Adaptive Feedback Methods in an Extended Boolean Model[1]

Jongpill Choi[+], Minkoo Kim[+] and Vijay V. Raghavan[#]

[+](cjp, minkoo)@madang.ajou.ac.kr
Dept. of Computer Engineering
Ajou University, Suwon, Korea

[#]raghavan@cacs.louisiana.edu
The Center for Advanced Computer Studies
University of Louisiana at Lafayette, Lafayette, LA 70504, USA

Abstract

Relevance feedback methods have been used in information retrieval to generate improved query formulations based on information contained in previously retrieved documents. The relevance feedback techniques have been applied to extended Boolean query formulations as well as to vector query formulations. In this paper, we propose an adaptive way to improve the retrieval performance in an extended Boolean model. We develop a neural network model in which the weights used in extended Boolean queries can be adjusted by users' relevance feedback. Experiments are performed on a TREC collection and the results show improved performance even after applying the previous feedback methods.

1 Introduction

Many relevance feedback methods have been studied [3, 7, 8, 9]. The main idea of relevance feedback is to improve the retrieval performance to reformulate the queries based on users' judgments of relevance of the retrieved documents. They are mostly based on vector model for information retrieval. As an alternative to the conventional Boolean model, the vector model is developed to represent documents and queries as vectors without using Boolean operators. However, since the structure provided by the Boolean operators is absent, the retrieval methodology in vector model is not compatible with the conventional Boolean method. To apply the advantages of the vector model to the Boolean method, the Boolean model has been extended [2, 12, 13], and also the relevance feedback method for the extended Boolean model has been proposed [11].

In the relevance feedback method for the extended Boolean model, an initial Boolean query is reformulated in a DNF (Disjunctive Normal Form) formula based on users' relevance feedbacks. Not only terms in a clause in the DNF formula but also the clause itself are weighted. In this study, we find that the retrieval performance of the reformulated queries can be further improved by adjusting the weight values with the same users' relevance feedbacks. To adjust the weight values of queries in DNF, we extend the idea of neural network leaning method that was proposed by Kim and Raghavan [4], and integrate it with the p-norm based extended Boolean model. In Kim and Raghavan's approach, they nicely mapped the p-norm based extended Boolean approach into a well known neural network model, called the multi-layered perceptron [6]. This approach is different from the previous neural network approaches for information retrieval [14, 16] in at least the following two aspects. In the previous approaches, they were mainly concerned with term associations. In this approach, they handle relations between concepts and Boolean expressions in which weighted terms are involved. Secondly, in the most of the previous approaches, they introduced their own network model in order to map the retrieval problems into the neural network domain. In this approach, they use an already proven neural network model in terms of its performance.

In the rest of this paper, we describe the p-norm based extended Boolean approach and its relevance feedback method, and revise the neural network method for relevance feedbacks. Finally, we provide the experimental results that are performed on a TREC collection,

2 P-Norm Based Extended Boolean Model

We first describe the p-norm based Boolean model. Salton *et al.*[12] proposed an extended Boolean model in order to overcome disadvantages in the conventional Boolean retrieval model [5]. Specifically, they introduce AND/OR

[1] This research was supported in part by the Korean Dept. of MIC, Grant No. AA-00-2060-00.

$$sim(D, Q_{OR(p)}) = \left[\frac{q_1^p a_1^p + q_2^p a_2^p + ... + q_n^p a_n^p}{q_1^p + q_2^p + ... + q_n^p} \right]^{1/p} \quad (1)$$

logical connectives with weighted terms as follows. Consider a set of terms $t_1, t_2 \ldots, t_n$ and let a_i denote the weight of term t_i in a document $D = (a_1, a_2, \ldots, a_n)$ where $1 \le i \le n$ and $0 \le a_i \le 1$. Suppose that an OR-query is given as $Q_{OR(p)} = OR^p(q_1, q_2, \ldots, q_n)$ where q_i indicates the weight of query term t_i, $0 \le q_i \le 1$, and $1 \le p \le \infty$. Similarly, suppose that $Q_{AND(p)}$ is given as $Q_{AND(p)} = AND^p(q_1, q_2, \ldots, q_n)$. The similarities between a given document $D = (a_1, a_2, \ldots, a_n)$ and $Q_{OR(p)}$, $Q_{AND(p)}$ are defined as

In this extended Boolean model, when $p = 1$, the distinction between the AND and OR connectives disappears. In this case, the similarity between queries and documents can be computed by the inner product between their weights. This means that a simple vector space model [9] is obtained when $p = 1$. When $p = \infty$ and all query

$$sim(D, Q_{AND(p)}) = 1 - \left[\frac{q_1^p (1-a_1)^p + q_2^p (1-a_2)^p + \ldots.. + q_n^p (1-a_n)^p}{q_1^p + q_2^p + \ldots.. + q_n^p} \right]^{1/p} \qquad (2)$$

terms are equal to 1, we can obtain $sim(D, Q_{OR(\infty)}) = \max(a_1, a_2, \ldots, a_n)$ and $sim(D, Q_{AND(\infty)}) = \min(a_1, a_2, \ldots, a_n)$. By varying the value of p between 1 and ∞, it is possible to obtain a system intermediate between a pure vector model ($p = 1$) and a conventional Boolean retrieval system ($p = \infty$).[2]

3 Relevance Feedback Method in Extended Boolean Model

Salton *et al.*[11] provided a relevance feedback method for the p-norm based extended Boolean model. The relevance feedback method consists of two processes. The first process is to construct of "good" term clauses, where a clause is defined as a single term or a conjunction of several terms connected by AND operators. The second process is to generate an extended Boolean query in a disjunctive normal form (DNF) with the properly chosen clauses as follows. Suppose that the reformulated query is expected to retrieve approximately T documents, where T is a parameter specified by the user. Among the constructed clauses, highly relevant single terms or conjunctions of several terms will be repeatedly chosen until the number of the retrieved documents expected by the chosen clauses reaches near to T. To decide the highly relevant clauses, two kinds of measure are used: 1) the relevance weight of clause, which expresses the degree to which the clause is likely to be useful in retrieving relevant documents from the collection; and 2) the expected postings frequency, which represents the approximate number of documents which the clause may be expected to retrieve. A reformulated query then consists of a set of the chosen clauses interconnected by OR operators.

The clause construction process identifies a set of good single terms (that is, terms with high relevance weight), as well as good ANDed term pairs and ANDed term triples. For each clause, an estimated postings frequency is generated, defined as $n_t, n_s \bullet n_t / N$, and $n_r \bullet n_s \bullet n_t / N^2$ for single term *t*, ANDed pairs *st*, and ANDed triples *rst*, respectively, where n_* is the postings frequency of term * and N is the total number of documents in the collection. The computed estimated postings frequencies are exact under the assumption that all terms are assigned independently of each other to the documents of a collection.

The query reformulation process constructs Boolean queries designed to retrieve the wanted number T of documents. The basic idea consists in starting with a broad query, for example, the set of all previously chosen single terms connected by OR operators, and comparing the estimated number of documents retrieved by such an initial query (*estret*) with the retrieval threshold T. As a first approximation, *estret* is computed as the sum of the postings frequencies of the individual terms in the query. Since *estret* will initially be much larger than T, single terms are now successively deleted in increasing order of their relevance weight (that is, worst terms are deleted first), and replaced by ANDed term pairs that are not subsumed by the singles still present in a given Boolean statement. Whenever *estret* exceeds T, shorter clauses are deleted and replaced by more specific longer ones, until eventually an *estret* value between T/2 and T is obtained. The final query is then formed by connecting the clauses still present with OR operators. For more detailed description, see [11].

[2] For more information, see [12].

4 Neural Network Model for Relevance Feedbacks

In this section, we propose a scheme to map a DNF formula generated by the above relevance feedback method, into a multi-layered perceptron. The main idea of this mapping is originated by Kim and Raghavan's work [4]. The following two subsections describe how to construct the corresponding multi-layered perceptron for the DNF formulas and suggest its activation function and learning rules.

4.1 Neural Network Structure for the DNF formulas

Let us consider a query q which is generated by the above relevance feedback method, for example, $q = OR((t_1, q_1), (t_2, q_2), (c_1, q_7), (c_2, q_8))$, where q_i is the corresponding weight to term t_i for $i = 1$ to 6, $c_1 = AND((t_3, q_3), (t_4, q_4))$, $c_2 = AND((t_5, q_5), (t_6, q_6))$, and q_7 and q_8 are the corresponding weights to the conjunction c_1 and c_2, respectively. For this DNF, we can consider a AND/OR tree[3] as in Fig 1. In this figure, we assume that the input term weights $a_1, \ldots, a_6$ are given to the leaf nodes which denote terms $t_1, \cdots, t_6$, respectively. If the input weights are the weights for a document d, we can compute the similarity between document d and query q using equation (1) and (2).

We transform the AND/OR tree for the generated DNF into a multi-layered perceptron. The transformed neural network has the same topology of the given AND/OR tree. For example, the AND/OR tree in Fig 1 is transformed into the form in Fig 2. However, the corresponding weights and inputs must be transformed as follows. For OR and AND node i, net input value h_i can be defined in equation (3) and (4), respectively. The input values are raised to the power p. The weight w_i for $1 \le i \le n$ is computed as in equation (5)

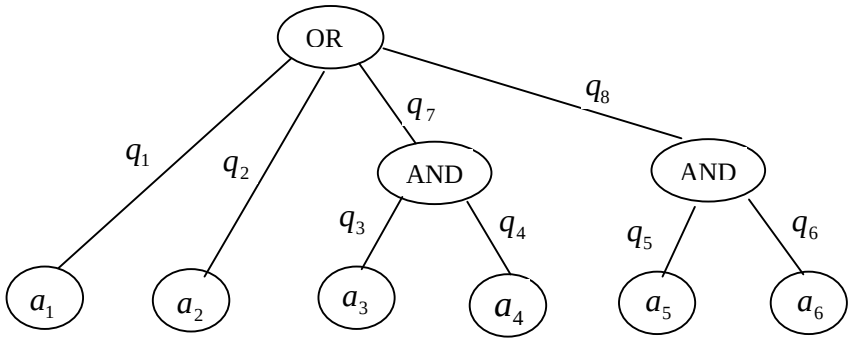

Fig 1. AND/OR tree for DNF q

$$\text{If the node } i \text{ is an } OR \text{ node}, \quad h_i = \sum_j w_{ij} a_j^p \tag{3}$$

$$\text{If the node } i \text{ is an } AND \text{ node}, \quad h_i = \sum_j w_{ij} (1 - a_j)^p, \tag{4}$$

where w_{ij} is the weight from the node j to the node i and,

if node j is a hidden node then a_j is the activation of node j,

otherwise (i.e., j is an input node) a_j is the input value.

$$w_i = \frac{q_i^p}{q_1^p + q_2^p + ... + q_n^p} \qquad \text{for } 1 \le i \le n \tag{5}$$

[3] Actually, this can be an acyclic undirected graph. For example, this is the case if $a_2 = a_3$.

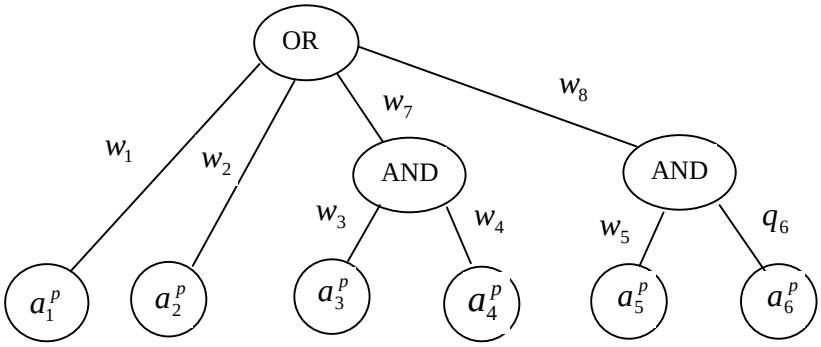

Fig 2. Transformed neural network of the AND/OR tree in Fig 1.

Now, let us consider the activation functions for OR and AND nodes. From equation (1) and (2), we can define the activation functions as follows.

$$a_i = F_{OR}(h_i) = h_i^{1/p} \qquad a_i = F_{AND}(h_i) = 1 - h_i^{1/p} \tag{6}$$

However, since these functions are not appropriate to be used as activation functions for the purpose of learning process, we use the same approximation functions used in Kim and Raghavan's work [4].

4.2 Activation Function and Learning Rule

From a mathematical point of view, the functions in equation (6), which have 1/p in the exponent, are exactly the correct choices to be used as the activation functions for the proposed neural network. However, since the derivative of such functions become extremely large near zero, we use an approximation of the 1/p powered functions that can easily get the convergence in the learning process. The adopted function is a variation of the sigmoid, which we call 2p-sigmoid function, defined as.

$$\hat{F}(x) = \frac{1}{1+e^{-2p(x-\theta)}}, \quad \text{where } \theta \text{ is 0.5.}$$
$$a_i = \hat{F}_{OR}(h_i) = \frac{1}{1+e^{-2p(h_i-\theta)}} \tag{7}$$

We use (x) above, instead of $x^{1/p}$, to approximate the activation functions for OR nodes and AND nodes, respectively, as follows.

For the learning process, we use the back-propagation learning rule [6]. Its derivation is fairly straightforward. The error measure *E* is defined as the total quadratic function at the output nodes:

$$a_i = \hat{F}_{AND}(h_i) = 1 - \frac{1}{1+e^{-2p(h_i-\theta)}} \tag{8}$$
$$E = \frac{1}{2}\sum (d_i - a_i)^2,$$

where d_i and a_i are the desired output and the actual output for the node i, respectively. Then, by the standard back-propagation formulas [6], we can write Δw_{ij}:

$$\Delta w_{ij} = \gamma \delta_i a_j^p \quad \text{for } OR \text{ node} \quad \text{and} \quad \Delta w_{ij} = \gamma \delta_i (1 - a_j)^p \quad \text{for } AND \text{ node } i$$

where $\gamma > 0$ is the learning rate and δ_i is the error signal given as follows. If the node i is an output node, δ_i is given by

$$\delta_i = (d_i - a_i)\hat{F}'(h_i)$$

When $_{OR}$ and $_{AND}$ are given as in equation (7) and (8), respectively, the derivatives are equal to

$$\hat{F}'_{OR}(h_i) = \frac{2p}{(1+e^{-2p(h_i-\theta)})^2} e^{-2p(h_i-\theta)} = 2pa_i(1-a_i)$$

$$\hat{F}'_{AND}(h_i) = \frac{-2p}{(1+e^{-2p(h_i-\theta)})^2} e^{-2p(h_i-\theta)} = 2pa_i(a_i-1)$$

If the node i is a hidden node, the error signal is determined recursively in terms of error signals of the nodes to which it directly connects and the weights of those connections. Therefore, the error signal is given by

$$\delta_i = \hat{F}'(h_i)\sum_k \delta_k w_{ki}, \quad \text{where } k \text{ is over all nodes in the layers above node } i.$$

5 Experimental Tests and Conclusions

For the experimental tests, DOE (Department of Energy) collection in TREC-1 is used. It contains about 220,000 documents. Three topics (topic 96, 134, 135) are chosen to evaluate the performance. The evaluation consists of two phases: learning phase and testing phase. In the learning phase, initial queries are reformulated by the relevance feedback method for the p-norm based extended model [11], and the reformulated queries then are tuned by our adjusting method using about 10,000 documents which are sampled from the DOE collection. For the phase, the improvement of relevance assessment is evaluated comparing the number of relevant documents newly retrieved by the reformulated and adjusted queries. The effectiveness of a retrieval operation is evaluated computing the recall-precision values. For the test, we construct three initial Boolean queries for the three topics as follows:

Topic 96: (medical AND computer) OR (equipment AND medical)
OR (medical AND diagnosis)
Topic 134: human AND genome
Topic 135: gene and medicine

Using the feedback method as described in Section 4, we reformulate the queries (DNFs) using the top ranked 100 documents retrieved from the sampled collection. Then, we adjust the weights of terms and clauses in the DNFs using our neural network with the same 100 documents as used in the reformulation step. The resulting performance is shown in Table 1. We can find that the reformulated queries (denoted by DNF in the table) retrieve more relevant documents within top 100 than the initial queries. We also note that the adjusted queries retrieve more relevant documents than the reformulated queries.

		No. of relevant documents within top 100	
		DNF	Neural Network
Topic 96	Initial query	8	
	1st feedback	13	14
	2nd feedback	13	
Topic 134	Initial query	14	
	1st feedback	16	
	2nd feedback	18	20
Topic 135	Initial query	13	
	1st feedback	14	16
	2nd feedback	14	

Table 1. Relevance feedback result and adjusting result.

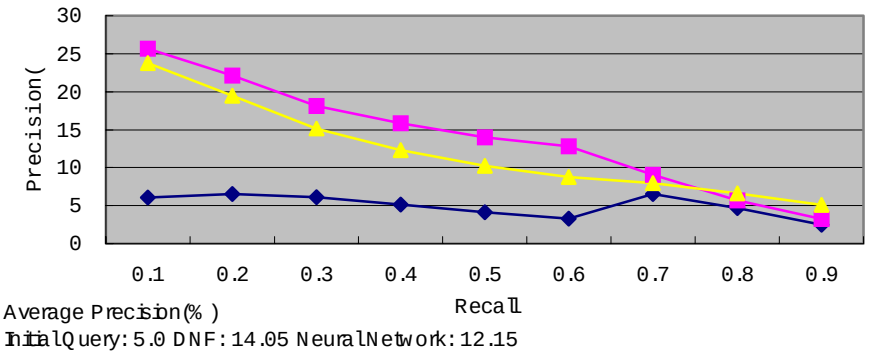

Fig 3. Recall-precision for topic 96.

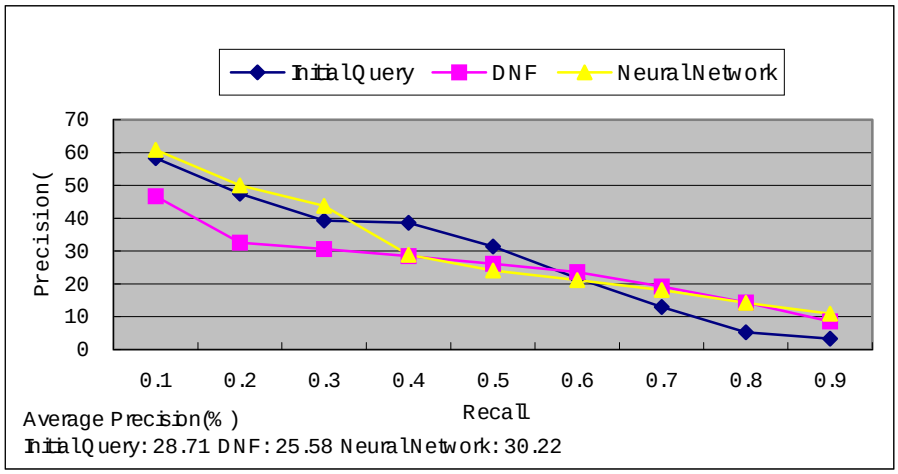

Fig 4. Recall-precision for topic 134.

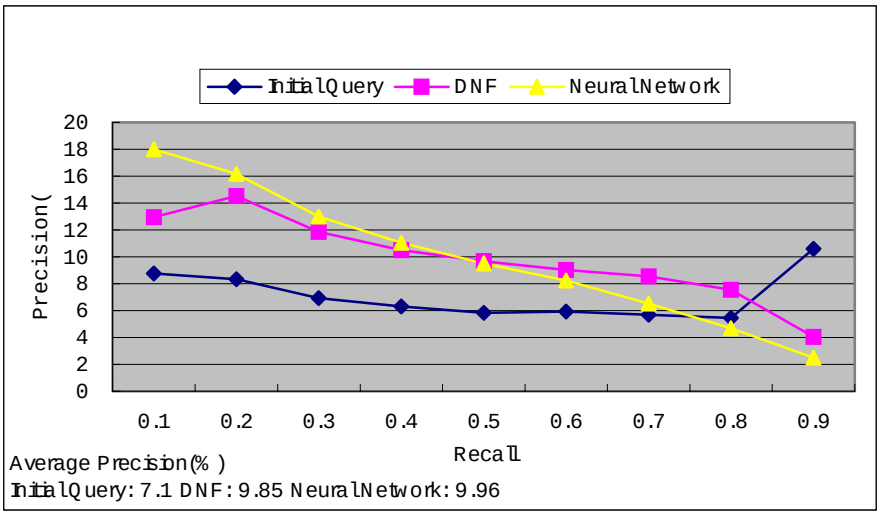

Fig 5. Recall-precision for topic 135.

To evaluate the effectiveness of a retrieval operation, it is common to compute the recall-precision values. Typically, recall-precision values are computed at fixed recall levels from 0.1 to 1.0 in steps of 0.1 [1, 10]. Using the reformulated queries and the adjusted queries, the precision/recall values are plotted in Figure 3, 4, and 5 for the three topics, respectively. In these figures, DNF indicates the feedback method to reformulate queries. As we can see, our neural network method is slightly better than the feedback method in terms of average precision for topic 134 and 135, although it is not the case for topic 96. Particularly, at the low levels of recall, our method is superior to the feedback method. In conclusion, we would like to say that the relevance feedback method usually improves

the retrieval performance, and that the adaptive method slightly improves the performance even after applying the feedback method.

References

1. Baeza-Yates, R. and Riberiro-Neto, B. *Modern Information Retrieval*. Addison-Wesley, 1999.
2. Bookstein, A. Fussy requests: An approach to weighted Boolean searches, *J. ASIS*, Vol 31, No. 4, July, 1980, pp. 275-279
3. Ide, E. New experiments in relevance feedback. In Salton, G., ed., *The Smart System – Experiments in Automatic Document Processing*, pp. 337-354. Englewood Cliffs, NJ: Prentice-Hall Inc.
4. Kim, M. and Raghavan, V. V. Adaptive concept-based retrieval using a neural network. In *Proc. Of ACM SIGIR 200 Workshop on Mathematical/Formal Methods in Information Retrieval*, Ayhens, Greece, July, 2000.
5. Lancaster, F. W. *Information retrieval systems: characteristics, testing and evaluation*, 2nd Ed., John Wiely and Sons, New York, 1979.
6. Lippmann, R. P. An introduction to computing with neural nets. *IEEE ASSP Magazine*, Vol. 3, No. 4, pp. 4-22, 1987.
7. Raghavan, V. V. and Wong, S. Acritical analysis of the vector space model for information retrieval. *Journal of the American Society for Information Science* 37(5): pp. 279-287, 1986.
8. Rocchio, J. J. Jr. Relevance feedback in information retrieval. In Salton, G., ed., *The Smart System – Experiments in Automatic Document Processing*, pp. 313-323. Englewood Cliffs, NJ: Prentice-Hall Inc.
9. Salton, G. and Buckely, C. Improving retrieval performance by relevance feedback. *J. of the American Society for Information Science*, 41(4): pp. 288-297, 1990.
10. Salton, G. and McGill, M. J. *Introduction to Modern Information Retrieval*. McGraw Hill, New York, 1983.
11. Salton, G., Fox, E. A., and Voorhees. Advanced Feedback Methods in Information Retrieval. *J. of the American Society for Information Science*, 36(3): pp. 200-210, 1985.
12. Salton, G., Fox, E. A., and Wu, H. Extnded Boolean Information Retrieval, Vol. 36, No. 11, December 1983, *Communication of the ACM*, pp. 1022-1036.
13. Waller, W. G. and Kraft, D. H. A mathematical model for a weighted Boolean retrieval system. *Information Processing and Management*, Vol 15, No. 5, 1979, pp. 235-245.
14. Wilkinson, R. and Hingston, P. Using the cosine measure in a neural network for document retrieval. In *Proceedings of ACM-SIGIR Conference*, 1991, pp. 202-210.
15. Wong, S.K.M., Ziarko, W., Raghavan,V., and Wong, P. C. N. Extended Boolean Query Processing in the Generalized Vector Space Model, *Information Systems* Vol. 14, No. 1, pp. 47-63, 1989.
16. Wong, S.K.M. and Cai, Y.J. Computation of term associations by a neural network. In *Proceedings of ACM-SIGIR Conference*, 1993, pp. 107-115.