Regression Analysis for Rival Penalized Competitive Learning Binary Tree

Xue Qun Li and Irwin King
{xqli, king}@cse.cuhk.edu.hk
Department of Computer Science & Engineering
The Chinese University of Hong Kong
Shatin, New Territories, Hong Kong

Abstract

The main aim of this paper is to develop a suitable regression analysis model for describing the relationship between the index efficiency and the parameters of the Rival Penalized Competitive Learning Binary Tree (RPCL-b-tree). RPCL-b-tree is a hierarchical indexing structure built with a hierarchical RPCL clustering implementation, which transforms the feature space into a sequence of nested clusters. Based on the RPCL-b-tree, the efficient Nearest-Neighbor search for a query can be performed with the branch-and-bound algorithm. The index efficiency of a RPCL-b-tree relates to a set of parameters: leaf node size of the tree, number of retrieved objects per search, feature dimensionality and database size. To formulate this relationship, we develop a nonlinear regression model in this paper. This regression model includes two components. One is used to describe the relationship between index efficiency and the number of retrieved objects per search; another is to describe the relationship between index efficiency and leaf node size of a RPCL-b-tree. In both of these two components, we consider the influence from the database size and the feature dimensionality. Our experimental results show that the proposed regression model has a high convergibility and high generalization ability. Moreover, this regression model is explainable on its coefficients, whose values directly reflect the index efficiency of a RPCL-b-tree. Depending on the regression model and its estimated coefficients, we can easily analyze the index efficiency of a RPCL-b-tree to be built. Because the parameters of the different kinds of indexing structures are very similar, this model also is suitable to analyze the other kinds of indexing structures. Thus, it is a powerful tool to construct the optimal RPCL-b-tree or other kinds of indexing structures for a database.

1 Introduction

One of the key issues in content-based Information Retrieval (IR) is the implementation of an efficient and robust indexing structure of the feature vectors. Without a properly designed indexing structure, the retrieval of information will be reduced to a linear exhaustive search. A good indexing structure will make the information retrieval perceptually accurate and computationally efficient. Rival Penalized Competitive Learning Binary Tree (RPCL-b-tree) is a hierarchical indexing structure, which is built by nested RPCL [7] clustering implementation. Based on the RPCL-b-tree, an efficient K Nearest Neighbor (NN) search can be implemented with a branch-and-bound algorithm, which ensures to search 100% K nearest neighbors like the linear search.

1.1 Problem Defined

A RPCL-b-tree has two type of nodes, leaf and nonleaf. The upper limit, M, of the leaf node sizes of a RPCL-b-tree needs to be pre-specified for the tree building implementation. The choice of M influences the index efficiency of a RPCL-b-tree. To choose a suitable M for obtaining the high index efficiency, we also consider the following parameters at the same time: (1) the size of a database, N, (2) the number of database objects retrieved per search, K, and (3) the feature dimensionality, D.

Here, we expect to obtain a formula to describe the relationship between the index efficiency and the parameters, M, N, K, and D of the RPCL-b-tree. Based on the formula, we can (1) choose the suitable parameters to build the optimal RPCL-b-trees, (2) analyze the index efficiency performances of the RPC-b-trees for different data distributions, and (3) compare the performances between the RPCL-b-tree and the other index structures.

It is quite difficult to obtain such a formula by using explicit mathematical deduction. The reasons are: (1) we have to know the exact data distribution of a database for the deduction but the underlying distribution of a database is

0-7695-0619-4/00/$10.00 (C) 2000 IEEE

unknown, and (2) we have to know the RPCL clustering results for a database but the clustering results are too complex to be estimated. On the other hand, it is not necessary to obtain a very exact formula since the approximate optimal values of the parameters are good enough to build a RPCL-b-tree in practice. According to the above considerations, a possible solution is to estimate the formula by using the regression analysis based on a suitable regression model. The unknown coefficients in the regression model are estimated with the test results of some pre-built RPCL-b-trees.

The problem is then to develop a suitable regression function for describing the relationship between the index efficiency and the parameters of a RPCL-b-tree. The basic requirements of this regression model are follows:

1. **High Convergibility:** In general, we estimate the coefficients of a regression model using an iterative method beginning with a set of initial coefficient estimates. The high convergibility means the regression model can converge easily from the initial states.

2. **High Generalization Ability:** Considering a typical large-sized database, it is difficult to build a test various RPCL-b-tree structures for the NN search. Therefore, the regression model should be robust where there is lack of tested data of the RPCL-b-tree. In other words, the estimated regression model based on the tested data can well be generalized to the untested data.

3. **Having Explainable Coefficients:** We expect the estimated coefficients in the regression model to be meaningful to explain the index efficiency of the RPCL-b-tree. It is very helpful to analyze the index efficiency of the RPCL-b-tree for different data distributions, and compare the index efficiency between RPCL-b-tree and the other indexing structures.

1.2 Previous Work

In [2, 3, 4, 5], we analyzed the influence of boundary problem for different data distributions, and then proposed the non-hierarchical and hierarchical indexing structure based on the RPCL clustering which is an unsupervised neural network learning algorithm. Using RPCL can quickly and accurately estimate the natural clusters in the feature space to overcome the boundary problem. In the related experiments, these RPCL-based indexing structures showed their abilities to overcome the boundary problem and improve the index efficiency. We discussed the experimental results, and then gave some suggestions to choose the suitable values of indexing structure parameters. However, no formula based on the regression analysis was provided in the above papers. Furthermore, we can rarely find the regression analyses for indexing structure efficiency in the current content-based IR research field. It is significant to find a suitable regression model to analyze the performances of the indexing structures.

In this paper, we propose a nonlinear regression model to describe the relationship between the index efficiency and the parameters of a RPCL-b-tree. This model consists of two decay growth components. One component describes the relationship between the index efficiency and the leaf node size of the RPCL-b-tree, and another describes the relationship between index efficiency and the number of retrieved objects. This model well satisfies the three basic requirements mentioned in the previous subsection. Because the parameters of the different kinds of indexing structures are very similar, the regression model also can be used to analyze the other kinds of indexing structures. It is a powerful tool to build the optimal indexing structure for a database in practice.

In Section 2, we will briefly describe the RPCL-b-tree. Then, we present the proposed regression model in Section 3. The experimental results are given in Section 4. We draw a conclusion in the last section.

2 Rival Penalized Competitive Learning Binary Tree

RPCL-b-tree is a hierarchical structure for indexing so that relationship can be found in the nodes between two levels. Given a set of feature vectors, the top-down nested RPCL clustering is performed to build a RPCL-b-tree.

The basic idea of RPCL-b-tree is that we use RPCL to cluster the feature vectors into two sub-clusters each time and then continue to do RPCL clustering hierarchically to each of the sub-clusters until each of the final sub-clusters contains less than a pre-specified number of data points. Thus, there are two kinds of nodes in the RPCL-b-tree:

Leaf Node A leaf node contains a cluster of at most M feature vectors calculated by RPCL clustering, where M is a predefined number.

Non-leaf Node A non-leaf node contains (1) a tuple which summarizes the clustering information including the number of feature vectors in this cluster and the center of the cluster, and (2) the pointers to its child nodes.

After the RPCL-b-tree is built, there exists a mapping function that maps the generated clusters to a binary indexing structure. For example, all the feature vectors are in one cluster at the root level and at most 2^i child nodes (clusters)

at depth i. Based on the built RPCL-b-tree for a database, we can retrieve relevant data for a query by performing the efficient NN search. During the search, a query vector is compared to the centers of the child nodes in the immediate lower level at the top level of the RPCL-b-tree. The child node with center closer to the query vector is selected. The search continually proceeds to the lower nodes until it reaches a leaf node. In this leaf node, the elements will be the results of the query if they satisfy the criteria of the nearest-neighbor search. To ensure the index accuracy is 100%, we can implement backtracking based on a branch-and-bound algorithm [1]. In this branch-and-bound algorithm, each node is tested to determine whether or not possibly containing the nearest-neighbor to a query by a set of rules [5].

3 Regression Model for RPCL-b-tree

3.1 Index Efficiency Measurement

In this paper, we define the index efficiency measurement, Y, as

$$Y = 1 - \frac{\text{\# of distance computations for the checked method}}{\text{\# of distance computations in linear search}}. \quad (1)$$

With the measurement, the efficiency of the linear search is 0. If the searching efficiency of a method is 0.8, the method needs only approximately 20% of the searching time needed by the linear search for retrieval.

3.2 Regression Function

We denote $\mu_Y(N, M, K, D)$ as the regression function of Y on parameters, N, M, K, D, which are defined in Section 1. Let $x_{NM} = N/M - 1$, $x_{NK} = N/k - 1$, $x_{KM} = K/M$ and $x_D = 1/D$, the regression function can be written as follow:

$$\mu_Y(N, M, K, D) = \mu_Y(x_{NM}, x_{NK}, x_{KM}, x_D) = \mu_{Y1}(x_{NM}, x_D) \cdot \mu_{Y2}(x_{NK}, x_{KM}, x_D), \quad (2)$$

where

$$\mu_{Y1}(x_{NM}, x_D) = 1 - c_1, \quad (3)$$

$$c_1 = \exp(-\alpha_1 \cdot x_{NM} \cdot x_D), \quad (4)$$

and

$$\mu_{Y2}(x_{NK}, x_{KM}, x_D) = 1 - c_2 \cdot c_3, \quad (5)$$

$$c_2 = \exp(-\alpha_2 \cdot x_{NK} \cdot x_D), \quad (6)$$

$$c_3 = \exp(-\alpha_3 \cdot x_{KM}^{\alpha_4} \cdot x_D). \quad (7)$$

In this regression model, α_1, α_2, α_3 and α_4 are the unknown coefficients which need to be estimated from the test results for the pre-built RPCL-b-trees. Next, we show the properties of each term in the regression model.

3.3 Properties of the Regression Function

- μ_{Y1} is used to describe the relationship between retrieval efficiency and M. In Equation (3), c_1 is an exponential function which corresponds to the partition proportion, N/M. $x_{NM} = N/M - 1$ is used to change the bias of N/M from 1 to 0. α_1 which satisfies the constraint, $\alpha_1 \geq 0$, is the unknown coefficient in c_1.

  Suppose M satisfies $M \leq N$, then $x_{NM} \geq 0$ and $0 \leq c_1 < 1$. Thus, $1 > \mu_{Y1} \geq 0$. When $M = N$, only one node is in the indexing structure, and the needed distance computation amount for a query is exactly as same as using linear search. In this case, $x_{NM} = 0$, $c_1 = 1$ and $\mu_{Y1} = 0$. Following the increase of x_{NM}, c_1 decreases and μ_{Y1} increases.

  In μ_{Y1}, the unknown coefficient α_1 reflects the effectiveness of a partition algorithm. If $\alpha_1 = 0$, $\mu_{Y1} = 0$ for any M. That means the partition algorithm is totally non-effective. The bigger the α_1, the more effective of the partition algorithm, and the more efficient of the RPCL-b-tree.

- μ_{Y2} is used to describe the relationship between retrieval efficiency and the number of retrieved objects. In Equation (5),

  1. c_2 corresponds to the proportion, N/K. $x_{NK} = N/K - 1$ is used to change the bias of N/K from 1 to 0. α_2 which satisfies the constraint, $\alpha_2 \geq 0$, is the unknown coefficient in c_2. Suppose $K \leq N$, then $x_{NK} \geq 0$ and $0 \leq c_2 < 1$. When $K = N$, each object in each leaf node needs to be visited and the distance computation amount is exactly as same as using linear search. In this case, $x_{NK} = 0$, $c_2 = 1$ and $\mu_{Y2} = 1 - c_3$. Following the increase of x_{NK}, c_2 decreases and μ_{Y2} increases. In c_2, the unknown coefficient α_2 mainly reflects the effectiveness of a searching algorithm. If $\alpha_2 = 0$, $\mu_{Y2} = 1 - c_3$ for any K, the searching algorithm is non-effective at all. The bigger the α_2, the more effective of the branch-and-bound algorithm, and the more efficient of the RPCL-b-tree.

2. c_3 corresponds to the ratio, $x_{KM} = K/M$. General speaking, the smaller the node size, the more the efficiency of an indexing structure, but the more the number of retrieved objects, the less the efficiency. So, we use c_3 to describe the combined influence concerning K and M. α_3 and α_4, which can be any real number, are two unknown coefficients in c_3. If $\alpha_3 > 0$, following the increase of x_{KM}, c_3 decreases and μ_{Y2} increases. If $\alpha_3 < 0$, following the increase of x_{KM}, c_3 increases and μ_{Y2} decreases. If $\alpha_3 = 0$, that means the influence of R and M can be counteracted by another. Hence, the unknown coefficient α_3 in c_3 reflects the combined effectiveness of the partition and searching algorithm. The bigger the α_3, the more the effectiveness. α_4 is a curve shape adjustment coefficient in c_3 to improve the convergibility of the regression model. It also reflects the combined effectiveness of partition and searching.

- $x_D = 1/D$ is used to describe the influence of dimensionality in c_1, c_2 and c_3. The bigger the D, the smaller the x_D and the bigger the c_1, c_2 and c_3.

The total estimate, μ_Y, of the index efficiency, Y, is the product of Equation (3) and Equation (5).

3.4 Estimating the Unknown Coefficients

Our regression function is a nonlinear function. So we perform nonlinear least-square data fitting technique by using the Gauss-Newton method [6] for estimating the unknown coefficients in the regression function. The Sum of Squared Errors (SSE) between Y and μ_Y is estimated from a set of tests, $\{y_i, n_i, m_i, k_i, d_i\}_{i=1}^{s}$, as follow:

$$SSE = \sum_{i=1}^{s} \left[y_i - \mu_Y(n_i, m_i, k_i, d_i) \right]^2 , \tag{8}$$

where s is the number of tests, and y_i, n_i, m_i, k_i and d_i are the index efficiency, database size, maximum leaf node size of the RPCL-b-tree, number of retrieved objects and dimensionality of the i-th test. The Gauss-Newton method is an iteration method to find the coefficients which minimize SSE. Here, we define two stop conditions for Gauss-Newton method: (1) $(|SSE - SSE_{old}|)/SSE \leq \sigma$, where σ is a small constant, or (2) the number of iterations is greater than a predefined maximum number.

4 Experiments

4.1 Experimental Setup

We use three different types of test feature vector sets in our experiments:

1. **Gaussian Mixture Distribution Data:** Synthetic feature vector sets generated from the Gaussian mixture distributions, where the numbers of mixtures are 10, 100 and 1,000. The dimensionality of the feature vector sets varies from 2 to 16. The sizes of the feature vector sets are 1,000, 2,000, 5,000, 10,000, 20,000 and 50,000.

2. **Uniform Data:** Synthetic feature vector sets in uniform distribution with dimensions varying from 2 to 16. The sizes of the feature vector sets are 1,000, 2,000, 5,000, 10,000, 20,000, and 50,000.

3. **Real Data:** Apart from synthetic data, we use an image database with 10,000 images. Each image is described using an 8-D feature vector, which is extracted from the 8-bucket color histogram of the image.

We build different RPCL-b-trees for each of the testing feature vector sets, with different pre-specified maximum node sizes, M = 100, 200, 500, 1,000 and 2,000. Then we test the built RPCL-b-tree by performing the K NN search with the branch-and-bound algorithm, and calculate the index efficiency with the Equation 1, where K = 1, 5, 20, 50, and 100. In each test, 10 different NN searches are performed and the average results are used for analysis.

4.2 Regression of Experimental Results

We apply Equation (2) with Gauss-Newton method to regress the three different sets of test results. The convergibility of our regression model is very high since almost each regression calculation can converge within 100 iterations with the following very loose initialization condition, $\alpha_1 = \alpha_2 = \alpha_3 = \alpha_4 > 0$. Table 1 shows the estimated coefficients in the regression model for each set of test results.

We test the generalization ability of the proposed regression model. We re-estimate the regression model by excluding the part of test results in each experiment. Even then, the estimated coefficients using the incomplete test

results are very similar with the one using the all results. For example, Table 2 shows the case where the test data with the maximum $M = 2,000$ or $K = 100$ are excluded from the original experimental results, but the estimated coefficients in Table 2 are almost as same as Table 1 although the intervals between the excluded and remained data are quite large. Thus, the generalization ability of the proposed regression model is very high. This is a robust regression model.

4.3 Analyze the Performances of RPCL-b-tree Based on Estimated Regression function

As we mentioned previously, the regression model is a powerful tool for us to analyze the index efficiency performance of the RPCL-b-trees. Some useful analyses are listed as follows:

- **Estimation of Index Efficiency:** Using the regression function 2, we can easily estimate the index efficiency of a RPCL-tree with any given parameters.
- **Analyses Using Plots:** Through the 2-D and 3-D plots of the regression function, we can perform intuitive analysis to build a suitable RPCL-tree. For example, Figure 1 shows the relationship between the index efficiency of M, K. From this figure, we can find (1) the index efficiency of RPCL-b-tree is very high for this set of real data, and (2) the influence from M, to index efficiency is very small. Therefore, considering that using too small M will lead to more clustering implementations and increase the tree building time, we can use a relatively big M for this data set.
- **Analyses Using the Estimated Coefficients in Regression Function:** We can analyze the index efficiency of RPCL-b-tree from c_1, c_2 and c_3 and the estimated coefficients. Two examples are given as follows,

  Comparison of Index Efficiencies of RPCL-b-tree for Different Data Distributions - Table 3 describes the relationship between the index efficiency and estimated coefficients for the feature vector sets with different distributions. From this table, we can know the RPCL-b-tree method is more effective for real data and Gaussian mixture distributed data than for uniform distributed data.

  Comparison between RPCL-b-tree and VP-tree - Table 4 shows the estimated coefficients in the regression models for Gaussian mixture distribution test set using RPCL-b-tree and VP-tree indexing structure. Table 5 shows the related analyses. From this table, we can know the RPCL-b-tree method is more efficient than the VP-tree method for indexing Gaussian mixture distributed data.

5 Conclusion

In this paper, we develop a nonlinear regression model to formulate the relationship between index efficiency and a set of parameters of the RPCL-b-tree. This regression model includes two components. One is used to describe the relationship between index efficiency and the number of retrieved objects per search; another is to describe the relationship between index efficiency and leaf node size of the RPCL-b-tree. The estimated coefficients in this model directly reflect the index efficiency of a RPCL-b-tree. Depending on the regression model with its estimated coefficients, we can analyze the index efficiency of the RPCL-b-tree for different data distributions and compare RPCL-b-tree and other indexing structure in detail. Our experimental results show that the proposed regression model has a high convergibility and high generalization ability. Because the parameters of the different kinds of indexing structures are very similar, this regression model also is very suitable to analyze the other kinds of indexing structures. Thus, it is a powerful tool to construct the optimal RPCL-b-tree or other kinds of indexing structures for a database in practice.

References

[1] B. Kamgar and L. N. Kanal. An improved branch and bound algorithm for computing k-nearest neighbors. *Pattern Recognition Letters*, 3(1):7–12, January 1985.

[2] Irwin King and Tak Kan Lau. Comparison of several partitioning methods for information retrieval in image databases. In *Proceedings of the 1997 International Symposium on Multimedia Information Processing (ISMIP'97)*, pages 215–220, 1997.

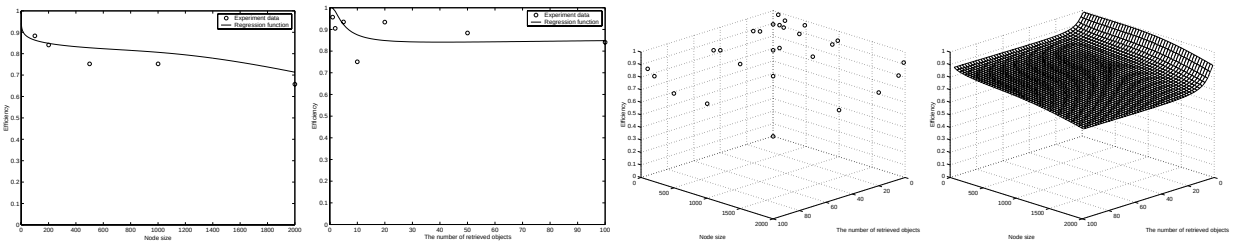
[3] Irwin King and Tak Kan Lau. Competitive learning clustering for information retrieval in image databases. In *Proceedings of the 1997 Internation Conference on Neural Information Processing (ICONIP'97)*, pages 906–909, 1997.

[4] Tak Kan Lau and Irwin King. Performance analysis of clustering algorithms for information retrieval in image databases. In *Proceedings to the International Joint Conference on Neural Networks (IJCNN'98)*, pages 932–937, 1998.

[5] T.K. Lau. Rival penalized competitive learning for content-based indexing. Master's thesis, The Chinese University of Hong Kong, 1998.

[6] John O. Rawlings. *Applied regression analysis*. Pacific Grove, Calif., 1988.

[7] L. Xu, A. Krzyżak, and E. Oja. Rival penalized competitive learning for clustering analysis, rbf net, and curve detection. *IEEE Transactions on Neural Networks*, 4(4):636–649, July 1993.



(a) (b) (c) (d)

Figure 1: The performance of RPCL-b-tree for indexing a set of real data. (a) Experimental results and estimated regression function, where $K = 100$ and $M = 1$ to 2000. (b) Experimental results and estimated regression function, where $M = 200$ and $K = 1$ to 100. (c) Experimental results and (d) estimated regression function, where $M = 1$ to 2000 and $K = 1$ to 100. In (a), (b), (c) and (d), the database size, N, is fixed on 10000 and the dimensionality, D, is fixed on 8.

Test Set	Indexing Method	Number of Tests	α_1	α_2	α_3	α_4
Gaussian-100	RPCL-b-tree	72	2.0565	0.06658	8.4514	0.6281
Uniform	RPCL-b-tree	72	2.2933	0.01065	3.9154	0.050
Real	RPCL-b-tree	27	4.5740	0.00445	15.4839	0.0795

Table 1: The estimated coefficients in the regression model for different test feature vector sets with RPCL-b-tree.

Test Set	Indexing Method	Number of Tests	α_1	α_2	α_3	α_4
Gaussian-100	RPCL-b-tree	54	2.0335	0.06519	8.8094	0.6494
Uniform	RPCL-b-tree	54	2.3174	0.01038	3.8989	0.0346
Real	RPCL-b-tree	18	4.872	0.00371	15.5793	0.0692

Table 2: The estimated coefficients in the regression model using parts of test results.

Test Set	α_1	c_1	α_2	c_2	α_3	c_3	μ_Y
Gaussian	Small	Big	Big	Small	Big	Small	**Middle**
Uniform	Small	Big	Middle	Middle	Small	Big	**Low**
Real Data	Big	Small	Small	Big	Very Big	Very Small	**High**

Table 3: The analyses of the index efficiency of RPCL-b-tree for different data distributions based on the estimated coefficients.

Test Set	Indexing Method	α_1	α_2	α_3	α_4
Gaussian	RPCL-b-tree	2.0565	0.06658	8.4514	0.6281
Gaussian	VP-tree	4.5592	0.00015	3.2215	-0.1969

Table 4: The estimated coefficients in the regression models for Gaussian mixture test set with RPCL-b-tree and VP-tree indexing structure.

Indexing Method	α_1	c_1	α_2	c_2	α_3	c_3	μ_Y
RPCL-b-tree	Small	Big	Big	Small	Big	Small	**High**
VP-tree	Big	Small	Small	Big	Small	Big	**Low**

Table 5: The comparison of index efficiency between RPCL-b-tree and VP-tree indexing structure based on estimated coefficients.