XIII. Experiments with a Fast Algorithm for Automatic Classification

R. T. Dattola

Abstract

An algorithm for automatically classifying items in a collection is described, where the time necessary is proportional to $N \cdot p \cdot \log_m p$, where N is the number of items in the collection, m is the final number of clusters desired, and p is the number of clusters produced at each level of the algorithm. Clusters are produced for two different document collections, and the results are evaluated by performing centroid searches on the clustered collection. Finally, a new evaluation measure is defined, and comparisons are made with clusters produced by other automatic methods.

1. Introduction

In order to use real-time automatic document retrieval systems on a very large data base, it is necessary to classify the documents so as to avoid searching the entire collection. However, the classification of a document collection containing more than a few thousand items is not feasible with most procedures for automatic classification. A method is needed which can classify hundreds of thousands of items into useful clusters in a reasonable amount of time.

The method presented in this study is an outgrowth of earlier attempts at fast procedures for automatic classification. [1,2] The algorithm is first presented in a general form, and a proof is given which describes how the time needed to classify a given collection is related to

the number of documents in the collection. Then the algorithm as implemented is described in detail and experiments in classification and evaluation are discussed.

2. General Description

Consider a document collection consisting of N documents with an average of r concepts per document. Assume that the document collection is partitioned into p equal sized clusters, where S_j is the set of documents in cluster j. Associated with each set S_j is a corresponding profile vector P_j, consisting of the rank values of all the concepts from the documents in S_j. The rank value is equal to a constant (base value) minus the rank of the concept, where concepts are ranked in decreasing order of the number of documents in the cluster in which they occur.

A cycle of the algorithm is defined as follows:

a) each document d_i in the collection is scored against each of the p profiles by a scoring function $g(d_i,P_j)$;

b) let $H_i = \max_{1\leq j\leq p} (g(d_i,P_j))$, (i.e., H_i is the highest score of d_i over all the profiles), and let

$$K_i = \begin{cases} H_i - a\cdot(H_i-K), & \text{if } H_i > K \\ K & \text{otherwise} \end{cases}$$

where $0 \leq a \leq 1$ and K is a specified cut-off score; then define new clusters $S'_j = \{d_i / g(d_i,P_j) \geq K_i\}$;

c) all documents d_i such that $H_i<K$ are assigned to a set L' of loose documents;

d) test for convergence; i.e., test if $S^*_j = S^{*\prime}_j$ and if $L' = L$, where $S^*_j = \{d_i / d_i \varepsilon S_j$ and $g(d_i,P_j)$ is the highest score of $d_i\}$.

If the convergence test fails, the cycle is repeated with the new clusters

S_j' and the new profiles P_j'. If the test succeeds, then an iteration of the algorithm has been completed. Since the algorithm is not guaranteed to terminate, an arbitrary upper bound B is placed on the maximum number of cycles allowed per iteration. [1]

Additional iterations of the algorithm can be performed by lowering the cut-off value K, thereby reducing the number of loose documents. One level of the algorithm is completed when the number of loose documents is less than a specified constant. Additional levels of the algorithm are executed by treating each of the p clusters as a separate collection and classifying each of these into p additional clusters. Thus, at the end of the first level the document collection has been classified into p clusters, at the end of the second level into p^2 clusters, etc.

In order to compute the time required for the algorithm, assume that m final clusters are desired, where $m = p^x$ for some positive integer x. An assumption is made that the time needed to compute the scoring function $g(d_i,P_j)$ is independent of the length of P_j, and depends only on the number of concepts in d_i (this assumption is valid for the scoring function that has been implemented). Since a document contains on the average r concepts, the time required for the computation of $g(d_i,P_j)$ is proportional to r.

In the following computation, it is assumed that the parameter a in step b of the cycle description is 0. As will be shown later, this specifies the overlap to be 0; i.e., documents cannot occur in more than one cluster. If the overlap is not 0, the computation time is increased since the average number of documents/cluster is greater. However, the increase in time depends only on the overlap and not on the number of documents in

in the collection. For example, if a document occurs on the average in two clusters instead of just one, then the computation time is doubled independent of N. Also, it is assumed that the average document size r is independent of N. For these reasons, overlap and document size will be included in k, the constant of proportionality.

On the first level of the algorithm, each of the N documents is compared against p profiles. The number of iterations and cycles required depend on the parameters B and K and again are independent of N. Thus, for the first level,

$$T = k \cdot N \cdot p \ .$$

On the second level, each of the p clusters now contain on the average N/p documents. These are compared against p new profiles, and this is repeated for each of the p clusters. Thus,

$$T = k \cdot (N/p) \cdot p \cdot p = k \cdot N \cdot p \ .$$

Similarly for all the other levels, $T = k \cdot N \cdot p$. At the end of the x'th level, there are p^x clusters. Thus, x levels must be executed to obtain $p^x = m$ clusters. Therefore, the total time required is $T = k \cdot N \cdot p \cdot x$, where

$$\begin{aligned} p^x &= m \\ \log p^x &= \log m \\ x \cdot \log p &= \log m \\ x &= \log m / \log p \end{aligned}$$

Taking logarithms to base p yields

$$T = k \cdot N \cdot p \cdot \log_p m \ .$$

Given a collection of N documents, if m final clusters are to be produced, what is the value of p which minimizes T ? This is solved by setting $dT/dp = 0$. Since $k \cdot N$ is a constant with respect to p , dT'/dp can be used, where $T' = p \cdot \log_p m$.

$$dT'/dp = p \cdot (d\log_p m/dp) + \log_p m \text{ , where } \log_p m = \log_m m/\log_m p$$

$$= 1/\log_m p \text{ (change of base formula).}$$

$$dT'/dp = p \cdot d(1/\log_m p)/dp + 1/\log_m p$$

$$= \frac{-p \cdot d(\log_m p)/dp}{(\log_m p)^2} + 1/\log_m p$$

$$= -p \cdot \log_m e/p \cdot (\log_m p)^2 + 1/\log_m p$$

$$= -\log_m e/(\log_m p)^2 + \log_m p/(\log_m p)^2$$

$$= (\log_m p - \log_m e)/(\log_m p)^2$$

$$= \log_m (p/e)/(\log_m p)^2 \quad .$$

Thus, $dT'/dp = 0$ when $\log_m (p/e) = 0 \Rightarrow p/e = 1$. Therefore, T is minimized for $p = e$. Rounding off to the nearest integer gives $p = 3$.

Although the computation time is minimized for $p = 3$, in practice it might not be useful to form only three clusters at every level. However, since T has only one minimum point, the time increases monotonically as p increases.

The algorithm as described can be used to generate multi-level clusters. However, it is generally more effective not to equate one level of the algorithm with one level of a multi-level classification scheme. For example, suppose a set of 10^6 documents is to be classified into two levels, with 100 centroids on the first level and 10,000 centroids on the second level.

This can be done directly with two levels of the algorithm where $p = 100$. In this case

$$T = k \cdot 10^6 \cdot 10^2 \cdot \log_{100} 10^4 = k \cdot 10^6 \cdot 10^2 \cdot 2 = 2k \cdot 10^8 \ .$$

However, the specifications can also be satisfied by using four levels of the algorithm instead of only two, where the results of the second and fourth levels are used as the two levels of clusters. In this case, $p = 10$ and

$$T = k \cdot 10^6 \cdot 10 \cdot \log_{10} 10^4 = k \cdot 10^7 \cdot 4 = 4k \cdot 10^7 \ .$$

3. Implementation

In this section the algorithm is described in detail exactly as programmed. The major change from the general description is that only one level of the algorithm has been implemented. Thus, in all the experiments, $p = m$ and

$$T = k \cdot N \cdot m \cdot \log_m m = k \cdot N \cdot m \ .$$

In addition, this section describes several alternate methods for generating the initial clusters that are necessary to start the algorithm, and a formal definition of overlap is given.

The algorithm is designed to control three basic parameters: number of clusters, amount of overlap, and size of clusters. The number of clusters and amount of overlap are input parameters, while the size of clusters is internally fixed to vary no more than one-half to twice the average cluster size. In addition to these, several other parameters are controlled as explained throughout this section.

A) Initial Clusters

In order to execute the classification algorithm, it is necessary to designate initial clusters. Four different methods of initial cluster generation are implemented, and they are referred to as the correlation, similarity correlation, frequency, and random methods. All of the methods involve choosing only one document per cluster that is used as a seed to start the algorithm.

The correlation method uses the cosine correlation to locate documents that are highly correlated with many other documents.

1) Randomly select $\sqrt{N}$ documents from the collection, but exclude those documents that have already been defined as cluster seeds.

2) Compute the global average cosine correlation (GAVG) and standard deviation (STD) for the sample:

   a) start with the first document in the sample and compute the cosine correlation between this document and all other documents in the sample;

   b) calculate the mean value (MEAN) of the correlations computed in step a;

   c) repeat steps a and b for all documents in the sample;

   d) define GAVG as the mean value of all the values computed in step b;

   e) calculate STD for step d.

3) Use the chosen documents as cluster seeds if MEAN $\geq$ GAVG + STD. Assuming a normal distribution, approximately 16% of the documents in the sample are used as seeds.

4) Repeat steps 1-3 until the number of seeds = m. Use each seed as an initial cluster.

The similarity correlation is identical to the previous method except

that step 3 is changed as follows:

3) Use the chosen documents as cluster seeds if MEAN $\geq$ GAVG + STD and if the document correlates below a specified cutoff with each of the documents already defined as seeds.

This additional condition prevents two very similar documents from being used as seeds for different clusters.

The frequency method does not use random samples or correlation coefficients, but directly locates documents which have many concepts in common with other documents.

1) Calculate the frequency score (FS) for every document in the collection:
   a) for every concept in the document calculate its frequency f , where f = the number of documents in the collection in which the concept occurs;
   b) FS = the sum of the frequencies of all the concepts divided by the number of concepts in the documents.
2) Rank the documents in order of decreasing frequency scores, and pick the top m documents as cluster seeds.

Finally, the random method simply picks out m random documents from the collection to be used as cluster seeds.

B) Overlap

The coefficient used to measure the overlap between clusters is an m dimensional extension of Tanimoto's two dimensional correlation coefficient. The most obvious way to measure overlap is to simply compute the average number of clusters/document. When this number is 1, there is no overlap between clusters. However, the main problem with this measure is that the

upper bound depends on the number of clusters; e.g., for ten clusters the upper bound is 10, for one hundred clusters the upper bound is 100. Of course, the upper bound is reached when every document occurs in every cluster; i.e., each cluster is equivalent to the entire collection.

The generalized form of Tanimoto's coefficient has a lower bound of 0 and an upper bound of 1 independent of the number of clusters. Let S_i = a binary vector of length N specifying the documents in cluster i ; i.e.,

$$S_i(j) = \begin{cases} 1 \text{ if document } j \text{ is in cluster } i \\ 0 \text{ otherwise} \end{cases}$$

Let $\#S_i = \sum_{k=1}^{N} S_i(k)$; i.e., the number of documents in cluster i .

Let $I_{i,j} = S_i \cap S_j$. Then $\#I_{i,j}$ = the number of documents occurring in both cluster i and cluster j . Tanimoto's two dimensional coefficient is:

$$T = \#I_{i,j}/(\#S_i + \#S_j - \#I_{i,j}) \ .$$

Extending this to m dimensions yields:

$$\phi = \sum_{i=1}^{m-1} \sum_{j=i+1}^{m} \#I_{i,j} \ /[(m-1) \sum_{i=1}^{m} \#S_i - \sum_{i=1}^{m-1} \sum_{j=i+1}^{m} \#I_{i,j}] \ .$$

From now on, the numerator of ϕ will be referred to by NUM .

Theorem ϕ has a minimum value of 0 and a maximum value of 1. Furthermore, it attains these bounds if and only if there is no overlap, or if all clusters are identical respectively.

Proof

a) Clearly ϕ is 0 when $S_i \cap S_j = \phi$ for all i and j, $s \neq j$,

since $\#I_{i,j}$ is always 0. On the other hand, ϕ can only be negative or undefined if

$$(m-1) \sum_{i=1}^{m} \#S_i \leq NUM .$$

However, this is impossible since

$$(m-1) \sum_{i=1}^{m} \#S_i = \sum_{j=1}^{m-1} \sum_{i=1}^{m} \#S_i > \sum_{j=1}^{m-1} \sum_{i=j+1}^{m} \#S_i \text{ and } \#S_i \geq \#I_{i,j} .$$

b) Let $S_1 = S_2 = \ldots = S_m$. Then $\#S_i = \#S_1$, and $I_{ij} = I_{1,1} = S_1 \Rightarrow \#I_{i,j} = \#S_1$ for all i. Therefore,

$$\phi = NUM/[(m-1) \sum_{i=1}^{m} \#S_1 - NUM] \text{ , but } NUM = \sum_{i=1}^{m-1} \sum_{j=i+1}^{m} \#S_1 \Rightarrow$$

$$\phi = [(m-1)+(m-2)+\ldots+1] \cdot \#S_1 / [(m-1) \cdot m\#S_1 - ((m-1)+\ldots+1) \cdot \#S_1]$$

$$= [m(m-1)\#S_1/2] / [m(m-1)\#S_1 - m(m-1)\#S_1/2]$$

$$= [m(m-1)\#S_1] / [2m(m-1)\#S_1 - m(m-1)\#S_1]$$

$$= [m(m-1)\#S_1] / [m(m-1)\#S_1] = 1 .$$

ϕ can only be greater than 1 if NUM is greater than the denominator. If so, then

$$NUM > (m-1) \sum_{i=1}^{m} \#S_i - NUM$$

$$\Rightarrow 2 \cdot NUM > (m-1) \sum_{i=1}^{m} \#S_i$$

$$= [(\#S_2+\ldots+\#S_m)+\#S_1] + [(\#S_3+\ldots+\#S_m)+(\#S_1+\#S_2)]$$

$$+ \ldots + [\#S_m + (\#S_1 + \ldots + \#S_{m-1})]$$

$$= \sum_{i=1}^{m-1} \sum_{j=i+1}^{m} \#S_j + (m-1)\#S_1 + (m-2)\#S_2 + \ldots + \#S_{m-1}$$

$$= \sum_{j=1}^{m-1} \#S_1 + \ldots + \sum_{j=m-1}^{m-1} \#S_{m-1} + \sum_{i=1}^{m-1} \sum_{j=i+1}^{m} \#S_j$$

$$= \sum_{j=2}^{m} \#S_1 + \ldots + \sum_{j=m}^{m} \#S_{m-1} + \sum_{i=1}^{m-1} \sum_{j=i+1}^{m} \#S_j$$

$$= \sum_{i=1}^{m-1} \sum_{j=i+1}^{m} \#S_j + \sum_{i=1}^{m-1} \sum_{j=i+1}^{m} \#S_j$$

$$= 2 \sum_{i=1}^{m-1} \sum_{j=i+1}^{m} \#S_j \quad .$$

But this is impossible since $S_j \geq I_{i,j}$.

C) Algorithm

After the initial clusters are generated, there exist m clusters with one document in each cluster. The cutoff point for loose documents (K) is now set so that at the end of the first cycle each cluster will average five documents. At the end of each iteration, K is reset so that x percent of the loose documents will be clustered at the end of the next cycle, where x is an input parameter. The iterations continue until less than y percent of the documents remain loose, where y is also an input parameter. Finally, an option is provided to allow the loose documents to be assigned to the cluster against which they score highest (blending), or to simply group them all together into an additional cluster.

So far nothing has been said about the choice of the base value that is used in calculating the rank values of concepts in profiles. As discussed

elsewhere (1), the base value should be set a little above the average cluster size at the beginning of each iteration. The purpose of this is two-fold: first, to prevent the rank values from dropping below 1, and to allow documents to move freely between clusters. If the base value is too low, many rank values drop below 1 and all are reset to 1 even though the concepts have different frequencies. Fig. 1 illustrates a case where the base value is too low. For present experiments the base value is set to twice the average cluster size at the beginning of each iteration, but experiments have shown that this is probably too high. Even in the largest clusters, the lowest rank values do not come close to 1.

Another parameter which is controlled is the size of the clusters. This is accomplished by deleting those clusters whose size falls below one-half the average, and by breaking up those clusters whose size exceeds twice the average. These checks are made at the end of every cycle of the algorithm. Clusters which get too large are broken up into two non-overlapping clusters by the following algorithm:

1) Select a random sample from among the documents in the cluster and generate seeds as in steps 1-3 of the correlation method for initial cluster generation.

2) Pick those two documents from the seeds that correlate lowest with one another, and use these two documents as cluster centers for the two new clusters.

3) Assign the documents in the original cluster to the cluster center to which they correlate highest.

Consider step b in the general description of the algorithm in section 2. The parameter a is used to control the overlap, where $0 \leq a \leq 1$. However, a does not correspond to the actual overlap between

Concept	Frequency	Rank	Rank Value
c_1	7	1	3
c_2	5	2	3
c_3	4	3	1
c_8	2	5	1
c_9	3	4	1
c_{10}	1	6	1
c_5	1	6	1
c_4	1	6	1

base value = 5

Example of Low Base Value

Fig. 1

clusters as computed by ϕ . After every cycle of the algorithm, the overlap is computed and compared with the input parameter OVER , which is the requested amount of overlap. The parameter a is then adjusted so that the value of ϕ approaches OVER . However, when loose documents are blended into the nearest cluster at the end of the algorithm, the amount of overlap between clusters decreases since each loose document is assigned to only one cluster. But as the percent of loose documents at the end of the algorithm is an input parameter, the effect of blending on the overlap can be predicted beforehand. Instead of adjusting a so that ϕ approaches OVER , ϕ should approach XOVER so that ϕ equals OVER <u>after</u> blending.

The change in 0 after blending occurs only in the term

$$(m-1) \sum_{i=1}^{m} \#S_i$$

since some clusters increase in size. The change is

$$(m-1) \cdot \left(\sum_{i=1}^{m} \#S_i + y \cdot N \right)$$

where y is the percent of documents loose at the end of the algorithm. After blending ϕ should equal OVER, where

$$\text{OVER} = \text{NUM} / \left[(m-1) \cdot \left(\sum_{i=1}^{m} \#S_i - y \cdot N \right) - \text{NUM} \right] .$$

Before blending,

$$\text{XOVER} = \text{NUM} / \left[(m-1) \cdot \sum_{i=1}^{m} \#S_i - \text{NUM} \right] .$$

XOVER must be expressed in terms of OVER . Rewriting OVER,

$$\text{OVER} = \text{XOVER} \cdot \left[(m-1) \cdot \sum_{i=1}^{m} \#S_i - \text{NUM} \right] / \left[(m-1) \cdot \sum_{i=1}^{m} \#S_i - \text{NUM} + (m-1) \cdot y \cdot N \right]$$

Solving for XOVER,

$$XOVER = OVER \cdot [(m-1) \sum_{i=1}^{m} \#S_i - NUM + (m-1) \cdot y \cdot N] / [(m-1) \sum_{i=1}^{m} \#S_i - NUM]$$

$$XOVER = OVER \cdot [1 + (m-1) \cdot y \cdot N / ((m-1) \sum_{i=1}^{m} \#S_i - NUM)] .$$

After every cycle, XOVER is reset by the above formula and a is adjusted as follows:

1) If ϕ <XOVER , set a = a + (1-a) (XOVER-ϕ) .
2) If ϕ >XOVER , set a = a + a $\cdot$(XOVER-ϕ) .

The quantity (XOVER-ϕ) is the difference between the required overlap and the actual overlap, and (1-a) or a represents the maximum amount that a can be changed in the proper direction.

One final input parameter controls the centroid definition. Cluster centroids are defined as the profile of concepts and their corresponding rank values as weights. An input parameter z is provided which determines what percent of the concepts in the profile will be used to define the centroid. The concepts in each profile are sorted in decreasing order by rank values, and the top z percent of the concepts are used to define the centroid. Normally more than z percent are actually used, since the last concept in the top z percent may have several ties, and all ties are also taken.

4. Evaluation

The final evaluation of the clusters can only be made by performing centroid searches on the classified collection. Two separate types of evalu-

ation are possible: internal and external. The internal evaluation attempts to determine how variations in the parameters of the classification algorithm affect the search results. The external evaluation compares the retrieval results from clusters produced by this algorithm with other algorithms and with a full search.

A) Evaluation Measures

Standard recall-precision curves that are used to evaluate full searches are not satisfactory for centroid search evaluation. The problem is that recall-precision curves do not take into consideration the amount of work which must be done by the retrieval system. Since the main advantage in centroid searching is a reduction in the number of query-document correlations, it is important to include the amount of work performed in any evaluation.

The total number of correlations in a centroid search is equal to the number of query-centroid correlations plus the number of query-document correlations. Dividing this number by the total number of correlations necessary for a full search (N), yields a fraction which compares the amount of work in a centroid search to a full search. This fraction is referred to as the correlation percentage (C.P.).[3] One way of representing the results is to include the correlation percentage along with the standard recall-precision measures. However, experiments have shown that the standard recall-precision measures improve as the C.P. increases, even when comparing the results from different sets of clusters. It is difficult to decide whether a cluster set that produces a normalized recall of .75 and a normalized precision of .60 with a C.P. of .20 is better or worse than another set of clusters producing a normalized recall of .70 and a nor-

malized precision of .55 with a C.P. of .15. The C.P. can be controlled to some extent during the centroid search, but quite often it varies by ten percent or more from the desired value. This is due to the difference in sizes of the clusters, and cannot be avoided as long as all the documents in the selected clusters are correlated against the query.

A method is proposed which modifies the recall-precision curve by taking into account the value of the correlation percentage. Consider a query Q which has two relevant documents, R_1 and R_2. Assume, for example, that the total number of documents in the collection is 20. Fig. 2a shows the results of a possible centroid search where one cluster containing five documents is selected. Fig. 2b shows similar results for a different set of clusters where one cluster containing ten documents is searched. In both cases the two relevant documents are retrieved at the same ranks, so the recall-precision figures are identical.* However, if there are four clusters in both cases, then C.P. = 9/20 for case (a), while C.P. = 14/20 for case (b). Thus, the evaluation measure should rate case (a) better than case (b), since the same retrieval results are obtained with less work.

Consider now a method in which the precision is <u>not</u> held constant after a recall of 1, but instead is allowed to drop until the rank is equal to the total number of query-centroid plus query-document correlations that have been made. Fig. 3 illustrates the recall-precision results for the previous example using this new method of evaluation.

---

*In the present SMART evaluation system, the precision is held constant after <u>all</u> the relevant documents are retrieved.

Rank	Doc.	R	P
1	R_1	.5	1.0
2	N	.5	.5
3	N	.5	.33
4	R_2	1.0	.5
5	N	1.0	.5
6	N	1.0	.5
7	N	1.0	.5
8	N	1.0	.5
9	N	1.0	.5
10	N	1.0	.5
11	N	1.0	.5
12	N	1.0	.5
13	N	1.0	.5
14	N	1.0	.5
15	N	1.0	.5
16	N	1.0	.5
17	N	1.0	.5
18	N	1.0	.5
19	N	1.0	.5
20	N	1.0	.5

a) Five Documents Compared

Rank	Doc.	R	P
1	R_1	.5	1.0
2	N	.5	.5
3	N	.5	.33
4	R_2	1.0	.5
5	N	1.0	.5
6	N	1.0	.5
7	N	1.0	.5
8	N	1.0	.5
9	N	1.0	.5
10	N	1.0	.5
11	N	1.0	.5
12	N	1.0	.5
13	N	1.0	.5
14	N	1.0	.5
15	N	1.0	.5
16	N	1.0	.5
17	N	1.0	.5
18	N	1.0	.5
19	N	1.0	.5
20	N	1.0	.5

b) Ten Documents Compared

Standard Evaluation Measure

Fig. 2

Rank	Doc.	R	P
1	R_1	.5	1.0
2	N	.5	.5
3	N	.5	.33
4	R_2	1.0	.5
5	N	1.0	.4
6	N	1.0	.33
7	N	1.0	.29
8	N	1.0	.25
9	N	1.0	.22
10	N	1.0	.22
11	N	1.0	.22
12	N	1.0	.22
13	N	1.0	.22
14	N	1.0	.22
15	N	1.0	.22
16	N	1.0	.22
17	N	1.0	.22
18	N	1.0	.22
19	N	1.0	.22
20	N	1.0	.22

a) Five Documents Compared

Rank	Doc.	R	P
1	R_1	.5	1.0
2	N	.5	.5
3	N	.5	.33
4	R_2	1.0	.5
5	N	1.0	.4
6	N	1.0	.33
7	N	1.0	.29
8	N	1.0	.25
9	N	1.0	.22
10	N	1.0	.2
11	N	1.0	.18
12	N	1.0	.17
13	N	1.0	.15
14	N	1.0	.14
15	N	1.0	.14
16	N	1.0	.14
17	N	1.0	.14
18	N	1.0	.14
19	N	1.0	.14
20	N	1.0	.14

b) Ten Documents Compared

New Evaluation Measure

Fig. 3

Another problem occurs when some of the relevant documents are not retrieved. Suppose as in the previous example, that there are two relevant documents but only one of them is retrieved. Fig. 4 illustrates these results where both set of clusters retrieve the relevant document at the same rank.* Once again the recall-precision results are identical, even though case (a) proved as effective as (b) at less cost.

Instead of assigning all the unrecovered relevant documents to the lowest ranks, they are distributed uniformly thoughout the ranks greater than the total number of correlations performed. The first unrecovered relevant is always assigned the middle rank, and the others are spaced uniformly above and below it. Fig. 5 illustrates this assignment for the sample case. Since case (a) did less work, the unrecovered relevant is assigned a higher rank than in case (b). A graph which plots recall vs. precision at every rank (document level) using the modified results reflects the superiority of case (a) as illustrated in Fig. 6. A graph which plots recall vs. precision at selected recall points (recall level) is not used in the evaluation, because only the highest precision for a given recall is used in the graph. Notice that in cases where all the relevant have not been retrieved, the precision is held constant after the assigned rank of the last relevant, since the C.P. is already taken into account during the assignment of the unretrieved relevant documents.

In order to illustrate the effects of the new evaluation process on an actual collection, a set of ADI clusters called BASE is considered using both evaluation measures. Four centroid searches are performed on

*Relevant documents not retrieved are assigned to the lowest possible ranks in SMART.

Rank	Doc.	R	P
1	R_1	.5	1.0
2	N	.5	.5
3	N	.5	.33
4	N	.5	.25
5	N	.5	.2
6	N	.5	.17
7	N	.5	.14
8	N	.5	.13
9	N	.5	.11
10	N	.5	.10
11	N	.5	.09
12	N	.5	.08
13	N	.5	.08
14	N	.5	.07
15	N	.5	.07
16	N	.5	.06
17	N	.5	.06
18	N	.5	.06
19	N	.5	.05
20	R_2	.5	.1

a) Five Documents Compared

Rank	Doc.	R	P
1	R_1	.5	1.0
2	N	.5	.5
3	N	.5	.33
4	N	.5	.25
5	N	.5	.2
6	N	.5	.17
7	N	.5	.14
8	N	.5	.13
9	N	.5	.11
10	N	.5	.10
11	N	.5	.09
12	N	.5	.08
13	N	.5	.08
14	N	.5	.07
15	N	.5	.07
16	N	.5	.06
17	N	.5	.06
18	N	.5	.06
19	N	.5	.05
20	R_2	.5	.1

b) Ten Documents Compared

Standard Evaluation Measure

Fig. 4

Rank	Doc.	R	P
1	R_1	.5	1.0
2	N	.5	.5
3	N	.5	.33
4	N	.5	.25
5	N	.5	.2
6	N	.5	.17
7	N	.5	.14
8	N	.5	.13
9	N	.5	.11
10	N	.5	.1
11	N	.5	.09
12	N	.5	.08
13	N	.5	.08
14	N	.5	.07
15	R_2	1.0	.13
16	N	1.0	.13
17	N	1.0	.13
18	N	1.0	.13
19	N	1.0	.13
20	N	1.0	.13

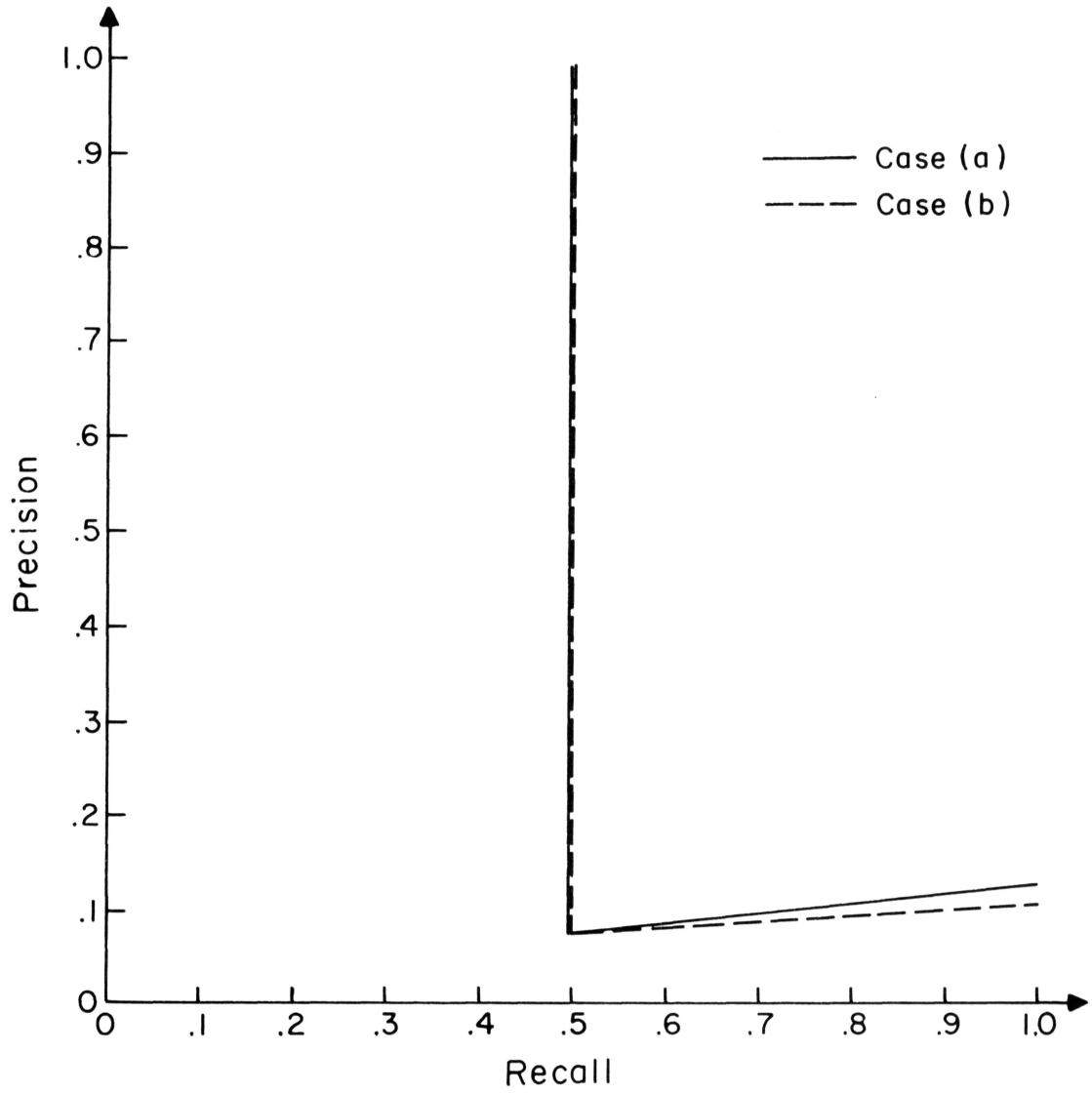
a) Five Documents Compared

Rank	Doc.	R	P
1	R_1	.5	1.0
2	N	.5	.5
3	N	.5	.33
4	N	.5	.25
5	N	.5	.2
6	N	.5	.17
7	N	.5	.14
8	N	.5	.13
9	N	.5	.11
10	N	.5	.1
11	N	.5	.09
12	N	.5	.08
13	N	.5	.08
14	N	.5	.07
15	N	.5	.07
16	N	.5	.06
17	N	.5	.06
18	R_2	1.0	.11
19	N	1.0	.11
20	N	1.0	.11

b) Ten Documents Compared

New Evaluation Measure

Fig. 5



Document Level Graph for Example of Fig.5

Fig. 6

the base set of clusters; BASE2, BASE3, BASE4, and BASE5. BASE2 searches the top two centroids, BASE3 searches the top three centroids, etc. Fig. 7 shows the search results using the standard recall-precision measure, and Fig. 8 shows the results using the new evaluation measure. Also included in each graph are the results of a full search.

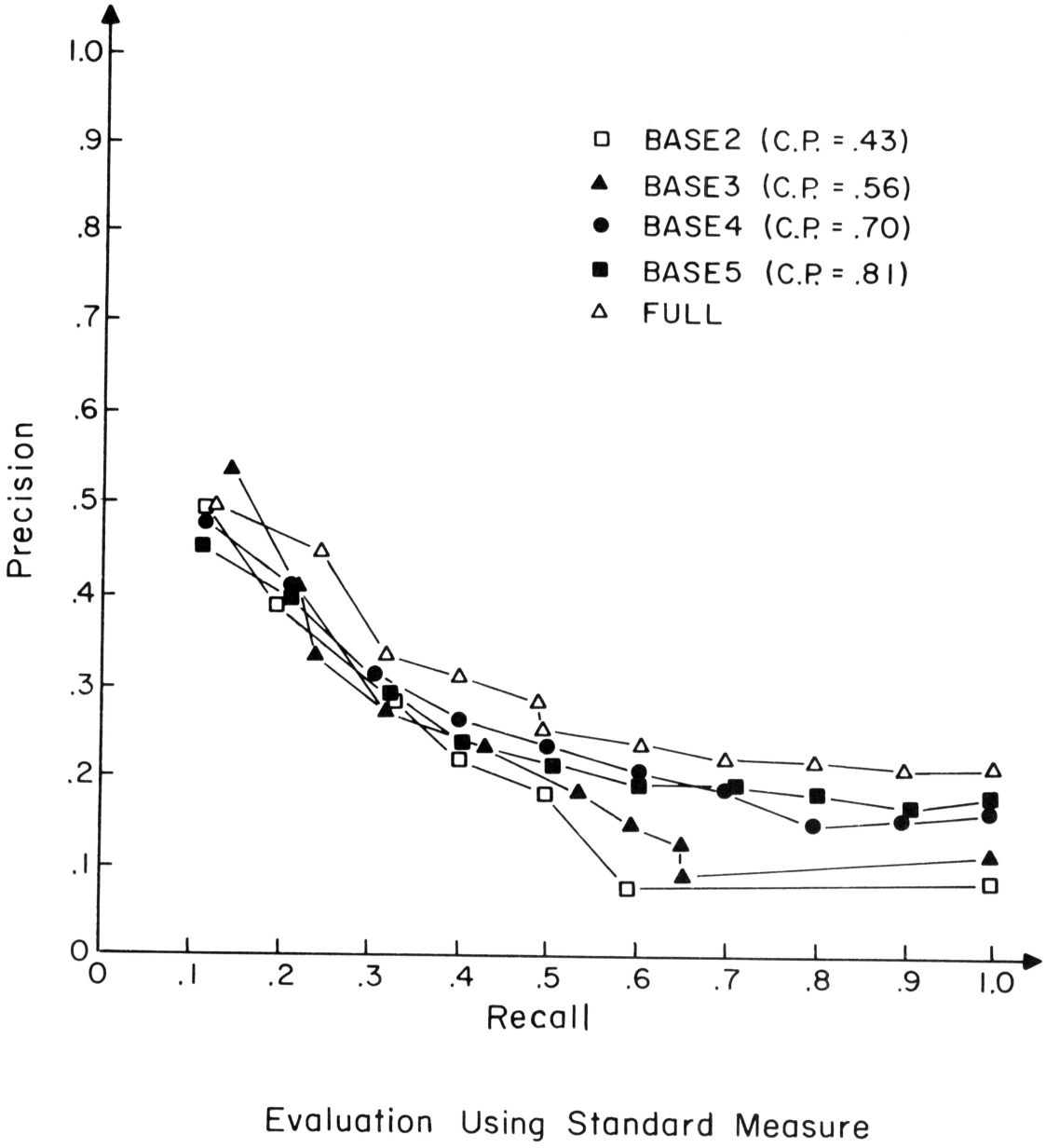
For the highest ranked documents, both curves are almost identical, since the new adjustments usually do not affect the high ranks. However, at the lower ranks the new adjustments favor the searches with the lower correlation percentage. The high recall end of Fig. 7 clearly illustrates the superiority of those searches with a high C.P. This is to be expected since more relevant documents are retrieved with higher C.P. searches.

In Fig. 8 the searches with the highest correlation percentages are still better at the high recall end, but the difference is much smaller than in Fig. 7. In fact, there is very little difference between BASE4, BASE5, and FULL, and between BASE2 and BASE3. Also, the difference between the BASE2 curve and the FULL curve are much less.

When comparing the search results obtained from different sets of clusters, an attempt is made to keep the C.P.'s as equal as possible, even though the new evaluation measure adjusts for differences. However, when the C.P.'s differ by as much as .15 (BASE4 compared to BASE3), the graph of the higher C.P. search can be expected to be higher.

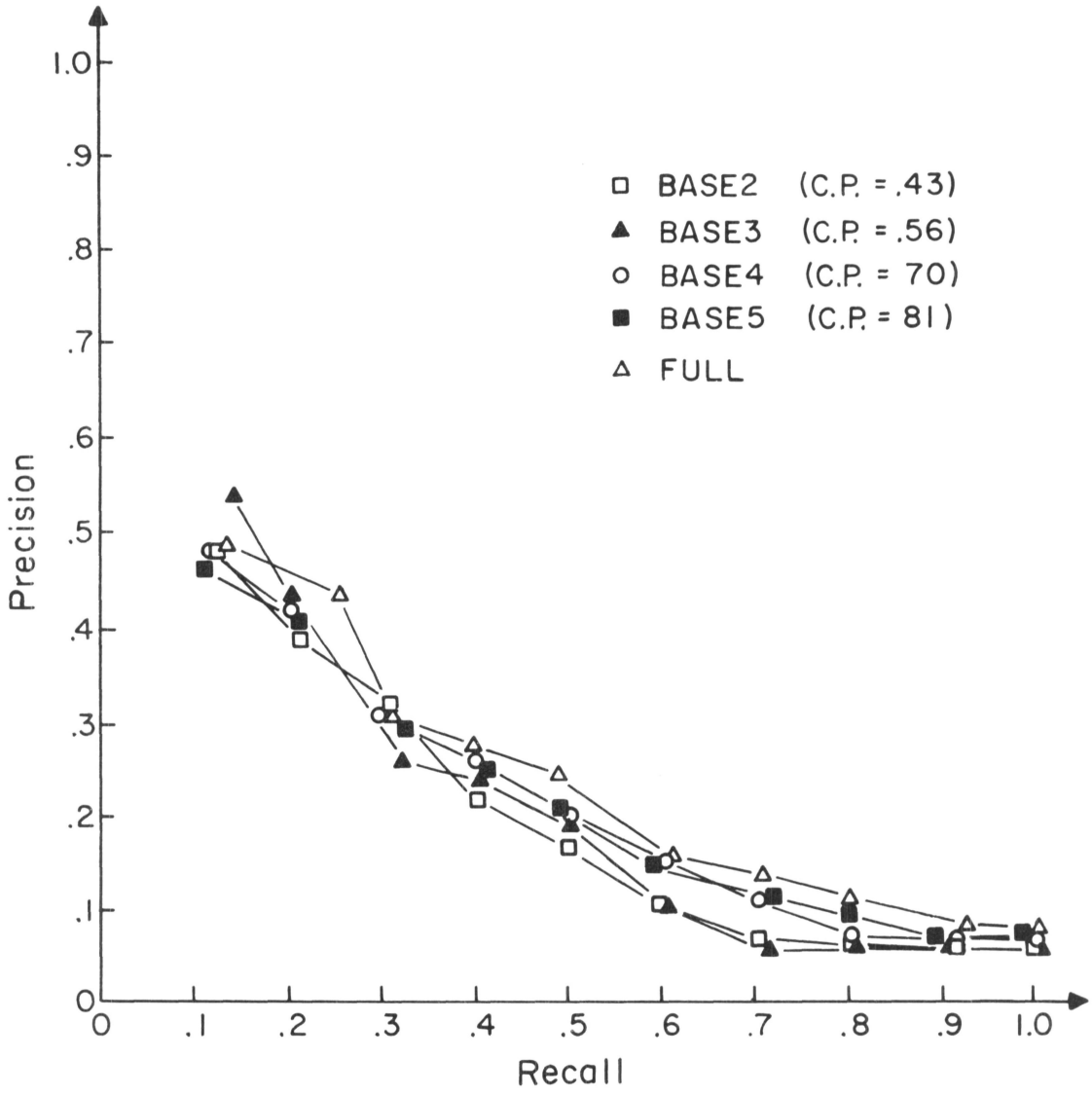
B) Internal Evaluation

The purpose of the internal evaluation is to investigate how changes in the input parameters affect the multi-level search results. All the experiments are conducted on the 82 document ADI collection, and the 200 document Cranfield thesaurus collection. The following parameters are in-



Evaluation Using Standard Measure

Fig. 7



Evaluation Using New Measure

Fig. 8

vestigated:

1) initial cluster generation
   a) correlation
   b) similarity correlation
   c) frequency
   d) random
   e) results from a one-pass algorithm; [5]
2) number of clusters;
3) overlap;
4) cutoff; i.e., percent of documents allowed to be loose at the end of the classification;
5) percent loose clustered on the next iteration.

In addition to evaluating the retrieval results, the internal evaluation should determine how close the classification comes to satisfying the input parameters. However, it is probably better not to "force" a classification algorithm exactly to satisfy input parameters such as the number of clusters and amount of overlap. The algorithm should allow some variation in these parameters depending on the collection used. For example, suppose a collection consists of repetitions of three distinct documents. Then clearly three clusters should be produced, no matter how many are requested. On the other hand, it is difficult to obtain experimental conclusions if the parameters are not controlled. Therefore, the algorithm is designed to keep the parameters within fairly small boundaries. In practice it might be better to relax some of these restrictions somewhat.

Experiments are performed by varying one of the five parameters and comparing the results with those obtained with a base classification. [4]

The parameters for the base classification usually fall somewhere near the middle of the two extremes. Table 1 lists the classification parameters for the ADI collection, and Table 2 illustrates the same parameters for the Cranfield collection. The first two columns specify the number of clusters requested (In) and the number actually produced (Out); the next two columns show the percent overlap requested and the percent produced, and the next two columns specify the percent of documents loose at the end of the algorithm. "Percent loose taken" specifies the percent of loose documents clustered at the end of each iteration. The results of the experiments are shown as document level recall-precision graphs using the new evaluation measure, and the C.P. figures are included when available.

C) Initial Clusters

The initial cluster evalution includes several sets of cluster generations and centroid searches. For the ADI collection, the following experiments are performed:

1) 25% loose, $\leq$ .5% overlap (Fig. 9);
   a) FREQ (frequency),
   b) CORR2 (correlation),
   c) FASTD (input from one-pass algorithm, 0% overlap);
2) 25% loose, 1-3% overlap (Fig. 10);
   a) CORRI (correlation),
   b) RANDOM (random),
   c) FASTO (input from one-pass algorithm, overlapping clusters);
3) 40% loose, 0% overlap (Fig. 11);
   a) OVRO (correlation),
   b) OVEROSIM (similarity correlation)
   c) OVEROFREQ (frequency).

Run	No. of Clusters In	Out	Percent Overlap In	Out	Cutoff (% Loose) In	Out	Percent Loose Taken	Approx. Time (min.)	No. of Iter	No. of Cycles
BASE	10	9	5	.3	10	12	40	1.1	3	6
CLUST5	5	4	5	1.2	10	13	40	1.0	5	12
CLUST15	15	15	5	.4	10	4	40	.8	1	4
OVRO	10	9	0	0	10	12	40	.7	3	5
OVR10	10	9	10	.3	10	12	40	.8	3	7
OVR15	10	8	15	13.3	10	12	40	1.3	4	12
CUTO	10	10	5	.1	0	2	40	1.0	5	9
CUT20	10	9	5	.2	20	23	40	.7	2	5
CUT30	10	9	5	.2	30	23	40	.7	2	5
LSE20	10	9	5	.2	10	12	20	1.0	5	9
LSE60	10	9	5	.4	10	3	60	.8	3	6
LSE80	10	9	5	0	10	6	80	.8	2	5
BASEFREQ	10	10	5	0	10	12	40	1.0	3	6
OVROFREQ	10	10	0	0	10	12	40	1.0	2	6
BASESIM	10	9	5	0	10	11	40	.9	4	9
OVROSIM	10	9	0	0	10	11	40	.9	4	9
FREQ	9	8	5	.5	5	7	25	1.7	7	17
CORR2	9	7	5	.2	5	7	25	1.3	6	17
CORR1	9	8	5	1.1	5	8	25	1.1	6	13
SIM	9	9	5	1.0	10	11	25	1.4	6	14
RANDOM	9	11	5	2.2	5	4	25	1.4	4	15
ROCCHIO	-	7	-	1.5	-	-	-	-	-	-
BEST	10	9	2	0	10	3	60	.7	3	6
FASTO	9	11	5	3.3	5	8	25	1.4	3	13
FASTD	9	11	0	0	5	5	25	1.0	3	7
MDJ	-	14	-	0	-	-	-	-	-	-

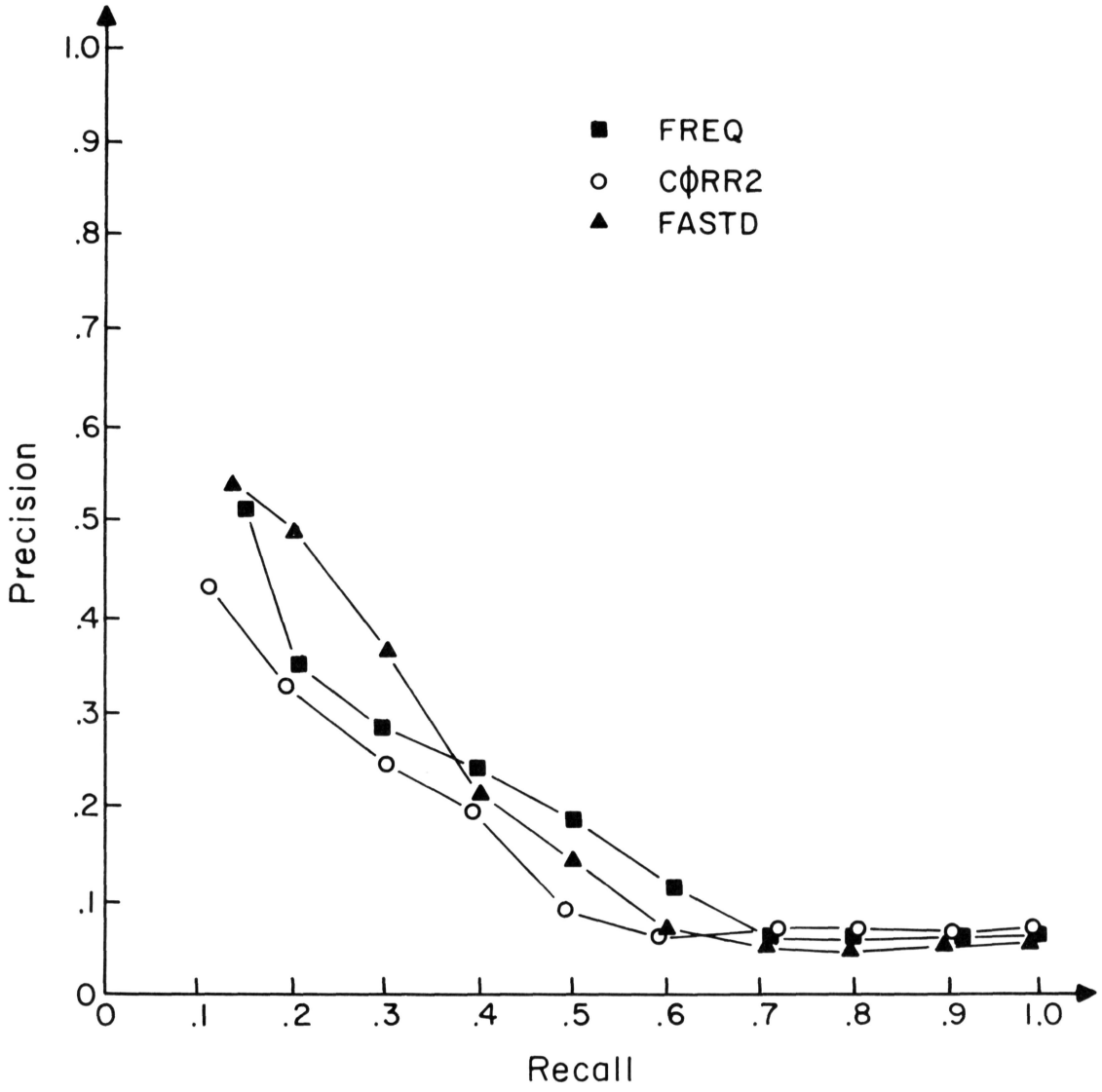
ADI Clusters

Table 1

Run	No. of Clusters In	Out	Percent Overlap In	Out	Cutoff (%Loose) In	Out	Percent Loose Taken	Approx. Time (min.)	No. of Iter	No. of Cycles
BASE	15	12	5	5.4	10	13	40	6.2	6	47
CLUST10	10	9	5	6.1	10	13	40	2.7	3	26
CLUST20	20	20	5	1.7	10	14	40	3.9	3	20
OVR0	15	13	0	0	10	9	40	2.5	5	17
OVR10	15	14	10	15.1	10	8	40	5.6	4	31
OVR15	15	14	15	15.3	10	8	40	6.0	4	35
CUT0	15	13	5	6.2	0	3	40	6.6	4	37
CUT20	15	13	5	6.4	20	16	40	4.5	4	29
CUT30	15	14	5	1.1	30	27	40	2.4	3	17
LSE20	15	13	5	4.1	10	11	20	9.1	9	69
LSE60	15	13	5	1.5	10	11	60	2.3	3	14
LSE80	15	15	5	1.3	10	9	80	2.0	2	10
BASEFREQ	15	13	5	3.8	10	9	40	3.5	5	24
OVROFREQ	15	12	0	0	10	10	40	2.4	4	13
BASESIM	15	13	5	5.7	10	12	40	4.4	4	30
OVROSIM	15	15	0	0	10	13	40	2.8	3	13
RANDOM	15	18	5	7.1	10	13	40	6.4	3	30
CORR	15	10	5	3.5	5	8	50	4.2	4	25
FREQ	15	13	5	8.8	5	4	50	6.0	4	35
ROCCHIO	-	18	-	2.9	-	-	-	-	-	-
BEST	15	15	2	.5	10	16	60	3.6	3	23
RANCLUST	-	15	-	3.2	-	-	-	-	-	-

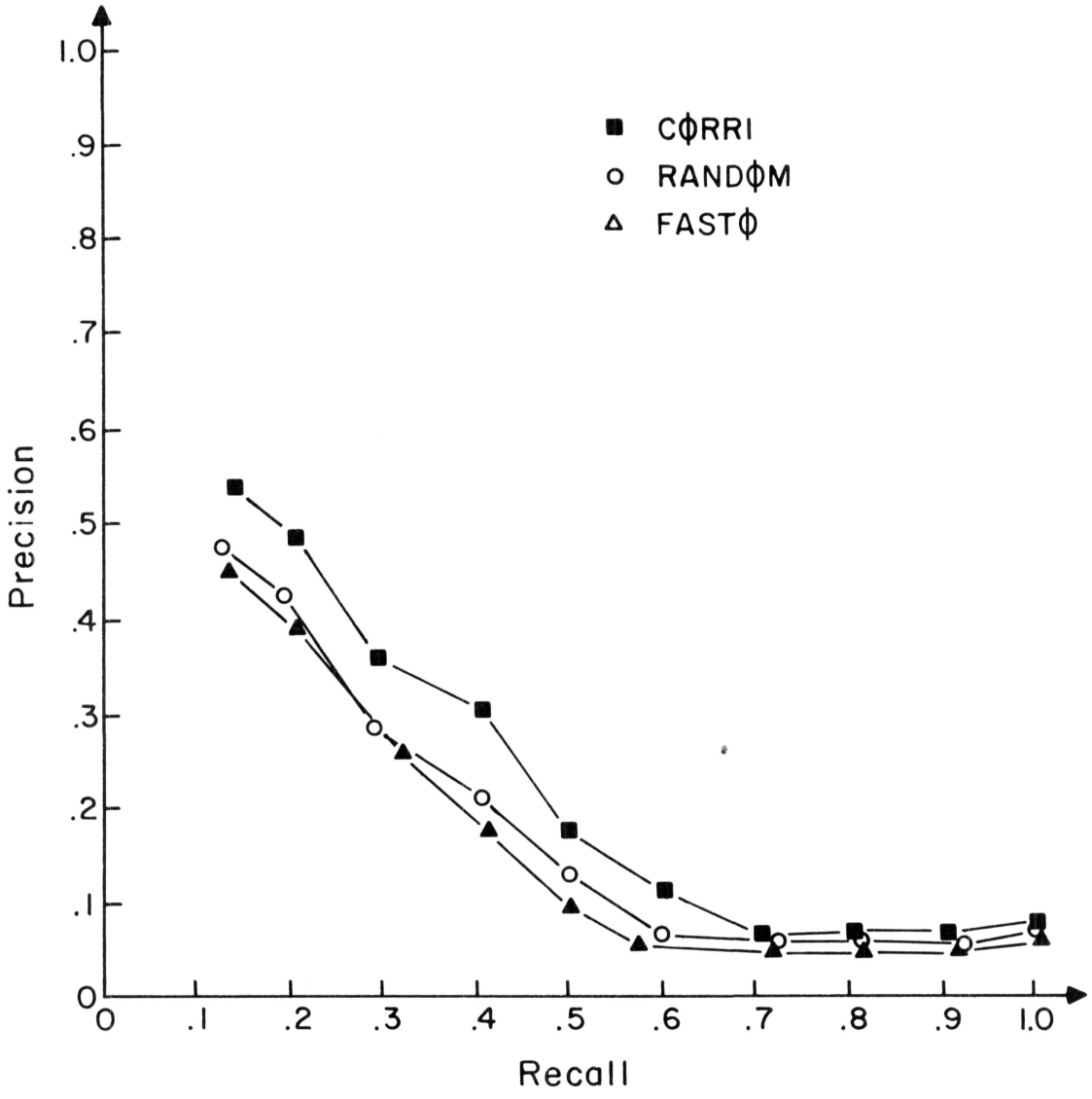
Cranfield Clusters

Table 2



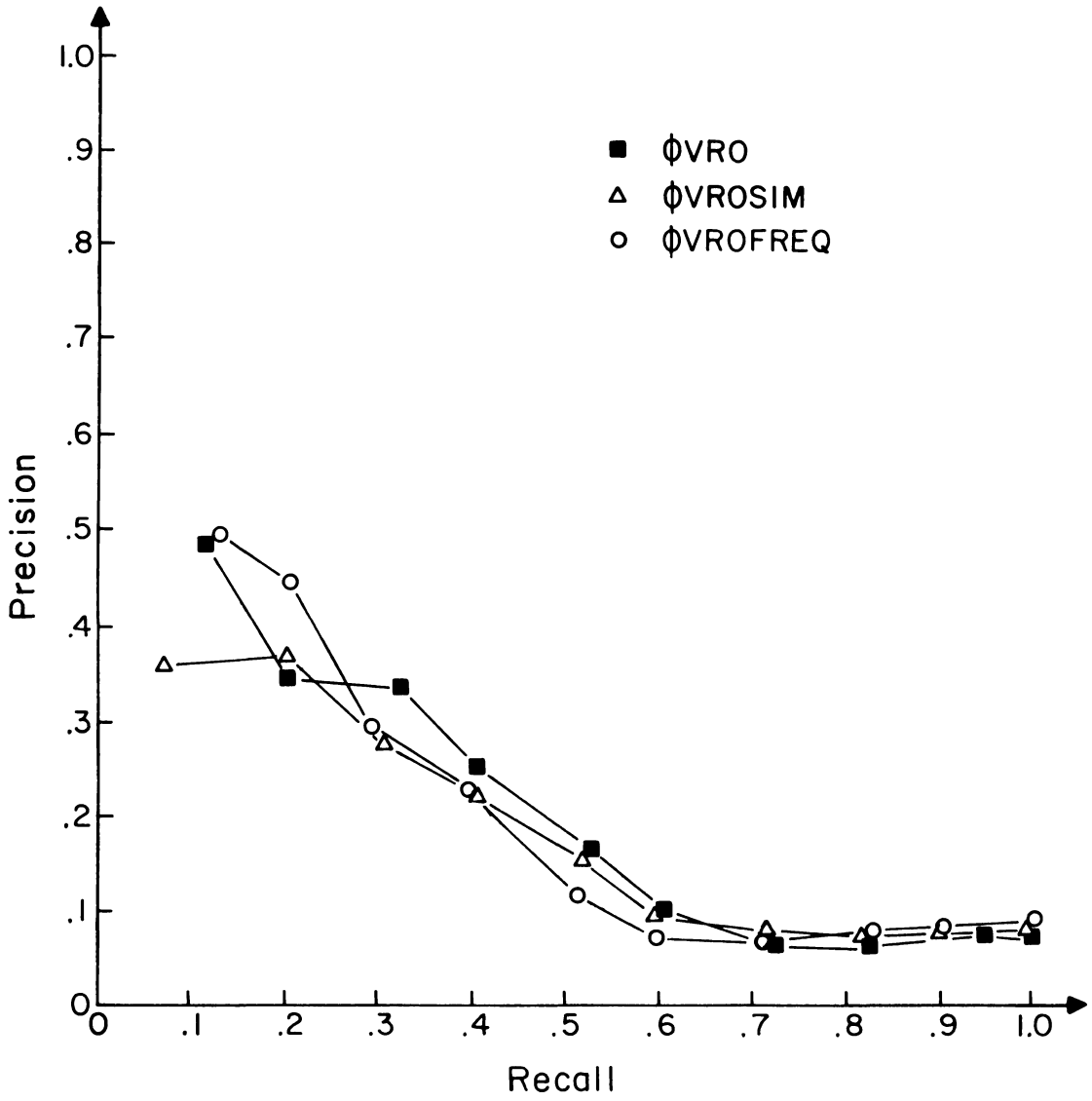
ADI Initial Clusters
25% Loose ≤ .5% Overlap

Fig. 9



ADI Initial Clusters
25% Loose, 1-3% Overlap

Fig. 10



ADI Initial Clusters
40% Loose, 0% Overlap

Fig. 11

With the exception of 1(c) (0% overlap requested), within each of the three experiments, all of the input parameters are identical. The only difference is the method used to generate the initial clusters. All evaluations are mady by visual inspection of the document level recall-precision curves. The following notation is used to indicate the results:

a) = two methods give approximately similar results, or it is impossible to determine from the graphs which method is better;

b) > the first method is better than the second;

c) >> the first method is much better than the second.

The results of the three ADI experiments with initial clusters are as follows:

1) (FREQ = FASTD) > CORR2

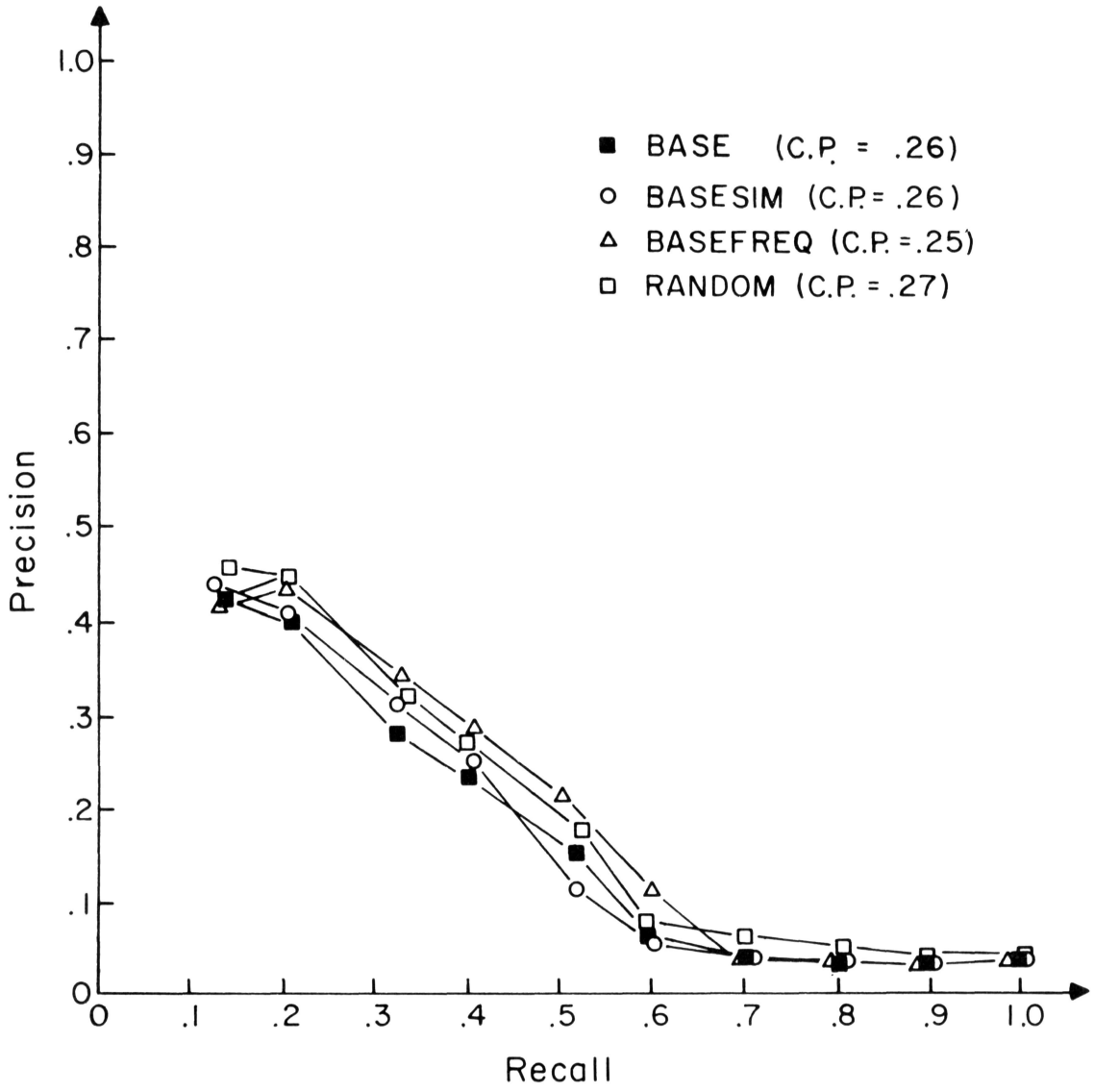
2) CORR1 > (RANDOM=FASTO)

3) OVRO = OVROSIM = OVROFREQ.

Unfortunately, the results do not show that any method is consistently better than any other. The only conclusion is that the frequency, the similarity correlation, and the use of disjoint clusters from a one-pass algorithm never perform worse than any other method, while the random method and overlapping clusters place last in their only test.

The following initial cluster experiments are performed with the Cranfield collection:

1) 40% loose, 4-7% overlap (Fig. 12);
   a) BASE (correlation),
   b) BASESIM (similarity correlation),



Cranfield Initial Clusters
40% Loose, 4-7% Overlap

Fig. 12

c) BASEFREQ (frequency),

d) RANDOM (random).

2) 40% loose, 0% overlap (Fig. 13);

a) OVRO (correlation),

b) OVROSIM (similarity correlation),

c) OVROFREQ (frequency).

The results of these two initial cluster experiments are:

1) (RANDOM = BASEFREQ) > (BASE = BASESIM)

2) OVRO >> OVROFREQ > OVROSIM

Once again the results are inconclusive. The correlation method performs better than the other two methods in the nonoverlapping case, but does not do as well as the random or frequency methods in the first experiment. Surprisingly, the random method actually performs better than the correlation or similarity correlation methods. This indicates that the algorithm is insensitive to the type of initial clusters used, or that none of the methods used so far is very good. Additional experiments must be carried out using different methods of initial cluster generation. One method which can be tried involves running the algorithm twice. The first time the initial clusters are generated using the random method. The clusters produced by this run are then used as the intial clusters for the second run.

D) Number of Clusters

In the remainder of the experiments, the correlation method of initial cluster generation is used, and unless otherwise specified, all other parameters are identical to the BASE run. For any given collection, the

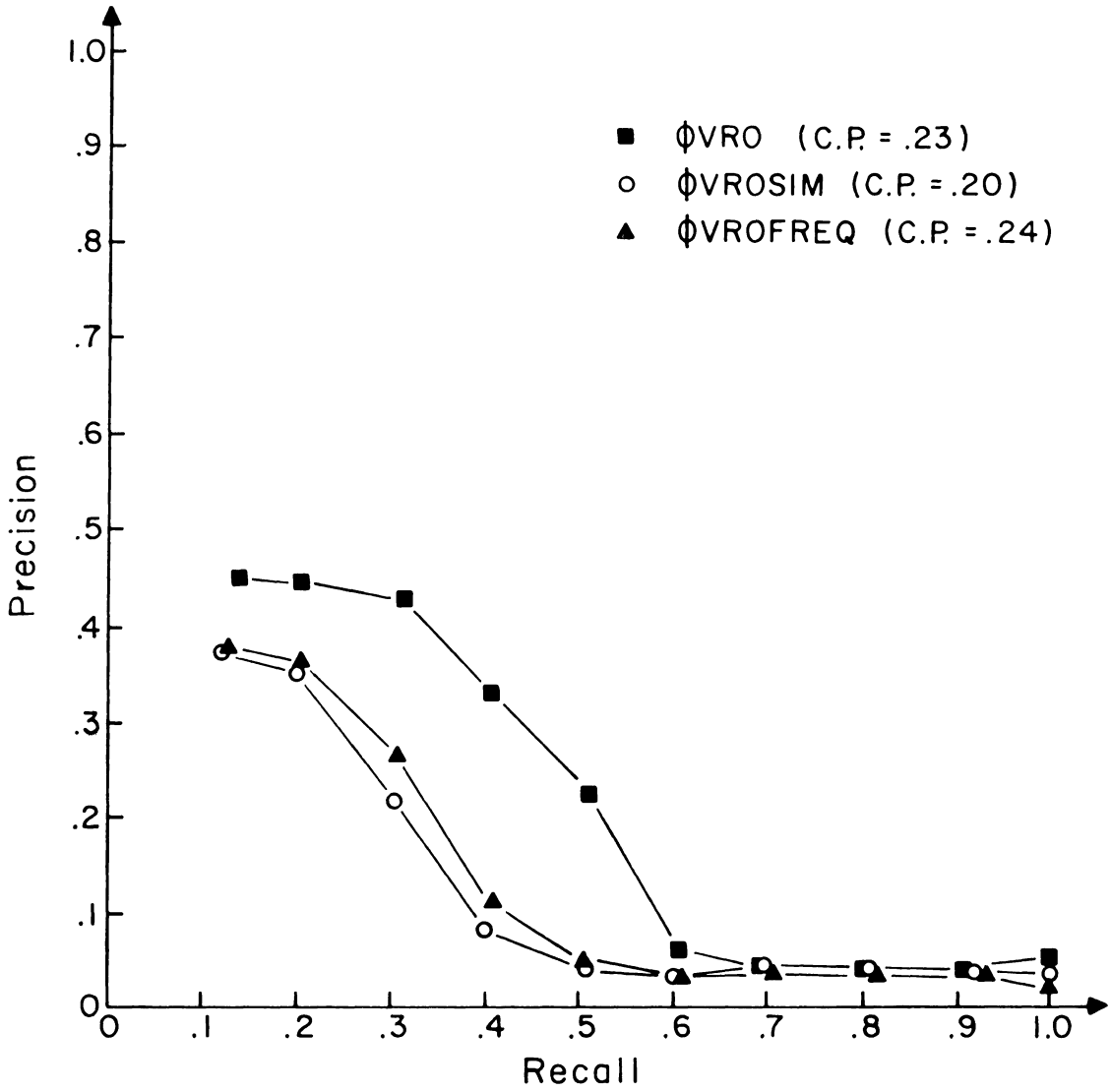

1.0
.9
.8
.7
.6
.5
.4
.3
.2
.1
0
Precision
0
.1
.2
.3
.4
.5
.6
.7
.8
.9
1.0
Recall
■ ϕVRO (C.P. = .23)
○ ϕVROSIM (C.P. = .20)
▲ ϕVROFREQ (C.P. = .24)
Cranfield Initial Clusters
40% Loose, 0% Overlap

Fig. 13

number of clusters can be chosen to minimize the search time, assuming that all clusters contain the same number of documents. This assumption is not too invalid since the cluster sizes are maintained between one-half and twice the average cluster size.

Consider a collection of N documents classified into m clusters, where each cluster contains on the average n documents. For a centroid search which looks at the documents in the top q clusters, the search time is $ST = m + q \cdot n$. Assuming $\phi = 0$, $n = N/m$ and $ST = m + q \cdot N/m$. The search time is minimized for $dST/dm = 0$.

$$dST/dm = 1 - q \cdot N/m^2 . \quad dST/dm = 0 \Rightarrow m = \sqrt{q \cdot N}$$

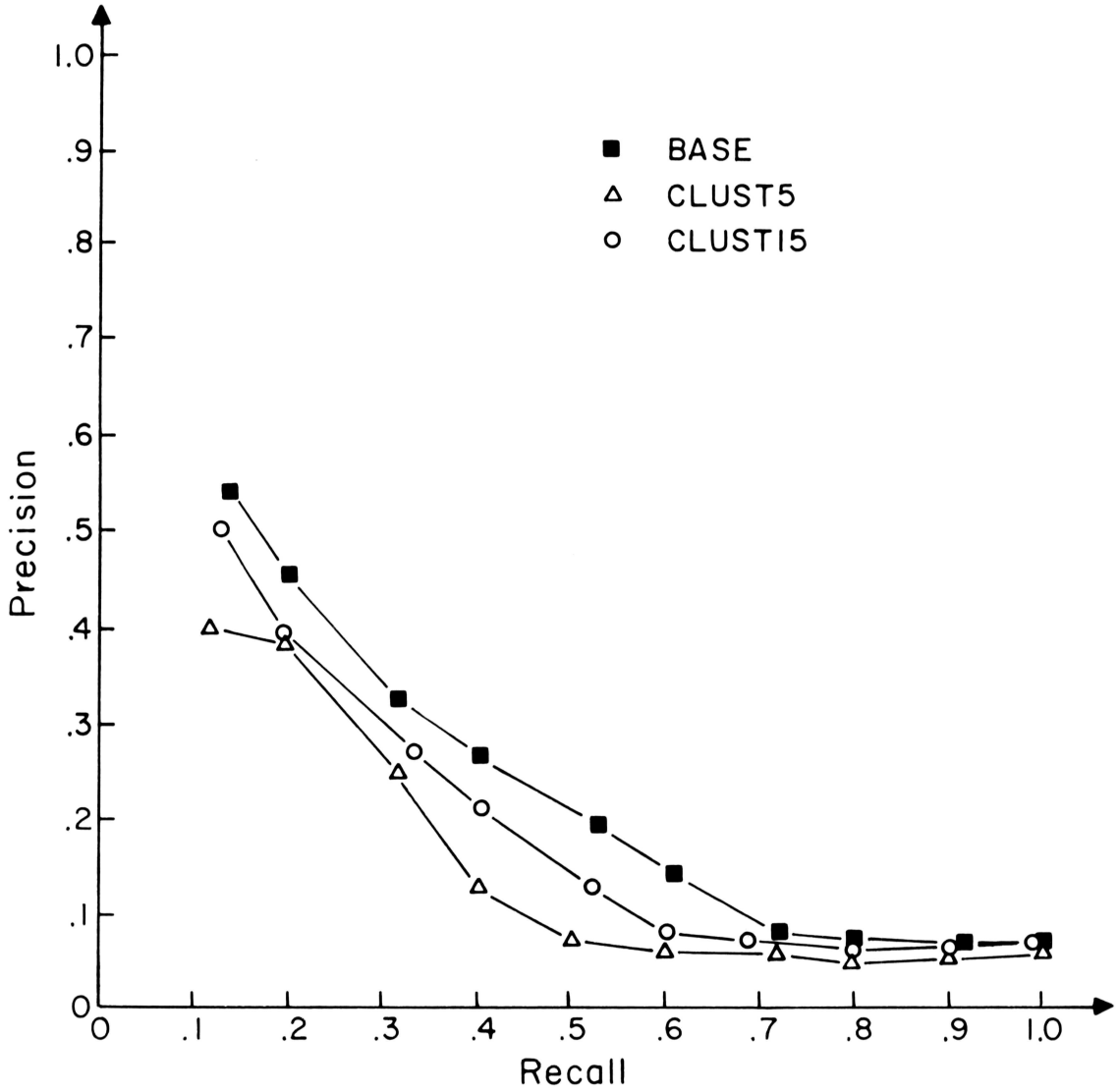
For the ADI collection, ST is minimized with $\sqrt{82} = 9$ clusters if one centroid is searched, and $\sqrt{164} = 13$ clusters if two centroids are searched. Of course, there is no guarantee that optimizing the search time yields the best results. The following experiments are performed with the ADI collection: (Fig. 14)

1) BASE (9 clusters),

2) CLUST5 (4 clusters),

3) CLUST15 (15 clusters).

The parameters for the centroid search allow the number of centroids searched per query to vary for different queries. A specified minimum is always searched for each query, but additional centroids may also be searched if the query-centroid correlation is close enough to the correlations of the centroids already chosen. Thus, the average number of centroids searched is usually higher than the minimum. In experiment 1 a minimum of two centroids is searched, while in experiment 2 one centroid is searched, and in



ADI Number of Clusters

Fig. 14

experiment 3 three centroids are searched. The minimum number of centroids searched is chosen to keep the search times approximately equal. The approximate search times for each experiment are:

1) ST = 9 + 2•82/9 = 9 + 164/9 = 9 + 18 = 27;

2) ST = 4 + 1•82/4 = 4 + 21 = 25;

3) ST = 15 + 3•82/15 = 15 + 246/15 = 15 + 16 = 31.

The results of the experiment show that BASE > CLUST15 > CLUST5 .

For the Cranfield collection, the following experiments are made: (Fig. 15)

1) BASE (12 clusters),

2) CLUST10 (9 clusters),

3) CLUST20 (20 clusters).

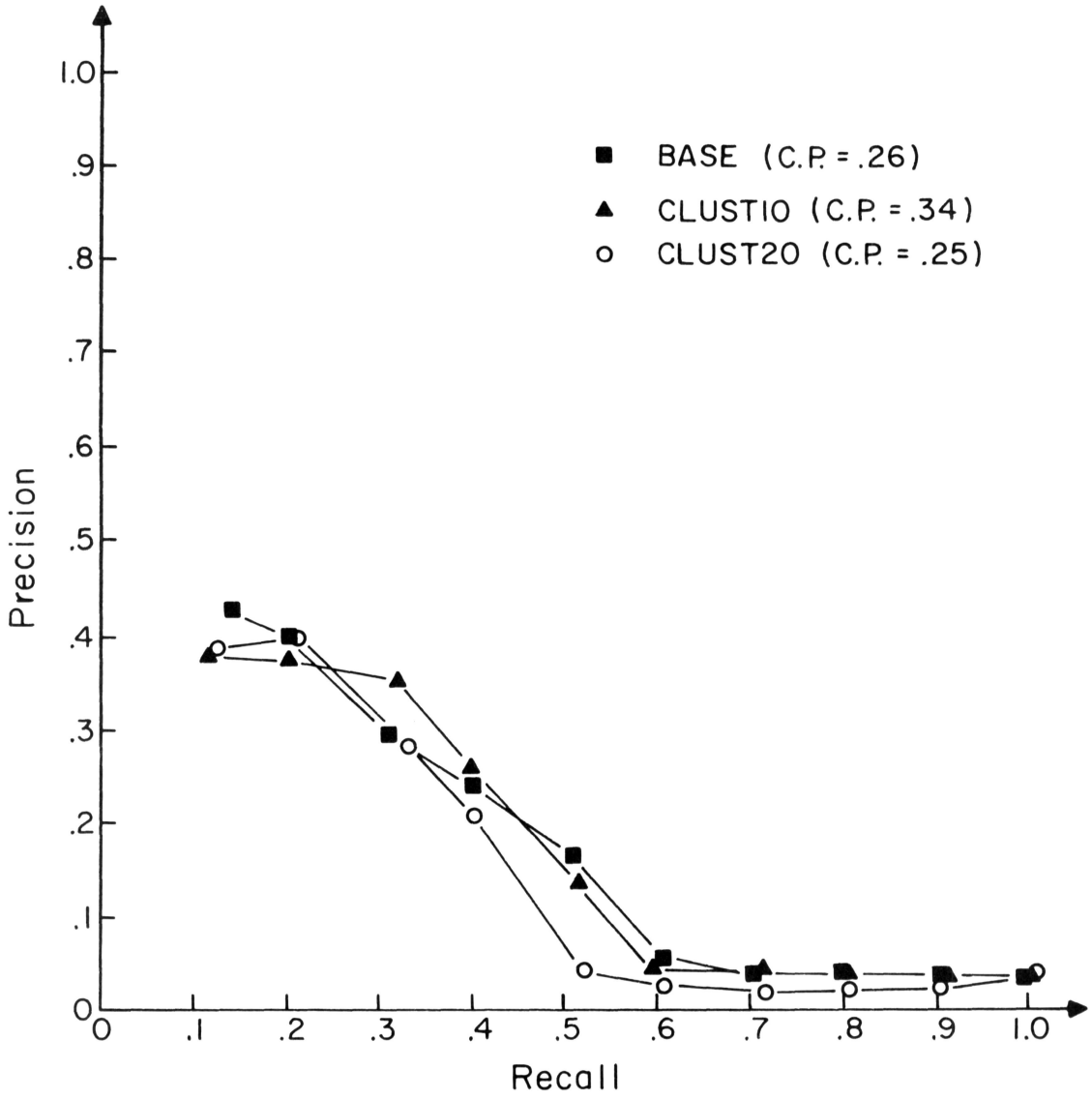
All of these searches are made using a minimum of one centroid, so the approximate search times may be quite different. Fortunately, the actual C.P.'s are available, and they can be compared with the ST figures:

1) ST = 12 + 1•200/12 + 17 = 29,
   C.P. = .26 => ST' (actual search time) = .26•200 = 52;

2) ST = 9 + 1•200/9 = 9 + 22 = 31,
   C.P. = .34 => ST' = .34•200 = 68;

3) ST = 20 + 1•200/20 = 20 + 10 = 30,
   C.P. = .25 => ST' = .25•200 = 50.

All the ST values are lower than the actual search time, but this is mainly due to the fact that more than one centroid is often used, and the overlap is not 0. Fig. 15 shows that (BASE = CLUST10) > CLUST20 . Unlike



Cranfield Number of Clusters

Fig. 15

the ADI results, the search with the smallest number of clusters does better than the search with the largest number. This is probably due to the larger number of clusters used with the ADI CLUST15. At any rate, both the ADI and Cranfield results indicate that the middle number of clusters does best. Notice that both sets of BASE clusters contain very close to $\sqrt{N}$ clusters ($\sqrt{82} = 9$, $\sqrt{200} = 14$).

E) Overlap

An inspection of Tables 1 and 2 shows that the amount of overlap produced is very often much different from the amount requested. The requested overlap is varied from 0% to 15% in steps of 5%, but in both collections several of the requests were not satisfied. However, the following comparisons can be made for the ADI experiments: (Fig. 16)

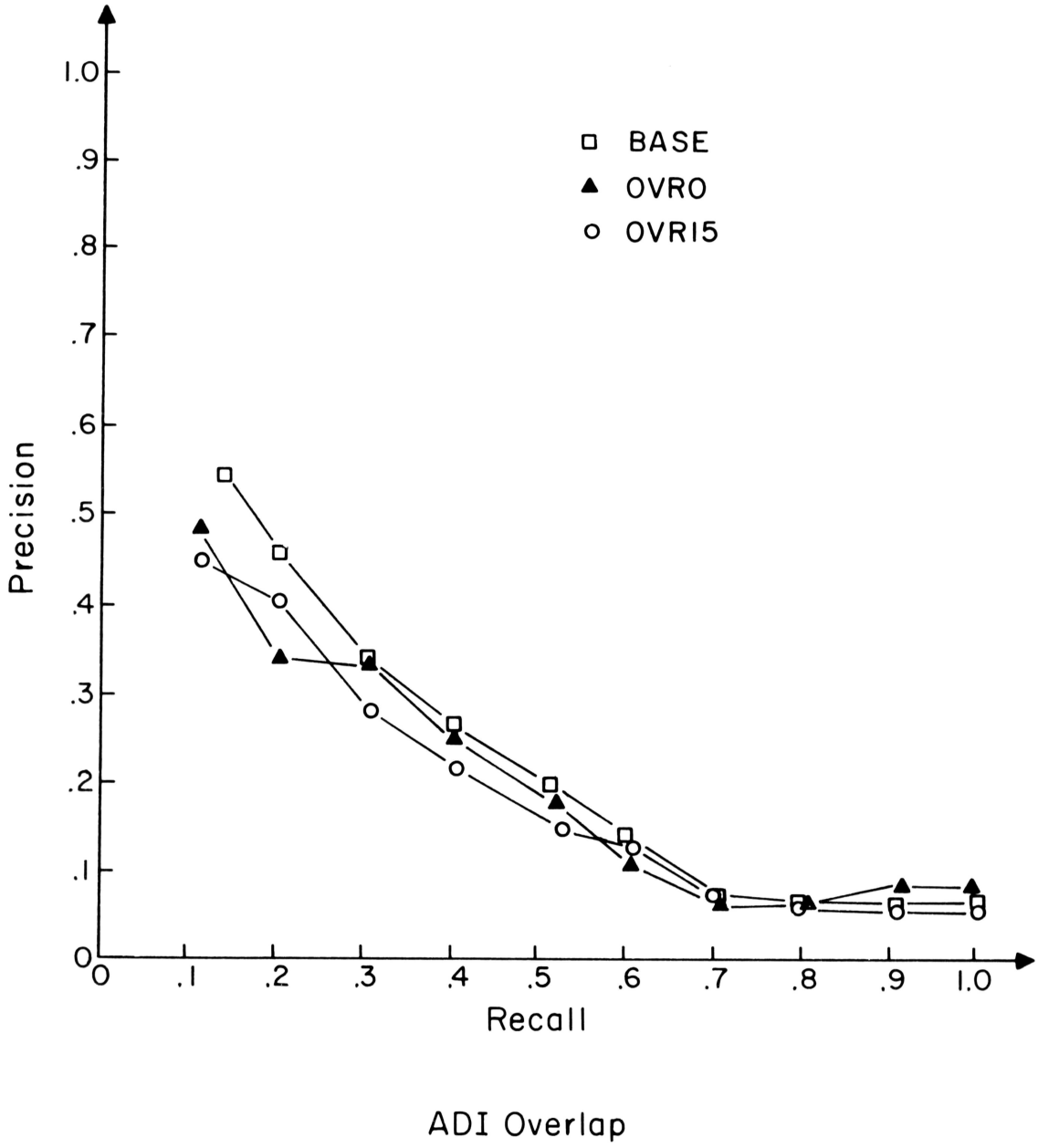
1) BASE (.3%),
2) OVRO (0%),
3) OVR15 (13.5%).

The results indicate that BASE > (OVRO=OVR15) . Thus, for the ADI collection a small amount of overlap performs better than a large amount of overlap and better than 0 overlap.

The following experiments are performed for the Cranfield collection: (Fig. 17)

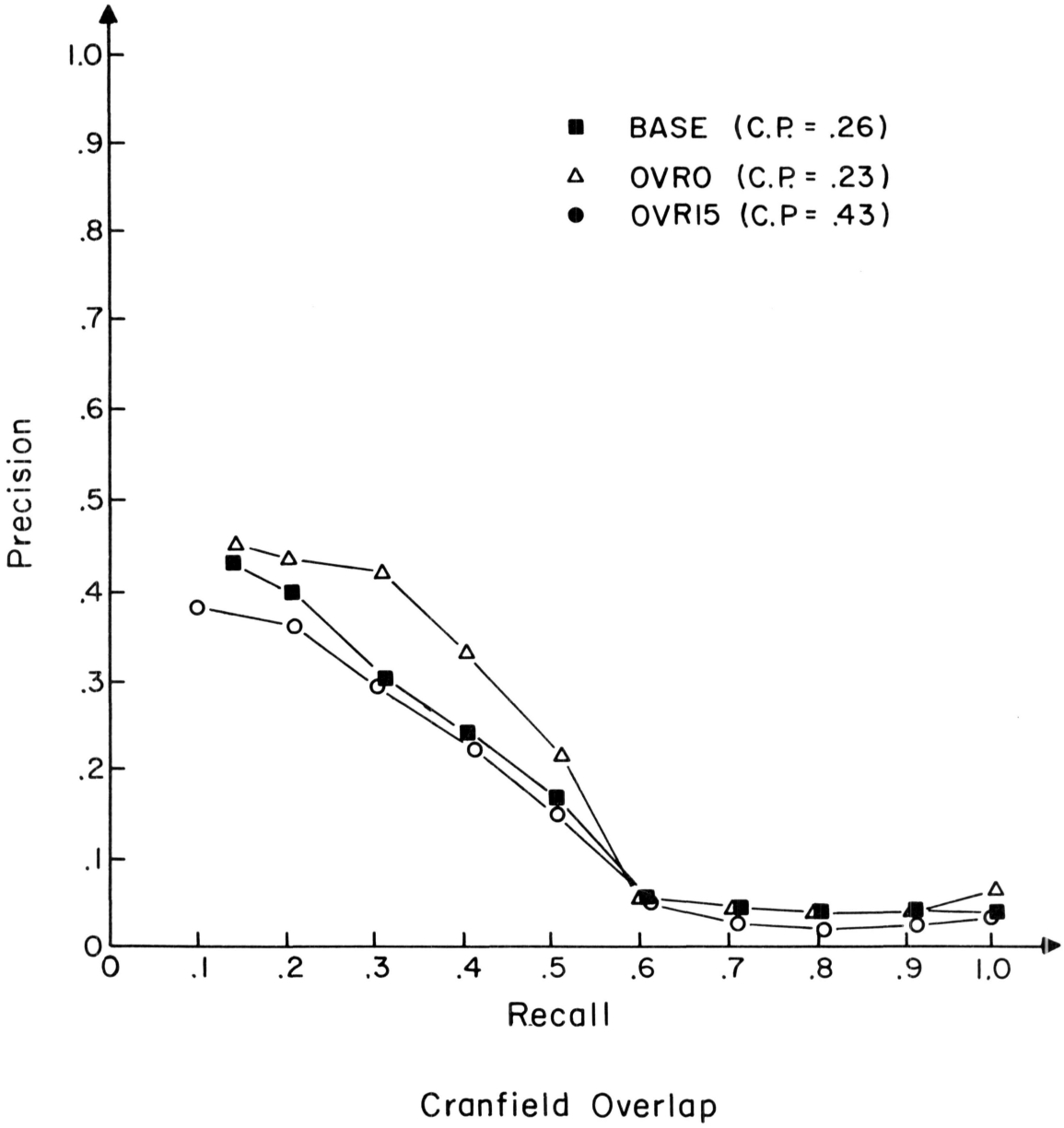
1) BASE (5.4%),
2) OVRO (0%),
3) OVR15 (15.3%).

These results indicate that OVRO >> BASE > OVR15 . Once again the clusters



ADI Overlap

Fig. 16



Cranfield Overlap

Fig. 17

with a very large amount of overlap perform poorly, even though the C.P. for OVR15 is about twice as great as the C.P. for BASE and for OVR0. However, a surprising result is the better performance of the clusters with 0% overlap compared to those with 5% overlap. Since the best ADI results are obtained with only .3% overlap, the conclusion is that a very small amount of overlap is helpful, but more than 5% is too much. More experiments need to be run using an overlap between 0—5%. It appears that the best results should be obtained in this range.

F) Cutoff

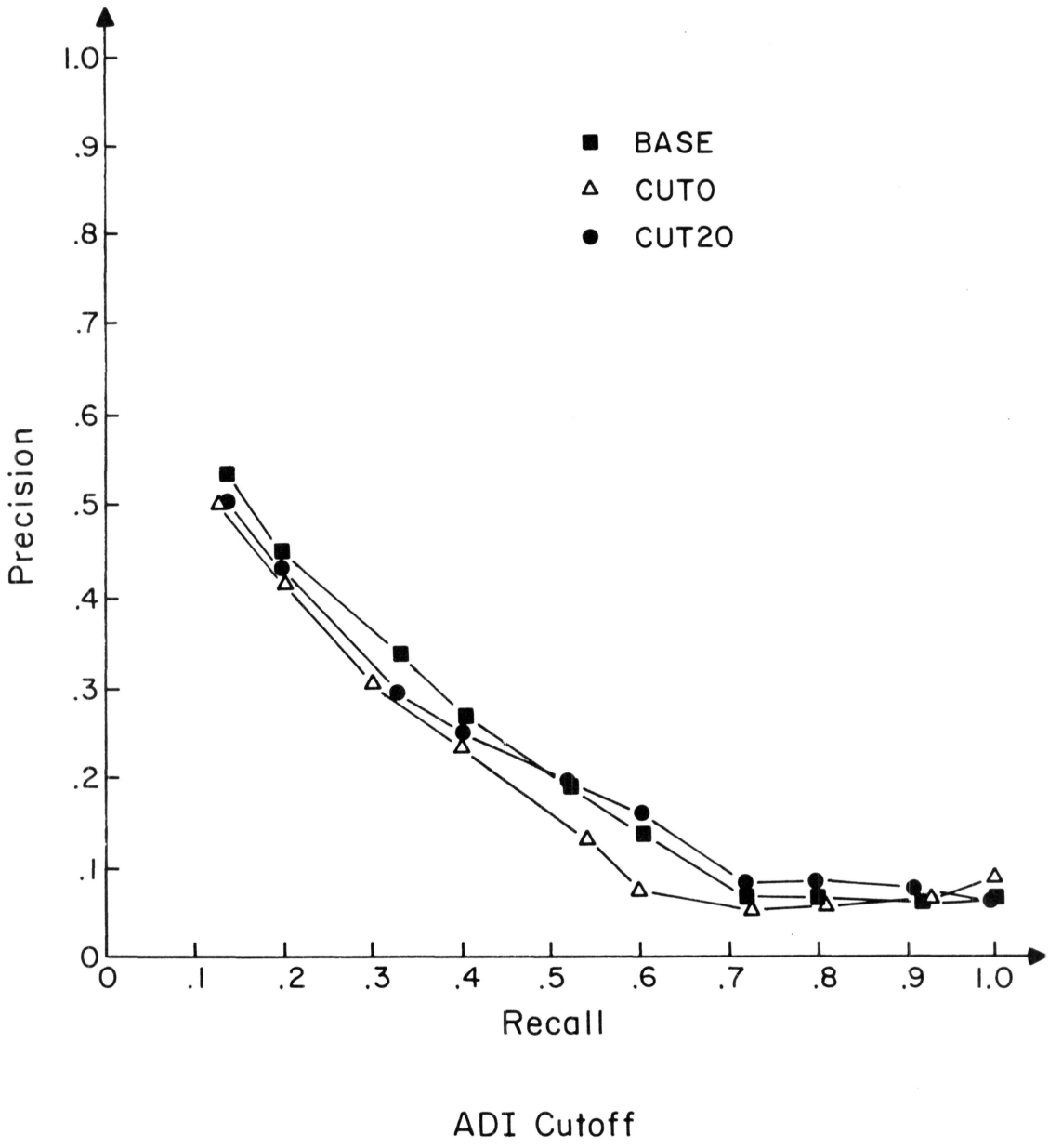
The value of the cutoff determines how many documents are left loose at the end of the algorithm. In all cases, the loose documents are blended into the nearest cluster. The following experiments are compared for the ADI collection: (Fig. 18)

1) BASE (12% loose),
2) CUT0 (2.5% loose),
3) CUT20 (23% loose).

The main advantage of a high cutoff is that it allows the algorithm to terminate faster. The cutoff should be chosen as high as possible to shorten the cluster time, without seriously affecting the search results. The ADI results show that (BASE = CUT20) > CUT0 . This is surprising since it is expected that the lower cutoff would produce better results.

The following experiments are run on the Cranfield collection: (Fig. 19)

1) BASE (13% loose),
2) CUT0 (3.0% loose),
3) CUT20 (16% loose),



ADI Cutoff

Fig. 18

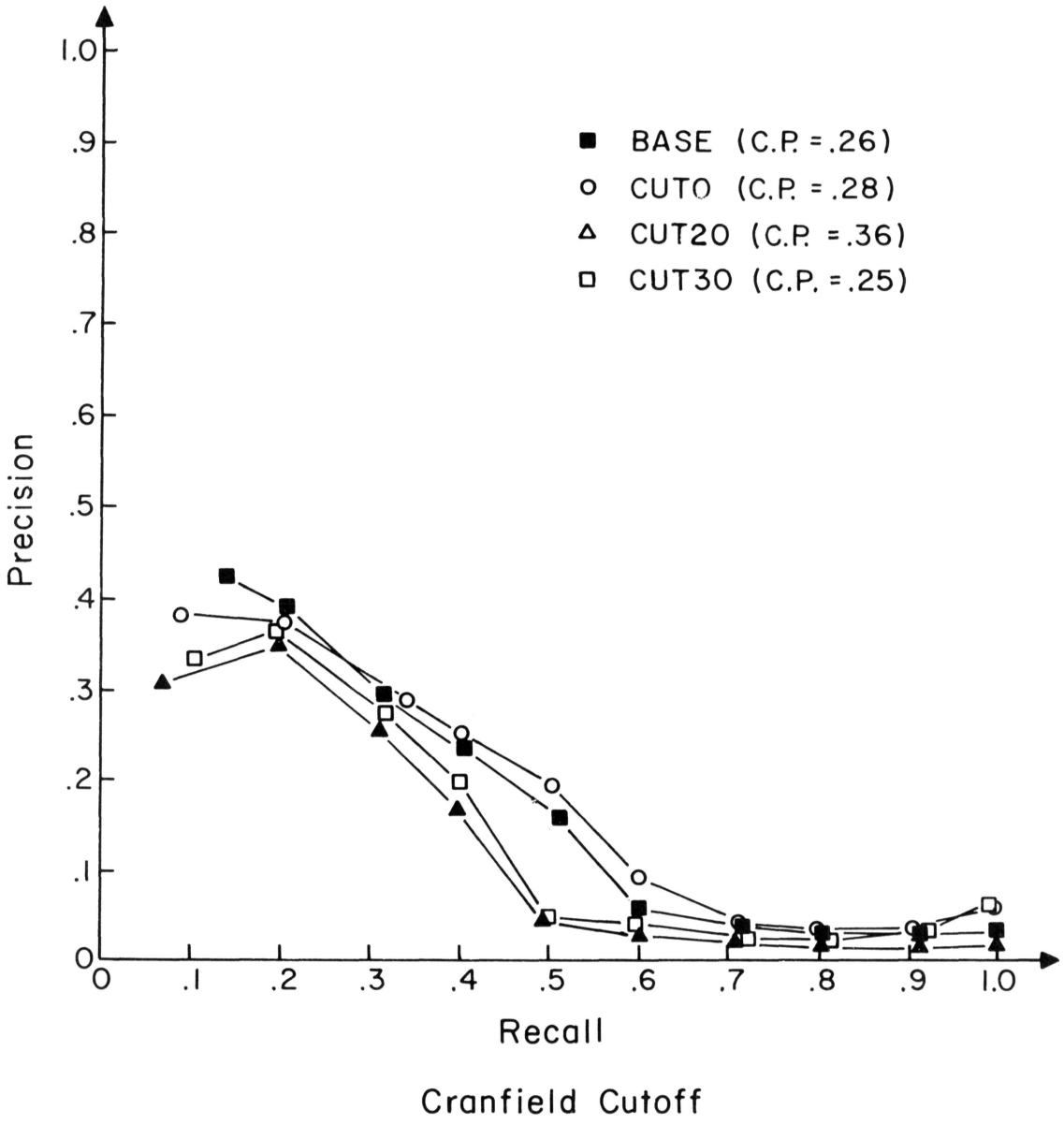

BASE (C.P. =.26)
CUTO (C.P. =.28)
CUT20 (C.P. =.36)
CUT30 (C.P. =.25)
Precision
1.0
.9
.8
.7
.6
.5
.4
.3
.2
.1
0
0
.1
.2
.3
.4
.5
.6
.7
.8
.9
1.0
Recall
Cranfield Cutoff

Fig. 19

4) CUT30 (27% loose).

Notice that the overlap for CUT30 is only 1.1%, while for the other runs it is 5-6%. The experiments on overlap indicate that 1% is probably better than 5-6%. Thus, CUT30 might do well because of its overlap. The results show that (BASE = CUT0) > CUT30 > CUT20 .

The Cranfield results favor the lower cutoff values, and it is possible that CUT20 would have done better than CUT30 if their overlaps were equal. However, the experiment with 3% loose does not do better than the run with 13% loose. Since the lower cutoffs do not perform as well in the ADI experiments, the best cutoff appears to be 12-13%.

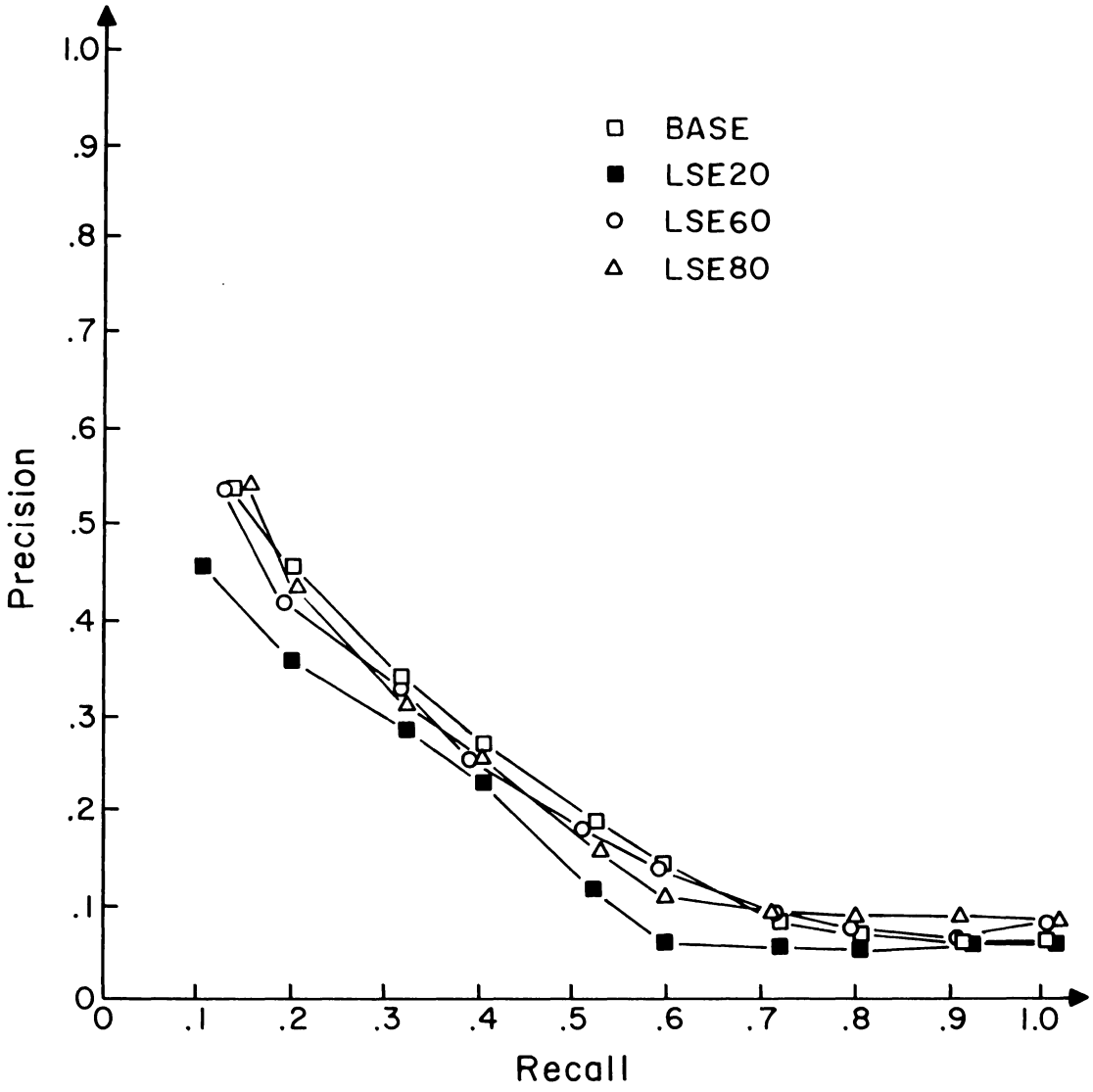
G) Percent Loose Clustered

This parameter controls the percent of the loose documents that are clustered at the beginning of the next iteration. Thus, 40% means that 40% of the remaining loose documents are assigned to clusters after the first cycle of the next iteration. For both the ADI and Cranfield collections the following experiments are performed:

1) BASE (40%),
2) LSE20 (20%),
3) LSE60 (60%),
4) LSE80 (80%).

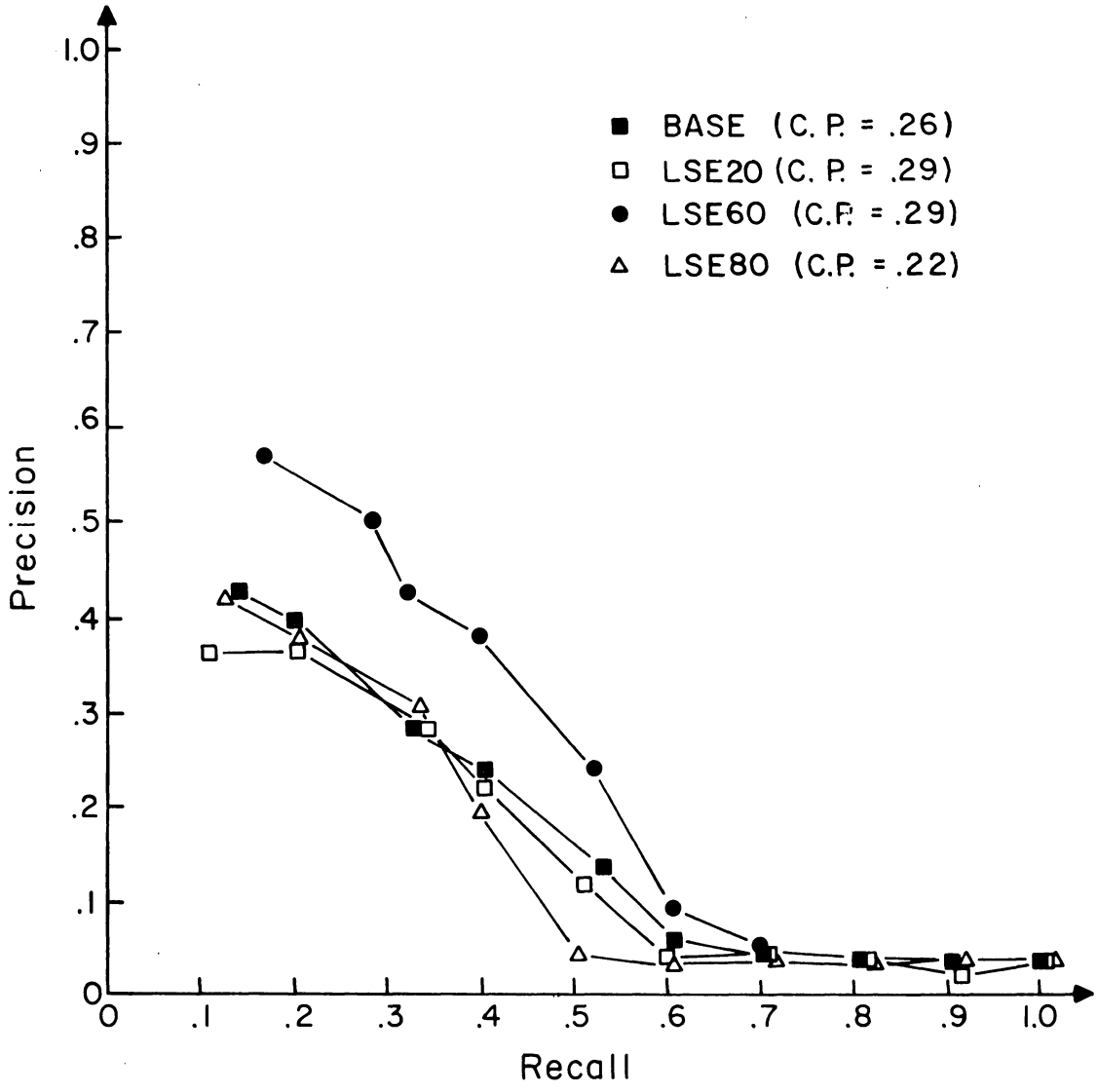
The ADI results appear in Fig. 20 and the Cranfield results in Fig. 21. For the ADI, (BASE = LSE60 = LSE80) > LSE20, and for the Cranfield, LSE60 >> BASE > (LSE80) .

In both cases LSE20 is poorest, but LSE60 is much better than the



ADI Percent Loose Clustered

Fig. 20



Cranfield Percent Loose Clustered

Fig. 21

others for the Cranfield results; therefore, 60% loose gives the best results over both collections.

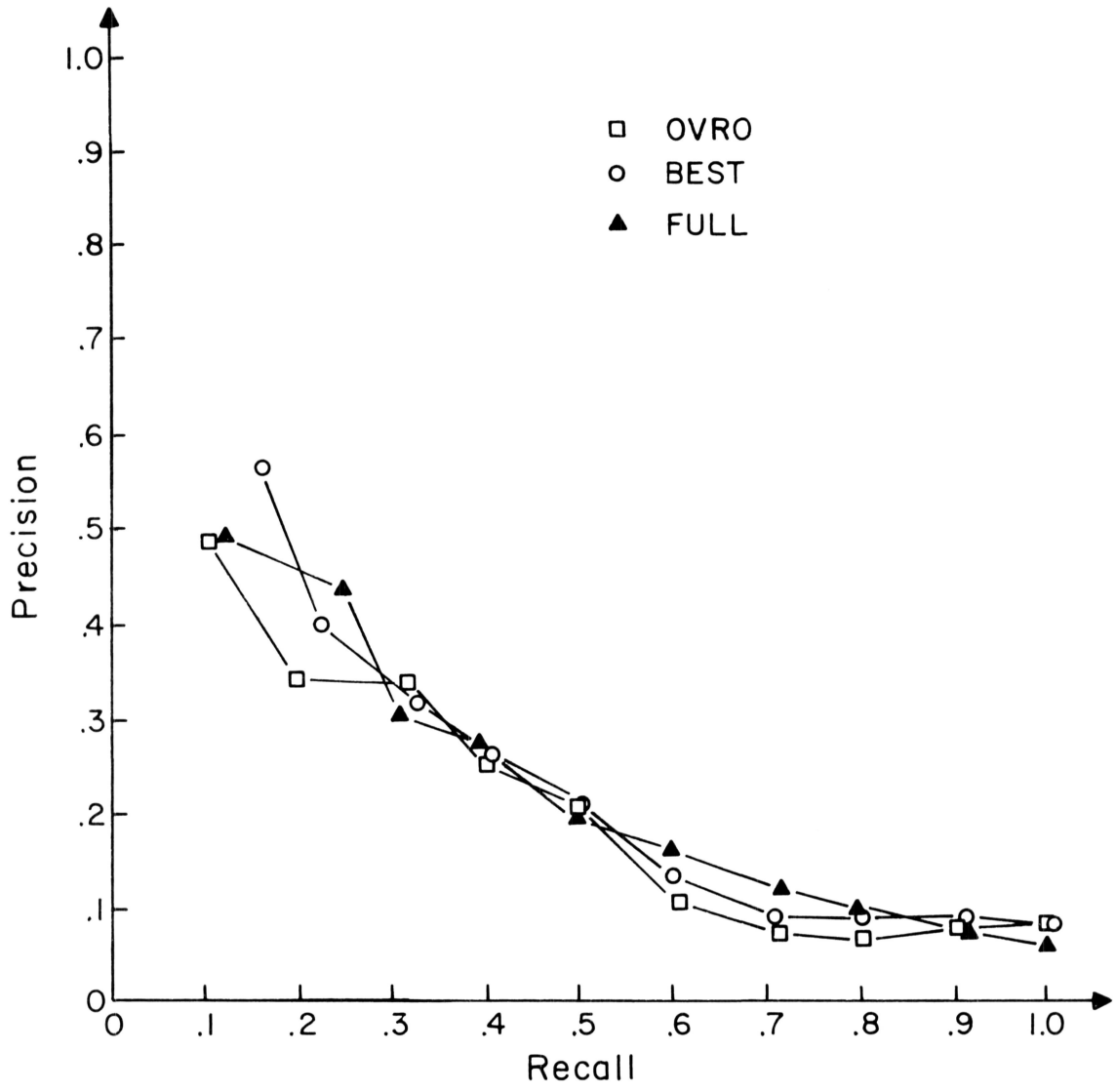
H) External Evaluation

On the basis of the internal evaluation results, the following conclusions are made regarding the "best" input parameters for the classification algorithm:

1) initial clusters — no conclusion (correlation used);
2) number of clusters — approximately $\sqrt{N}$ clusters; e.g., the best results are obtained with 9 clusters for the ADI collection and 13 for the Cranfield collection;
3) overlap — probably somewhere between 1-3% (2% used);
4) cutoff — 10%;
5) percent loose clustered — 60%.

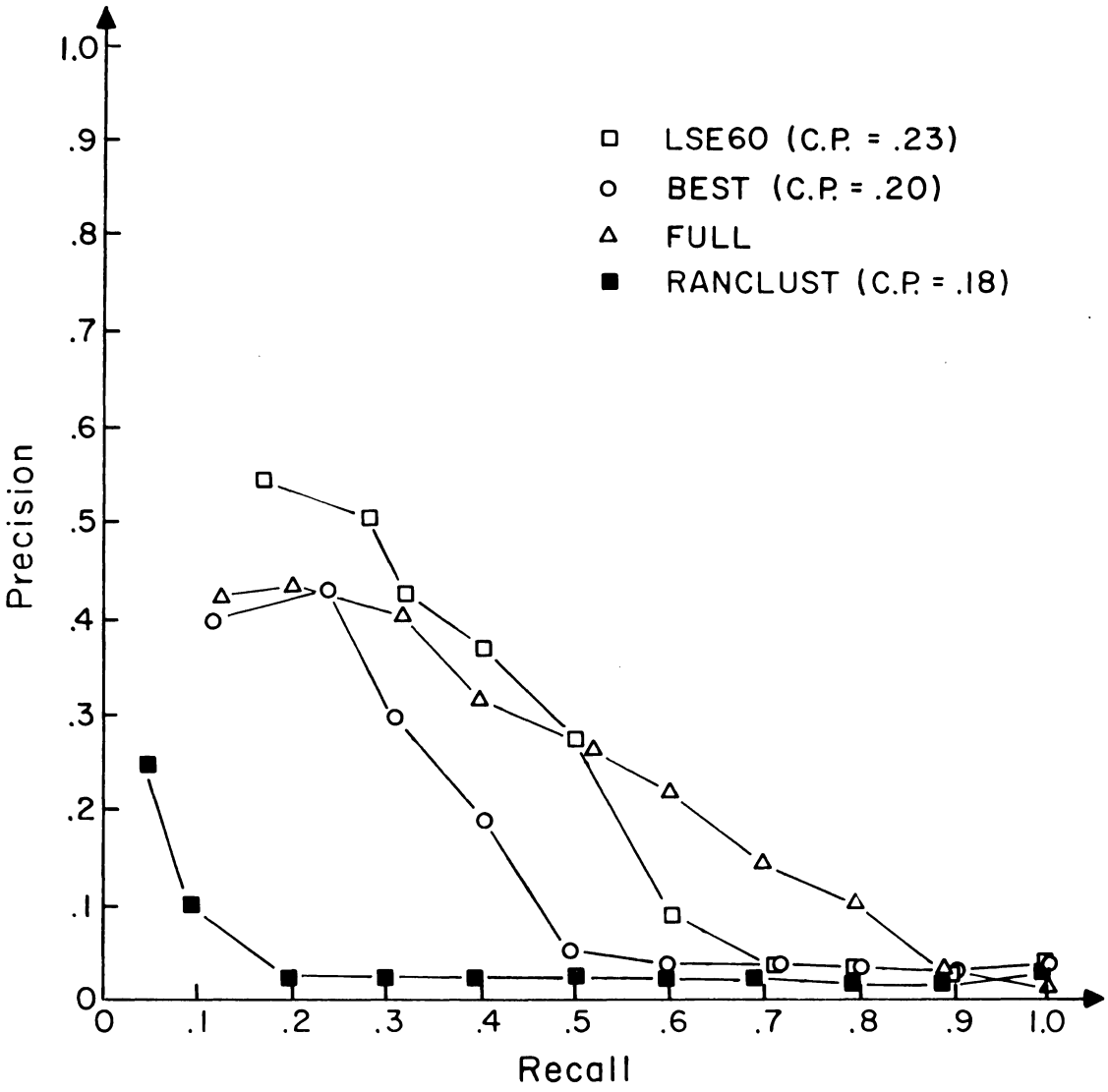
An additional run called BEST is made using these parameters as input. Unfortunately, all of the output parameters did not satisfy their input requests; e.g., the overlap is 0% for the ADI and .5% for the Cranfield. However, LSE60 from the Cranfield clusters matches the parameters almost exactly — 13 clusters, 1.5% overlap.

Fig. 22 shows the recall-precision curves for the ADI BEST, OVRO, and FULL. Fig. 23 shows BEST, LSE60, and FULL for the Cranfield collection. Also included in Fig. 23 are the results of a random clustering of the collection. Both ADI BEST and OVRO have 0% overlap, and the results show that FULL > BEST > OVRO . In the Cranfield results, the surprising fact is the poor showing of BEST. However, the run which actually contains the desired output parameters, LSE60, performs better than the full search up to 50% recall. Thus, (FULL = LSE60) >> BEST >> RANCLUST .



ADI Full Search Comparison

Fig. 22



Cranfield Full Search Comparison

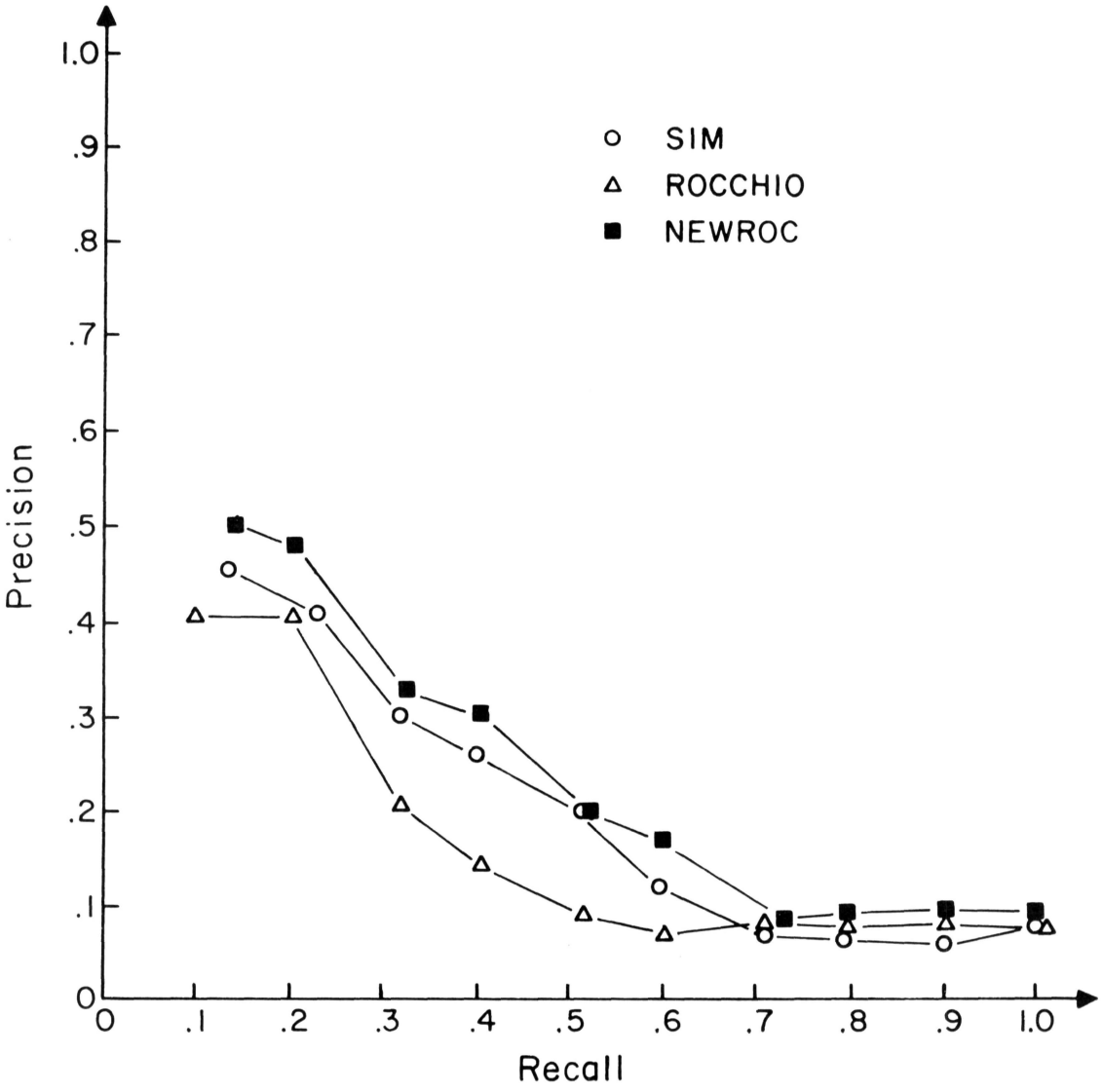
Fig. 23

Comparisons can also be made with cluster results from Rocchio's classification algorithm. [6] Several sets of clusters are available as output from Rocchio's algorithm, and those which yield the best retrieval results are chosen for comparison. These clusters are called ROCCHIO in Table 1 and Table 2. The Rocchio clusters are compared against those runs which most closely match the number of clusters and overlap of ROCCHIO. Thus, SIM is used for the ADI results, and RANDOM is used from the Cranfield results. The search parameters are adjusted so that exactly two centroids are chosen for every query. Fig. 24 and Fig. 25 show that SIM >> ROCCHIO and RANDOM2 >> ROCCHIO . However, another run called NEWROC is made by using Rocchio's clusters with the same centroid definition that is used for SIM and RANDOM2 .

The centroids for Rocchio's clusters are defined by using all the concepts and taking the sum of the weights; i.e., if concept i occurs in two documents within the cluster with weights of x_1 and x_2 , then its weight in the centroid is $x_1 + x_2$. The centroids for the clusters in this study are defined as the top y percent of the profile vector; i.e., the weights of the concepts are equal to their rank values. For the ADI experiments a minimum of 50% of the concepts are used. The rank value differs from Rocchio's weights in two important respects:

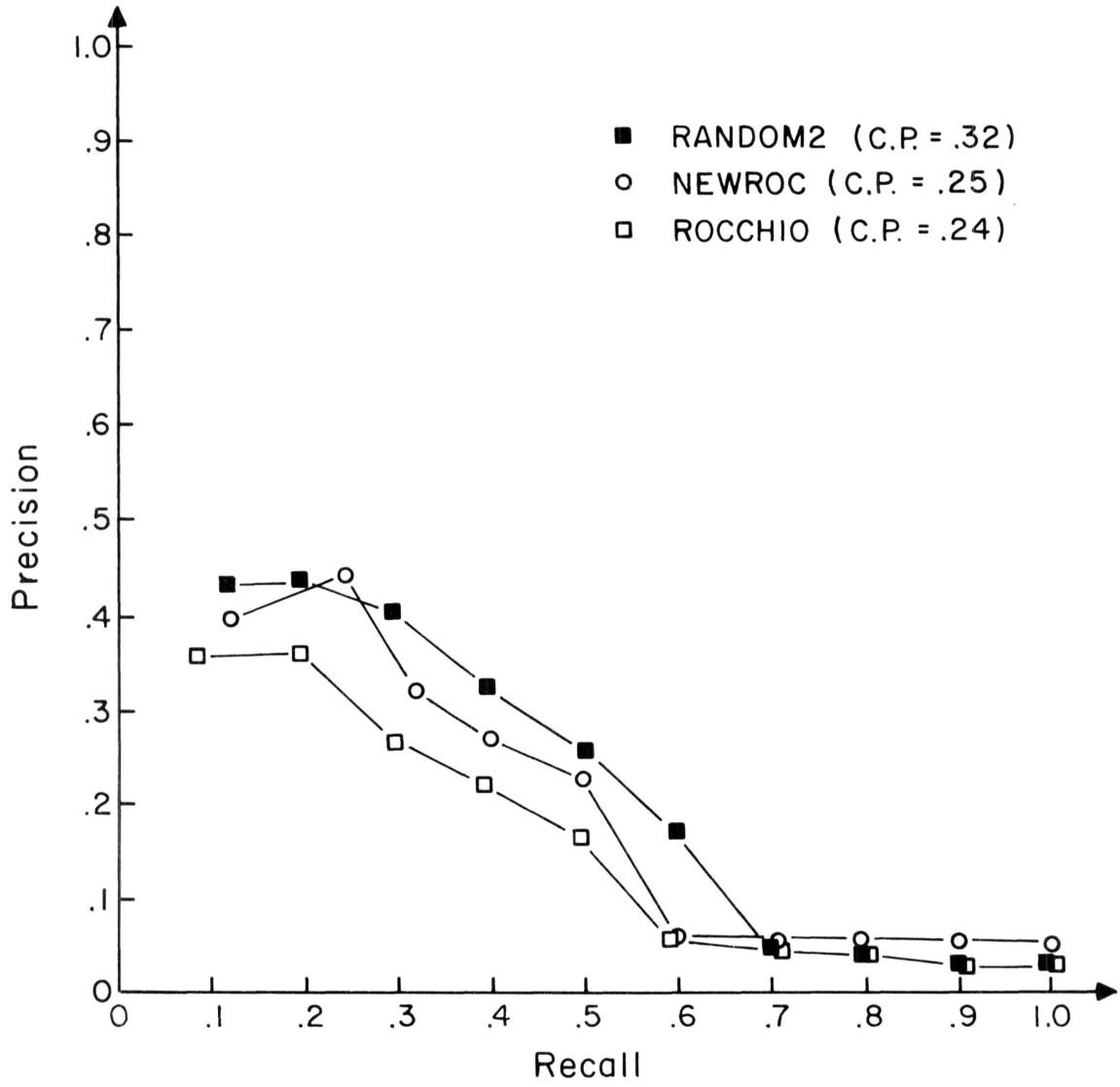
1) the weight of a concept within a particular document is ignored in computing the rank value; i.e., all concepts in a document are assigned the same weight;

2) the magnitude of the difference in the number of documents within the cluster in which a concept occurs is ignored; i.e., if concept i is ranked first and occurs in 10 documents, and concept j is ranked second, then it doesn't matter if



ADI Rocchio Comparison

Fig. 24



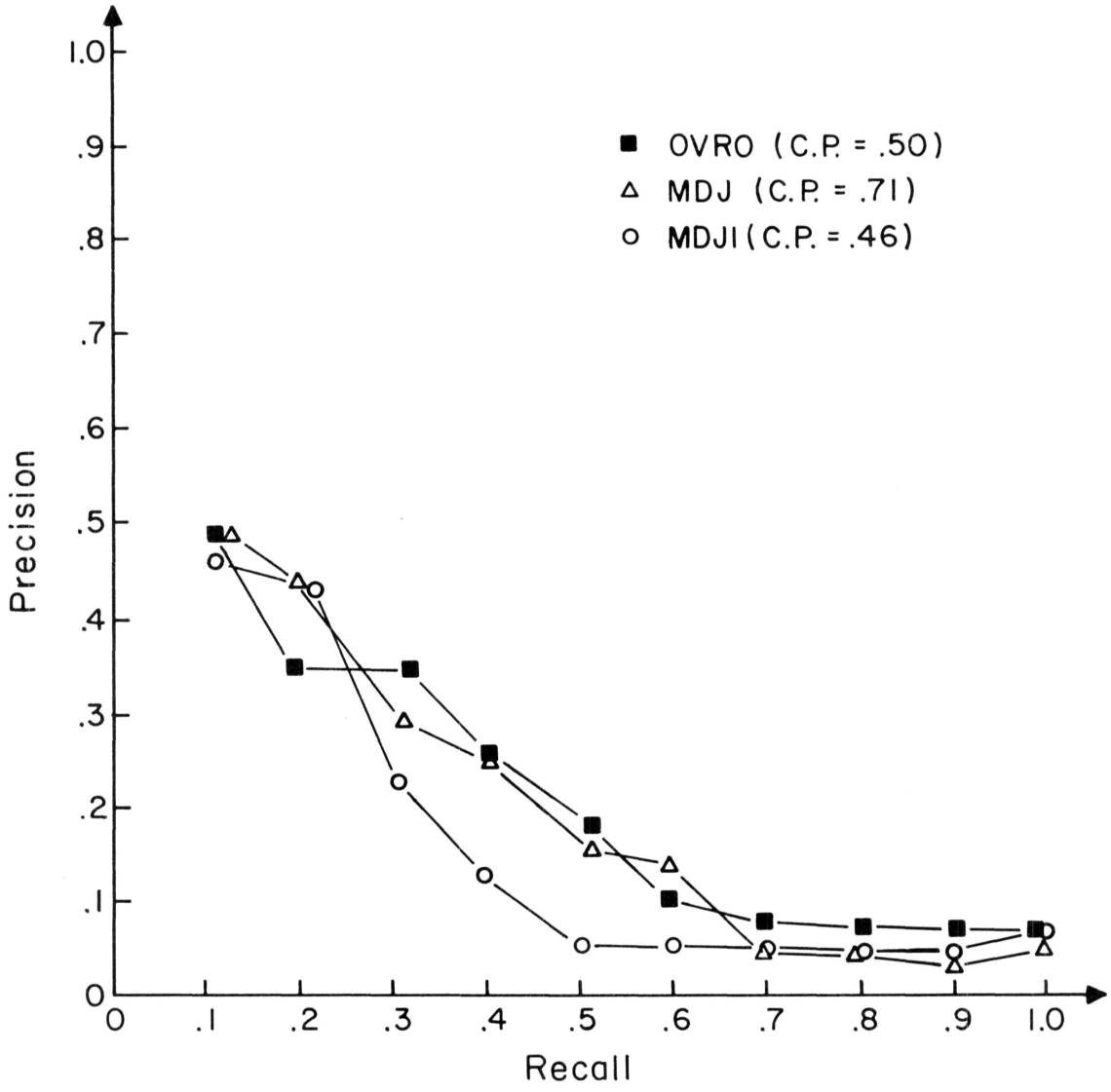
Cranfield Rocchio Comparison

Fig. 25

concept j occurs in 9 documents or only 2 documents, its rank value is still one less than the rank value of concept i .

Fig. 24 shows that NEWROC > SIM >> ROCCHIO , while Fig. 25 shows RANDOM2 > NEWROC >> ROCCHIO . Thus, in both cases the results improve greatly over the Rocchio clusters simply by changing the centroid definition. However, the present classification algorithm still performs better than NEWROC for the Cranfield results, although NEWROC does better for the ADI results.

The final external evalution is made by comparison with the results from a one-pass clustering algorithm. A description of the algorithm and the results are presented elsewhere; [5] unfortunately the results are misleading. The best set of one-pass clusters is chosen (MDJ - 0% overlap) and plotted against OVRO using the new evaluation measure. Fig. 26 shows that OVRO = MDJ . However, C.P. = 0.50 for OVRO and C.P. = 0.71 for MDJ. Both searches are made using a minimum of two clusters per query. The reason for the very high correlation percentage for the one-pass clusters is due to the size of the clusters. MDJ contains 14 clusters, but the three largest clusters, which are usually chosen in the search, contain 72% of the documents; seven of the clusters contain only one document. Another run is made called MDJ1 where exactly one cluster is chosen per query. Even in this case C.P. is quite high — =.46, but now OVRO > MDJ1. It is clear that the one-pass algorithm needs to be modified so that the size of the clusters does not vary so much. Perhaps an additional pass should be made to break up large clusters and to merge smaller clusters.



ADI Fast Cluster Comparison

5. Conclusion

The multi-level classification algorithm presented in this study runs in time $T = k \cdot N \cdot p \cdot \log_p m$, where k is a constant, N is the number of documents in the collection, m is the final number of clusters desired, and p is the number of clusters produced at each level of the algorithm. It is shown that the closer p is to e, the faster will the algorithm operate. The complete multi-level algorithm is not yet implemented, so that all the experiments are run with $p = m \Rightarrow T = k \cdot N \cdot m$. With $m = \sqrt{N}$, $T = k \cdot N^{3/2}$. This is, of course, much better than classification methods that run in time proportional to N^2, but it is still not satisfactory for very large collections. For these collections it is necessary to implement the entire algorithm, and to run with small values of p. With $N = 10^6$, it is theoretically possible for T to equal $k \cdot 10^6 \cdot 3 \cdot \log_3 m$, where $p = 3$. If once again $m = \sqrt{N}$, then $T = k \cdot 10^6 \cdot 3\log_3 10^3 = k \cdot 10^6 \cdot 3 \cdot 6.3 = k \cdot 10^6 \cdot 18.9 = 2k \cdot 10^7$. This is much better than using only one level, where $p = m$ and $T = k \cdot 10^6 \cdot 10^3 = k \cdot 10^9$.

Fortunately, many of the input parameters which yield the best search results also help to lower the constant of proportionality k. Table 2 shows that LSE60, the best cluster run, took approximately 2.3 minutes for the 200 document Cranfield collection, while LSE20 took over 9 minutes! The high percent of loose clustered, and the low amount of overlap both reduce the clustering time.

One of the major problems with the algorithm is the failure to satisfy the requested amount of overlap. This is due to the fact that the parameter a is not changed enough after each cycle. Recall that a is reset to $a + (1-a) \cdot (XOVER - \phi)$ if $\phi < XOVER$, and reset to $a + a \cdot (XOVER - \phi)$ if $\phi > XOVER$.

One method of improving this is to introduce a parameter b such that,

$$a = \begin{cases} a + b\cdot(1-a)\cdot(XOVER-\phi) & \text{if } \phi < XOVER \\ a + b\cdot a\cdot(XOVER-\phi) & \text{if } \phi > XOVER \end{cases}$$

The parameter b is set to 1.0 at the beginning of each iteration. After every cycle, b is reset as follows:

$$b = b\cdot[1 + |XOVER-\phi|]$$

Thus, b ranges between 1.0 and 2.0.

A comparison with Rocchio's algorithm shows that the method used to define centroids is very important. Certainly the rank values prove to be better than the sum of the weights. However, it is not clear whether the rank values are better because they ignore weights within documents, or because they ignore the magnitude of differences in the number of documents in which each concept occurs. This can be decided by summing the weights as Rocchio does, but instead of using this number as the weight, by ranking the concepts according to their sum and then calculating the rank values to be used in the centroid definition.

All the evaluations performed in this study are done by visual inspection of the document level recall-precision graphs. This rather inexact method can be improved by using statistical tests such as the sign test and the t-test to compare two curves. Routines are being programmed to perform such tests, and they will be used in the future.

Finally, all of the conclusions and evaluations are based on results from an 82 document and a 200 document collection, containing 35 and 42 queries respectively. These results should be supplemented by experiments

run on the 424 document Cranfield collection containing 155 queries, and eventually on the 1400 document Cranfield collection.

References

[1] R. T. Dattola, A Fast Algorithm for Automatic Classification, Scientific Report No. ISR-14 to the National Science Foundation, Section V, Department of Computer Science, Cornell University, October 1968.

[2] L. B. Doyle, Breaking the Cost Barrier in Automatic Classification, SDC paper SP-2516, July 1966.

[3] S. L. Worona, Query Clustering in a Large Document Collection, Scientific Report No. ISR-16 to the National Science Foundation, Section XV, Department of Computer Science, Cornell University, September 1969.

[4] K. S. Jones, D. Jackson, The Use of Automatically-obtained Keyword Classifications for Information Retrieval, Final Report, Cambridge Research Unit, Cambridge, England, February 1969.

[5] V. P. Marathe, S. L. Rieber, A One-Pass Clustering Algorithm, Scientific Report No. ISR-16 to the National Science Foundation, Section XIV, Department of Computer Science, Cornell University, September 1969.

[6] J. J. Rocchio, Document Retrieval Systems — Optimization and Evaluation, Scientific Report No. ISR-10 to the National Science Foundation, Harvard University, March 1966.