intractable. Their aim is to reduce the dimensionality of the data by removing the irrelevant and redundant information; thus allowing the Ranking program to operate more effectively. The method introduced in this paper differs from them especially in that it uses an efficient feature selector 0/1 knapsack based heuristic with generalization and Relevance Feedback approach to determine the usefulness of features and evaluates its effectiveness with three common approaches such as Markov Random Field Model, Correlation Co-efficient and Count Difference.

The first methodology is Markov random field Model[5] which is undirected graphical model that provide a compact and flexible way of modeling joint distributions over a query $Q = q_1, \ldots, q_n$ and a document D. The underlying distribution over pairs of documents and queries is assumed to be a relevance distribution. This is a canonical method for describing textual information retrieval features. The representation makes it easy to generate large sets of features that can be used. That is, sampling from the distribution gives pairs of documents and queries, such that the document is relevant to the query.

The second methodology, the most familiar measure of dependence between two quantities is the Pearson product-moment correlation coefficient, or "Pearson's correlation." It is obtained by dividing the covariance of the two variables by the product of their standard deviations. Correlation based feature selection (CFS)[10] uses a search algorithm along with a function to evaluate the merit of feature subsets. The heuristic by which CFS measures the goodness of feature subsets takes into account the usefulness of individual features for predicting the class label along with the level of inter correlation among them.

The third methodology is Count Difference [14], which is based on the difference between the relative document frequencies of a feature for both relevant and irrelevant classes. Document frequency stands for the number of documents in which a feature occurs in a document collection. This method favors features whose document frequencies fall into a mid-range, since low-frequency features do not contribute much to the distinction of most documents and high-frequency features are so common that they reduce the distinction between documents.

Lastly but most importantly, the proposed approach uses an efficient and dynamic feature selector 0/1 knapsack based heuristic with generalization and Relevance Feedback approach. The knapsack problem is a problem in combinatorial optimization: Given a set of items, each with a weight and a value, determine the number of each item to include in a collection so that the total weight is less than a given limit and the total value is as large as possible. Here the Fall_Out ratio is the weight and the Information gain is the value for our consideration of feature selection. The features extracted from the knapsack [6] are the initial seeds for the Generalization module. Association rule induction is employed to capture feature co-occurrence patterns. Generalized features are constructed by applying these rules. Essentially, rules preserve implicit semantic relationships between features. Finally the retrieved features are ranked by NDCG (Normalized Discount

Cumulative Gain)[3] and the documents are ranked by the sophisticated ranking function Okapi BM25 Model. Expectation Maximization algorithm is used for Relevance feedback [2] and the documents are again ranked by Okapi BM25 Model.

The remainder of this paper is laid out as follows. Section 2.1 provides background material on the MRF model for IR. In Section 2.2 we explain Correlation Co-efficient method for representing features. Next, Section 2.3 describes Count Difference. Section 3 demonstrates our proposed feature selection approach. Then, in Section 4 and 5 we formally evaluate various aspects of our proposed algorithm and compare the effectiveness of the learned models to several other retrieval models. Finally, in Section 6 we summarize our contributions and outline potential directions for future work.

2. EXISTING METHODS

2.1 MARKOV RANDOM FIELD MODEL FOR FEATURE SELECTION

A MRF[4] is defined by a graph G and a set of non-negative potential functions over the cliques in G. The nodes in the graph represent the random variables and the edges define the independence semantics of the distribution. A MRF satisfies the Markov property, which states that a node is independent of all of its non-neighboring nodes given observed values for its neighbors. Since most information retrieval models are concerned with ranking documents in response to a query, MRF focus on textual features defined over query/document pairs.

Each feature is represented using a 3-tuple of the form (dependence model type, clique set type, weighting function).The input to the feature induction algorithm will be a set of 3-tuples, each of which represents a single feature. In this work, the focus is on three dependence model types. The three types are full independence (FI), sequential dependence (SD), and full dependence (FD). The second entry in the tuple, the clique set type, describes the set of cliques within the graph that the feature is to be applied to. Each feature is applied to one or more cliques within the graph. If it is applied to more than one clique, then all of the cliques that share the feature also share the weight for that feature.

The set of cliques available for this model are single term, ordered terms and unordered terms. However, if the clique set contains one or more cliques, then the feature value is the sum of the feature weights for each clique in the set. For example, given the query new york city, using the full independence model and the single term clique set, we would obtain a feature value of f(new, D)+f(york, D)+ f(city, D). Therefore, clique sets act to anonymize query terms. In this way, clique sets can be thought of as feature types.

Finally, the third entry in the tuple is the weighting function, which describes how the feature values are computed. Weighting function (The Popular BM25) that can be used with clique sets. It is straightforward to use the standard forms of these weighting functions for the single term clique set. By using an automatic feature selection algorithm, noisy or redundant features in a large feature set can be reduced. Such features may reduce training efficiency and may result in a model that contains a number of non-identifiable parameters. Non-identifiable parameters are those that cannot be reasonably estimated given the training data. Finally, feature selection can provide insights into the important features for a given task or data set. Let M_t denote the model learned after iteration t. Features are denoted by f and the weight (parameter) associated with feature f is denoted by λ_f. The candidate set of features is denoted by F. The entire set of feature weights for a model is denoted by Λ. A model, then, is represented as set of feature/weight pairs. Finally, we assume that SCORE(M) returns the utility or 'goodness' of model M with respect to some training data. The utility function and the form of the training data largely depend on the underlying task. The important thing to note here is that any utility function, regardless of whether or not it is differentiable with respect to the model parameters, can be used. Therefore, the ultimate goal of our feature selection algorithm is to select features and set feature weights in such a manner as to maximize the metric imposed by SCORE(·).

The algorithm begins with an empty model (i.e., $M_0 = \{\}$). Then, temporarily add a feature f to the model. Then hold all weights except λ_f fixed and find the setting for λ_f that maximizes the utility of the augmented model. The utility of feature f ($SCORE_f$) is defined to be the maximum utility obtained during training. The feature's utility measures how good the current model would be if the feature were added to it. This process is repeated for every f Є F, resulting in a utility being computed for every feature in the candidate pool. The feature with the maximum utility is then added to the model and removed from F. After the new feature is added, retrain the entire set of weights (optional). The entire process is then repeated until either some fixed number of features has been added to the model or until the change in utility between consecutive iterations drops below some threshold. But this algorithm is not guaranteed to find the global maximum for SCORE(M). Instead, it is guaranteed to find local maxima. Many factors, including properties of SCORE(M), the number of features used, and the properties of the feature used, will affect the quality of the learned model.

Under this parameterization, documents are ranked in descending order according to P(D|Q), which can be shown to be rank equivalent to:

$$P(D \mid Q) \overset{rank}{=} \sum_{c \in C(G)} \lambda_c f_c(c) \qquad (1)$$

Where C(G) is the set of cliques in G and λ_c is the weight (parameter) associated with clique c. The automatic feature selection algorithm associates new feature functions and weights (parameters) with cliques in G, which results in new $\lambda_f(\cdot)$ components being added to the ranking function.

2.2 CORRELATION-BASED FEATURE SELECTION

Correlation [12] measures are also known as dependency measures or similarity measures. They measure the ability to predict the value of one variable from the value of another. The feature subset selector used here is of Filter type. A feature is arbitrary chosen as the start state and greedy hill climbing search is used to expand the current node and moves to the child with the highest evaluation. The child with highest evaluation is one that is highly correlated (predictive of) that class and

uncorrelated with the parent nodes. By and large most classification tasks in ML involve learning to distinguish between nominal class values, but may involve features that are ordinal or continuous.

The features may be ordinal or continuous. A measure based on conditional entropy is used to measure correlations between features and the class, and between features. Continuous features are first converted to nominal by binning. If X and Y are discrete random variables with respective ranges R_x and R_y, Equations 3 and 4 give the entropy of Y before and after observing X.

$$H(Y) = -\sum_{y \in R_y} p(y)\log(p(y)) \quad (2)$$

$$H(Y \mid X) = -\sum_{x \in R_x} p(x) \sum_{y \in R_y} p(y \mid x)\log(p(y \mid x)) \quad (3)$$

Equation 4 gives a measure of correlation or dependency of Y on X. This measure is sometimes called the uncertainty coefficient of Y.

$$C(Y \mid X) = \frac{H(Y) - H(Y \mid X)}{H(Y)} \quad (4)$$

This measure lies between 0 and 1. A value of 0 indicates that *X* and *Y* have no association; the value 1 indicates that knowledge of *X* completely predicts *Y*.

Correlation-based Feature Selection uses a search algorithm along with a function to evaluate the merit of feature subsets. Good feature subsets contain features highly correlated (predictive of) with the class, yet uncorrelated with (not predictive of) each other.

$$Gs = \frac{k\overline{rci}}{\sqrt{k + k(k-1)\overline{rii}}} \quad (5)$$

k is the number of features in the subset; r_{ci} is the mean feature correlation [12] with the class, and $r_{ii'}$ is the average feature inter-correlation. It is Pearson's correlation, where all variables have been standardized. The numerator can be thought of as giving an indication of how predictive of the class a group of features are; the denominator of how much redundancy there is among them. The heuristic goodness measure should filter out irrelevant features as they will be poor predictors of the class. Redundant features should be ignored as they will be highly correlated with one or more of the other features.

2.3 COUNT DIFFERENCE BASED FEATURE SELECTION

Term Discrimination tries to measure the ability of a feature for distinguishing one document from the others in a collection. A very popular feature often has a negative discrimination value, since it tends to reduce the differences between documents, while a rare feature usually has a close-to-zero value, since it is not significant enough to affect the space density. Related to this concept, a new feature selection method called Count Difference (CD), which is based on the difference between the relative document frequencies of a feature for both relevant and irrelevant classes.

A feature whose document frequency for one class is higher than that for the other class is desirable than the rare feature in the training collection, since it helps distinguishing between the two classes. So this method favors features whose document frequencies fall into a mid-range, since low-frequency features do not contribute much to the distinction of most documents and high-frequency features are so common that they reduce the distinction between documents.

Given a feature, the set of training documents are partitioned into four regions such as relevant & feature present, relevant & feature absent, irrelevant & feature present and irrelevant & feature absent. The relative document frequency, which is the ratio of the document frequency of a feature for one class over the average document frequency for the same class:

$$relativDF(t, y) = at / \bar{a} \quad (6)$$

$$relativDF(t, y1) = bt / \bar{b} \quad (7)$$

Here $\bar{a}$ and $\bar{b}$ denote the average document frequencies for the relevant and irrelevant classes, which are computed as follows:

$$\bar{a} = \frac{1}{M}\sum_{t=1}^{M} at \qquad \bar{b} = \frac{1}{M}\sum_{t=1}^{M} bt \quad (8)$$

where M is the number of original features before the selection process. With the relative document frequencies, the count difference score of a feature is defined as the difference between its two relative document frequencies:

$$CD(t) = \left(at/\bar{a} - bt/\bar{b}\right)^2 \quad (9)$$

Intuitively, the relative document frequency measures the importance of a feature against the average feature for one class. If a feature is rare, its relative document frequency will be low, whereas if a feature is popular, its relative document frequency will be high. The count difference tends to favor features whose relative document frequencies for one class are higher than those for the other class. If a feature is popular for both classes, its count difference score will be reduced.

3. PROPOSED METHOD

In this section we present a new methodology for enhancing existing IR systems with the goodness of the features. We begin with the n-gram architecture, and then focus on its core, the feature selection module. An n-gram is a subsequence of n-items in any given sequence. In computational linguistics n-gram models are used most commonly in predicting words (in word level n-gram) or predicting characters (in character level n-gram) for the purpose of various applications. To analyze the efficiency of this methodology we used the on-line medical information database. Normally n-grams of length 2 or 3 are most useful for feature selection. Using gram lengths more than 3 reduces the performance of feature selection.

3.1 FEATURE SELECTION WITH DYNAMIC 0/1 KNAPSACK

Greedy approach and Dynamic Programming are the two ways to solve Optimization problems. Most of the problems can be solved by both Greedy and Dynamic Programming. But it is more difficult to determine whether a Greedy algorithm always

produce an optimal solution. But in the case of Dynamic programming [8] we need only to determine whether the principle of optimality applies. We show the principle of optimality by solving the 0/1 Knapsack by Dynamic Programming. This is used for optimization problems, where we want to find the 'best way' of doing something. This often produces a polynomial time for finding the optimal solution when brute force enumeration of possibilities would be exponential

Pseudo code

Dynamic 0/1 knapsack (n, W, v1, …, v_n, w_1, …, w_n)

1. for w from 0 to W, set V[0, w] = 0
2. for k from 1 to n
3. set V[k, 0] = 0
4. for w from 1 to W
5. if w_k > w
6. then set V[k, w] = V[k -1, w]
7. else
8. if V[k -1, w] > v_k + V[k -1, w - w_k]
9. then set V[k, w] = V[k -1, w]
10. else set V[k, w] = v_k + V[k -1, w - w_k]

Given: a set of Features F = {f_1, f_2, …, f_n} of n features where each f_i has Information Gain(IG) value v_i and Fall_Out as weight w_i .

Required: to choose a subset O of F such that the total weight (Fall_Out) of the items chosen does not exceed W and the sum of the values v_i of items in O is maximal.

A feature's discriminatory power is a useful gauge of its goodness and is commonly ascertained using the Information Gain (IG) score as

$$IG(X,Y) = \sum_{x\in 0,1}\sum_{y\in 0,1} P(X=x,Y=y).\log_2 \frac{P(X=x,Y=y)}{P(X=x).P(Y=y)} \quad (10)$$

Fall_Out is the ratio of the retrieved non relevant documents and the total number of non relevant documents in the collection and is given by

$$fall\_out = \frac{\left|\{non-relevant\ documents\} \cap \{retrieved\ documents\}\right|}{\left|\{non-relevant\ documents\}\right|} \quad (11)$$

Suppose the optimal solution for F and W is a subset O in which f_k is the highest numbered item. Then O-{f_k} is an optimal solution for F_{k-1} = {f_1, …, f_{k-1}}and total weight W-w_k. And the value of the global solution O is v_k plus the value of the sub problem solution. Given a target weight w and a set F_k = {f_1, …, f_k }imagine examining all the subsets of F_k whose total weight is £ w. Some of these subsets might have bigger total values than others. Let V[k, w] be the biggest total value of such a subset of F_k. Now we give a recursive definition of V[k, w].V[k, w] = 0 if either k = 0 or w = 0, otherwise V[k, w] = if w_k > w then V[k -1, w] else max{V[k -1, w], v_k + V[k -1, w - w_k] }.The recursive definition of V[k, w] says that the value of a solution for stage F_k and target weight w either includes item f_k, in which case it is v_k plus a sub problem solution for F_{k-1} and total weight w-w_k, or doesn't include f_k , in which case it is just a sub problem solution for F_{k-1} and the same weight w.

3.2 FEATURE GENERALIZATION WITH ASSOCIATION RULE INDUCTION

The features that are extracted from the 0/1 knapsack Algorithm are the initial seeds for the Association rule Algorithm [1]. Apriori is a well known association rule induction algorithm introduced for the market-basket analysis domain where one wishes to find regularities in people's shopping behavior. It generates rules of the form H ← B, where the rule body B is a conjunction of items, and the rule head H is a single item. To select interesting rules from the set of all possible rules, constraints on various measures of significance and interest can be used. The best-known constraints are minimum thresholds on support and confidence. The support supp(*X*) of an item set *X* is defined as the proportion of transactions in the data set which contain the item set. The confidence of a rule is defined conf(X➔Y) = supp (XUY)/supp(X).

Association rules are discovered in two stages. Firstly Apriori identifies sets of items that frequently co-occur, i.e. above a given minimum threshold. It then generates rules from these item sets ensuring frequency and accuracy are above minimum thresholds. This means that rules can be used to predict the presence of the head feature given that all the features in the body are present in the document. This means that a case satisfying the body even when the head feature is absent will not be considered.

An informed rule selection strategy is necessary because Apriori typically will generate many rules. The rule selection strategies are the coverage, accuracy and information gain for each generated rule. Generally the first two measures are used by Apriori during rule generation to prune the search space. Here coverage (or frequency) is the percentage of documents in which a rule is applicable; and confidence (or accuracy) is the proportion of documents in which the rule prediction is correct. The third measures the gain in information due to the rule's body, and indicates how well the body is able to predict the presence or absence of the head feature.

3.3 NORMALIZED DISCOUNT CUMULATIVE GAIN (NDCG) FOR FEATURE RANKING

DCG (discounted cumulative gain) is a measure of usefulness, or gain, of a document based on its position in the result list. The gain is accumulated cumulatively from the top of the result list to the bottom with the gain of each result discounted at lower ranks. DCG originates from an earlier, more primitive, measure called Cumulative Gain which is the sum of the graded relevance values of all results in a search result list.

NDCG is designed for measuring ranking accuracies when there are multiple levels of relevance judgment, so the cumulative gain at each position for a chosen value of *p* should be normalized across queries. This is done by sorting documents of a result list by relevance, producing an ideal CG at position n. For features for which the number of retrieved documents is less than n, NDCG is only calculated for the retrieved documents. In

evaluation, NDCG is further averaged over all features. Refer equation (16).

3.4 DOCUMENT RANKING USING BM25 WEIGHTING FUNCTION

Finally the documents are ranked by the BM25 weighting Function. A ranking is a relationship between a set of items such that, for any two items, the first is either 'ranked higher than', 'ranked lower than' or 'ranked equal to' the second.

BM25 is a bag-of-words retrieval function that ranks a set of documents based on the query terms appearing in each document, regardless of the inter-relationship between the query terms within a document (e.g., their relative proximity). It is not a single function, but actually a whole family of scoring functions, with slightly different components and parameters. Okapi BM25 is a ranking function used by search engines to rank matching documents according to their relevance to a given search query. It is based on the probabilistic retrieval framework and is presented by the equation

$$fT, BM25(qi, D) = \frac{(k1+1)tfw; D}{k1((1-b)+b(|D|/|D|avg))+tfw; D} \log \frac{N-dfw+0.5}{dfw+0.5} \quad (12)$$

Where $tf_{w,D}$ is the number of times the word w matches in document D, df_w is the total number of documents that have at least one match for word w, |D| is the length of document D, $|D|_{avg}$ is the average document length, N is the number of documents in the collection and b is the weighting function hyper parameter.

The top ranked documents are given as Relevance Feedback documents to Expectation Maximization.

3.5 EXPECTATION MAXIMIZATION FOR RELEVANCE FEEDBACK

Personalization in full text retrieval or full text filtering implies reweighing of the query terms based on some explicit or implicit feedback from the user. Relevance feedback [11] inputs the user's judgments on previously retrieved documents to construct a personalized query or user profile. A standard procedure for estimating probabilities of unknown parameters from incomplete data is the expectation maximization algorithm (EM algorithm).

EM is particularly useful when the likelihood is an exponential family: the E-step becomes the sum of expectations of sufficient statistics, and the M-step involves maximizing a linear function. In such a case, it is usually possible to derive closed form updates for each step. An EM algorithm can be easily modified to find the maximum a posteriori (MAP) estimates for Bayesian inference. The algorithm iteratively maximizes the probability of the query t1, t2,..., tn given R relevant documents d1, d2,..., dR. The resulting EM-algorithm is defined as follows

$$E\_step: ri' = \sum_{j=1}^{R} \frac{\lambda i^{(p)} P(Ti = ti \mid Dj = dj)}{(1-\lambda i^{(p)}) P(Ti = ti) + \lambda i^{(p)} P(Ti = ti \mid Dj = dj)} \quad (13)$$

$$M\_step: \lambda i^{(p+1)} = \frac{ri'}{R} \quad (14)$$

Speaking of an expectation (E) step is a bit of a misnomer. What is calculated in the first step are the fixed, data-dependent parameters of the function *Q*. Once the parameters of *Q* are known, it is fully determined and is maximized in the second (M) step of an EM algorithm.The expectation step calculates the expected number of documents in which t_i is important. The maximization step simply involves a maximum likelihood estimate. The EM-algorithm should be used by initializing the relevance weights to some initial value, e.g. $\lambda_i^{(0)} = 0.5$ and then iterate through the E-step and M-step until the value of λ_i does not change significantly anymore (p denotes the iteration number). λ_i is an unknown parameter, denoting the probability that the term on position i in the query is important. So, for $\lambda_i = 0$ the term is definitely unimportant, whereas for $\lambda_i = 1$ the term is definitely important. Based on these new features the test documents are ranked again by the BM25 weighting function.

4. EVALUATION

Feature selection and generalization techniques enable organizing and ranking the documents. The features retrieval performance using test set accuracy is used to compare the above algorithms with our proposed method. The dynamic 0/1 Knapsack algorithm [7] is used for initially selecting the features which differs from Greedy approaches that have been used by the other Algorithms. This is because of a recursive approach that involves breaking a global problem down into more local sub problems and assumes an optimal substructure i.e. There is a simple way to combine optimal solutions of sub problems to get an optimal global solution. And also this avoids the inefficiency that straightforward recursion may suffer from sub problem overlap (i.e. when decomposition results in the same sub problems occurring often and being solved many times).The feature selection on its own has not improved accuracy and ranking, but feature selection combined with generalization is significantly better than all other algorithms. We introduced a relevance feedback algorithm for the identification of feedback documents that shows improved performance compared to the other methods.

4.1 DATASET USED FOR FEATURE SELECTION

Dataset [9] used in our experiment is the OHSUMED data, which was used in many experiments in information retrieval, including the TREC-9 filtering track. OHSUMED test collection is a set of 348,566 references from MEDLINE, the on-line medical information database, consisting of titles and/or abstracts from 270 medical journals over a period 1987, developed by the Oregon Health Sciences University.

4.2 N-GRAM GENERATION

The training text of size is 12,363 features were constructed by evenly combining text blocks from the OHSUMED data corpus. Comprehensive ngram statistics were automatically generated and stored for training text. The experimental conditions were established by bigrams with the training texts.

4.3 EVALUATION MEASURES

4.3.1 Mean Average Precision:

Precision and recall are single-value metrics based on the whole list of documents returned by the system. For systems that return a ranked sequence of documents, it is desirable to also consider the order in which the returned documents are presented. Average precision emphasizes ranking relevant documents higher. It is the average of precisions computed at the point of each of the relevant documents in the ranked sequence:

$$Avep = \frac{\sum_{r=1}^{N}(P(r)\times rel(r))}{number\ of\ relevant\ documents} \qquad (15)$$

where r is the rank, N the number retrieved, $rel()$ a binary function on the relevance of a given rank, and $P()$ precision at a given cut-off rank.

4.3.2 Normalized Discount Cumulative Gain:

NDCG is designed for measuring ranking accuracies when there are multiple levels of relevance judgment. Given a query, NDCG at position n in is defined as

$$N(n) = Zn\sum_{j=1}^{n}\frac{2^{R(j)}-1}{\log(1+j)} \qquad (16)$$

where n denotes position, $R(j)$ denotes score for rank j, and Zn is a normalization factor to guarantee that a perfect ranking's NDCG at position n equals 1.

4.3.3 Kendall Tau Distance:

The Kendall tau distance is a metric that counts the number of pair wise disagreements between two lists. The larger the distance, the more dissimilar the two lists are. $K(\tau_1,\tau_2)$ will be equal to 0 if the two lists are identical and $n(n-1)/2$ (where n is the list size) if one list is the reverse of the other. Often Kendall tau distance is normalized by dividing by $n(n-1)/2$ so a value of 1 indicates maximum disagreement. The normalized Kendall tau distance therefore lies in the interval [0,1].Kendall tau distance may be defined as

$$K(\tau_1,\tau_2) = \sum_{\{i,j\}\in P}\overline{K}i,j(\tau_1,\tau_2) \qquad (17)$$

5. EMPIRICAL RESULTS

Table 1 summarizes the relative performance results achieved on the test set, by the existing feature selection systems and our dynamic selection system evaluated with mean average precision, Normalized Discount Cumulative Gain, Kendall tau distance as evaluation metrics. The larger the tau distances then the selected features are very distinct. Similarly higher the mean average precision values then the retrieved performance too is higher. Count Difference has low Kendall value, MAP and NDCG values when compared to all other methods. This is due to following reasons. Count Difference focuses much on the relative relevant document frequency and relative irrelevant document frequency and not on the similarity between the features while selecting them. If feature's relevant document frequency is higher than the irrelevant, then that feature is present more in relevant documents than in irrelevant ones. This also means that the feature is present in both the classes, leads to feature diversion. Since the features selected are present in both the classes, the order of the documents does not suit much for ranking. Moreover the features selected are not much distinct. This leads to low MAP (62%), Kendall tau distance value (92%) and NDCG (82%) values than all the other three methods. Our proposed method obtained 74% of MAP, 84% of NDCG and 96% of Kendall values. Thus the optimal solution of Dynamic feature selection algorithm, best association feature analysis of Association rule, and Expectation Maximization performance of our proposed approach produces the maximum MAP, NDCG and Kendall tau values and performs better than the other three existing methods.

The time complexity of MRF method is O(n). As the ultimate goal of MRF is to maximize the MAP of the model, it checks whether the combination of features maximize the MAP or not. So the computational time depends linearly on the number of features(n). The correlation method requires $m((n^2-n)/2)$ operations for computing the pair wise feature correlation matrix. As per equation (5) for a feature subset containing n features, n additions are required in the numerator (feature-class correlations) and $(n^2-n)/2$ additions are required in the denominator (feature-feature inter-correlations). Thus the time complexity $O(mn^2/2)$ where m is the number of documents and n is the number of features. Count Difference method requires O(mn) time as it purely depends on relative document frequency. It is a ratio of document frequency over the average document frequency(ADF). ADF is an average value of n feature's document frequency calculated in m documents. Our proposed approach involves Knapsack algorithm, Apriori algorithm and Expectation Maximization. The time complexity of Knapsack is O(nW) as the aim of the algorithm is to select features from n features till the total IG value is less than or equal to the give limit W. The major determining parameter for the complexity of the Apriori algorithm is $C=\sum_k x_k k$ where $x_k = |C_k|$. We know that $m_1 = n$ as one needs to consider all single items. Furthermore, one would not have any features which alone are not frequent and so one has $m_2= n(n-1)/2$. Here, we stop with 2-item candidate set and also including the dependence on the number of documents (i.e m) the computational time is $O(mn^2)$. The e-step of EM algorithm calculates the value of r simply using the term frequency and document frequency of all features in m documents. The m-step maximizes the value of unknown parameter. These two steps are repeated for all n features. So e-step requires O(mn) computation time and m-step has a time complexity of O(n). So the time complexity of our proposed approach is $O(mn^2)$.

Table 2 shows increment in MAP values for all systems when the number of features increases. Large number of features will give more information to retrieve more relevant documents. This means that when the documents are ranked with more number of features the document getting higher BM25 value (i.e. containing all the terms that the user needs) is more relevant and it's contribution is higher which leads to high MAP values. Therefore the increases in number of features result in increase of MAP values for all methods.

Table 3 shows increment in MAP values for our proposed system with the increment in the number of feedback documents. Feedback avoids the risk of bringing the unwanted terms into the ranking model. Important terms from the feedback documents have potential of retrieving more relevant documents.

Moreover here NDCG is used to evaluate the selected features which results in features that are capable of presenting the relevant documents at the top itself from which the feedback (i.e. relevant to user) documents are chosen. Again from this user judged documents, features are selected using EM algorithm which focuses mainly on term frequency. So when the feedback documents are increased the term having higher term frequency will be selected in EM. Ranking again with these features leads to higher MAP values such as 80%.

Table 4 presents the difference in the performance of our proposed method with and without considering the feedback documents and features. Feedback documents are nothing but user judged documents. Selecting features from user judged (i.e. feedback) documents and ranking again the test documents with these features will be more relevant and the order of relevancy suits more BM25 ranking which leads to higher MAP values than ranking without considering the feedback documents.

Table.1 Performance evaluation of Feature selection techniques

Methods	NDCG	MAP	Kendall Tau dist	Time Complexity
M RF	0.8406	0.7195	0.9560	O(n)
Correlation	0.8346	0.6286	0.9534	O(mn^2)
Count difference	0.8222	0.6251	0.9190	O(mn)
Proposed	0.8464	0.7413	0.9685	O(mn^2)

Table.2 Statistics of feature selection

Features	MRF	Correlation	Count difference	Pro-posed
3	0.5666	0.5731	0.5121	0.5793
7	0.6298	0.5946	0.5501	0.6623
9	0.6724	0.5972	0.5623	0.6739
11	0.6985	0.5987	0.5873	0.7088
13	0.7195	0.6286	0.6251	0.7413

Table.3 Statistics of Feedback documents

Feedback Docs	MAP
3	0.5556
5	0.6287
7	0.6379
9	0.7736
11	0.8009

Table.4 Feature selection method with/without feedback

Methods	Feed back	Without Feed back
Proposed	0.7413	0.7013

Fig.1 shows the MAP values achieved by all of the four feature selection methods on test set, and proves that our proposed approach outperforms other three existing methods. All the four methods show decrement in the MAP value with the increment in the number of documents. All the four methods present the relevant documents at the top itself which results in large MAP values when the numbers of documents are less in number as MAP emphasizes ranking relevant documents higher. Then when the number of documents goes on increasing, the number of relevant documents may increase, but the position of the relevant documents does not change which results in low MAP values as MAP is the average of precisions computed at the point of each of the relevant documents in the ranked sequence. Moreover increase in documents might result in more irrelevant documents than relevant ones, which thereby decreases MAP value.

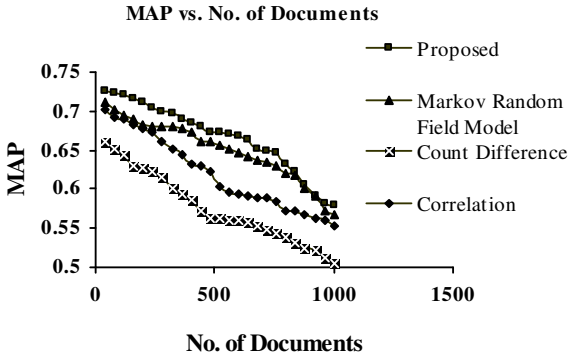

Fig.1. MAP vs. Number of Documents

6. CONCLUSIONS AND FUTURE WORK

In this paper we present a novel feature selection technique which is a combined form of dynamic feature selection algorithm and association rules. It uses NDCG which is a popular metric to measure the usefulness of the ranking algorithm based on relevance feedback. It also incorporates relevance feedback concept using Expectation Maximization methodology which suits most for the BM25 ranking algorithm. Thus the results obtained from the experiments show that our feature selection technique using the 0/1 Knapsack algorithm can efficiently provide solution and evidently improve the performance of information retrieval as well or better than other feature selection systems. It would be interesting to combine our feature selection approach with the Concept and Context based environment. Relevance feedback can be used to automatically induce new features that can then be automatically selected into a model using our algorithm. But it would also be interesting to apply a semi –automatic (user and System) relevance feedback approach for retrieval which yield additional important features into the system. There are two main problems in Relevance Feedback. First, if none of the top N documents are relevant, we may still find relevant documents in top ranked documents, which is more inclusive but usually unreachable when people are doing relevance feedback in interactive ad-hoc search, from which we can draw feedback terms. Second, the top N documents may be related to the topic, but nonetheless irrelevant. In this case, we may still extract useful terms from these documents, even if they do not qualify as relevant ones. An efficient Relevance feedback technique can be used to address the above problems. For many optimization problems a solution

using dynamic programming can be costly. There are two key attributes that a problem must have in order for dynamic programming to be applicable: optimal substructure and overlapping subproblems. A generalized algorithm can be used for rectifying the above defined problems.

REFERENCES

[1] Nirmalie Wiratunga, Ivan Koychev, and Stewart Massie, 2004, "Feature Selection and Generalisation for Retrieval of Textual Cases," in Proceedings of the 7th European Conference on Case-Based Reasoning, Springer, Madrid, Spain, pp. 806.

[2] Matthew Lease, 2008, "Incorporating Relevance and Psuedo-relevance Feedback in the Markov Random Field Model," in proceedings of the 17th Text REtrieval Conference, pp. 500.

[3] Xiubo Geng, Tie-Yan Liu, Tao Qin, and Hang Li, 2007, "Feature Selection for Ranking," in Proceedings of the 30th annual International Conference on Research and development in Information Retrieval, pp. 407.

[4] Donald Metzler, and William Bruce Croft, 2005, "A Markov random field model for term dependencies," in Proc. 28th Ann. Intl. ACM SIGIR Conf. on Research and Development in Information Retrieval, pp. 472.

[5] Donald Metzler, and William Bruce Croft, 2006. Linear feature based models for information retrieval. Journal of Information Retrieval (10): pp.257-274.

[6] Antonio Lopez Jaimes, Carlos Artemio Coello Coello, and Debrup Chakraborty, 2008, "Objective Reduction using a Feature Selection Technique," in proceedings of the 10th annual conference on Genetic and Evolutionary Computation, pp. 673.

[7] Shinn-Ying Ho, Jian-Hung Chen, and Meng-Hsun Huang. 2004. Inheritable genetic Algorithm for Biobjective 0/1 Combinatorial Optimization Problems and its Applications. Systems, Man and Cybernetics Part B (34): pp. 609-620.

[8] Ellis Horowitz, Sartaj Sahni, and Sanguthevar Rajasekaran. 2007. Computer Algorithms/C++, 2nd Edition, Universities Press.

[9] The TREC Conference, National Institute of Standards & Technology, U.S. Department of Commerce, Webpage: http://trec.nist.gov/data/t9_filtering/oshu-trec.tar.gz.

[10] Mark Andrew Hall, 2000, "Correlation-based Feature Selection for Discrete and Numeric Class Machine Learning," in Proceedings of the Seventeenth International Conference on Machine Learning, pp. 359.

[11] Tao Tao, and ChengXiang Zhai, 2006, "Regularized estimation of mixture models for robust pseudo-relevance feedback," in Proc of the 29th annual international ACM SIGIR conf on Research and development in information retrieval, pp. 162.

[12] Michael Edward Houle, and Nizar Grira, 2007, "A Correlation-Based Model for Unsupervised Feature Selection," in Proc of the 16th ACM SIGIR conf. on information and knowledge management, pp. 897.

[13] Roberto Ruiz, Jesus Aguilar-Ruiz, and Jose Cristobal Riquelme Santos. 2004. Wrapper for Ranking Feature Selection. Intelligent Data Engineering and Automated Learning-Lecture Notes, volume 3177/2004: pp. 384-389

[14] Edda Leopold and Jorg Kindermann. 2002. Text categorization with support vector machines. How to represent texts in input space? Machine Learning (46): pp. 423–444.

[15] Hui Fang, and ChengXiang Zhai, 2006, "Semantic term matching in axiomatic approaches to information retrieval," in Proc. 29th Ann. Intl. ACM SIGIR Conf. on Research and Development in Information Retrieval, pp.115.