Information Technology: Research and Development (1982), *1* (275–283)

A VERY FAST, EXACT NEAREST NEIGHBOUR ALGORITHM FOR USE IN INFORMATION RETRIEVAL

F. MURTAGH

Department of Computer Science, University College Dublin, Dublin 4, Ireland

(*Received 26 April 1982, revised 30 June 1982*)

ABSTRACT

The finding of the document(s) closest to a query, or the finding of document(s) closest to a given document, is a central problem in information retrieval. It is often practical to presuppose that both the document/index term file and the inverted term/document file are available. In this context, a recently proposed algorithm used an upper bound on closeness, thus obviating the need for an exact calculation of closeness in many instances. An improved version of this algorithm is presented here. Within the framework of this approach, we also look at how certain similarity or dissimilarity coefficients might be preferred to others. Finally, we specify the condition under which the use of both document/term and term/document files will lead to more efficient search algorithms.

1. BACKGROUND

The nearest neighbour (NN) problem—the problem of locating the closest point to a given point, using some definition of 'closeness'—is a central problem in many areas. In information retrieval, the particular features of document/index term data have given rise to a number of suggested approaches, referenced in later sections, for efficiently calculating NNs of given documents or queries. One recent approach (Smeaton and van Rijsbergen, 1981) determined a least possible dissimilarity (greatest possible similarity) using information that was storable in central memory. If such a lower bound on the dissimilarity (upper bound on the similarity) showed sufficient promise, the full calculation of the (dis)similarity was proceeded with, and then tested against the previous best NN. Smeaton and van Rijsbergen went further and used the bound to eliminate some of the longer inverted file lists *in toto*. The bounding operation they described can, however, be substantially tightened, and this general approach to the NN problem can further suggest which (dis)similarity function should be used for best results.

2. INTRODUCTION

A *document collection* consists of n documents, each characterized by one or more of m index terms. A *document* may thus be represented by a vertex of a hypercube in m-dimensional space. Define such a point by the binary vector $\{i(j) : j = 1, 2, \ldots, m\}$. A *term* is similarly a vertex of the unit hypercube in n-dimensional space. A term will be denoted by the binary vector $\{j(i) : i = 1, 2, \ldots, n\}$. A *query* is mathematically defined in an identical manner to a document, and we will consider here only the problem of determining the NN of a document. Other similar problems such as the all-points NN problem or the k-NN problem are immediate generalizations.

Define the set of *possible NNs* of document i, POSS(NN(i)), as the set of documents i' which share at least one term in common with i:

$$\text{POSS}(\text{NN}(i)) = \{i' : (\exists\, j)\,(i(j) = i'(j) = 1)\}$$

Croft (1977) first proposed that the set POSS could be calculated easily using an inverted file of terms/documents as well as the initial document/term file (i.e., both the set of document vectors and the set of term vectors, stored in some efficient manner). Since his implementation took POSS as a multiset—some documents were included many times over—Willett (1981) has proposed a modified way of determining POSS to correct this. Harding and Willett (1980) have also discussed when the determining of POSS would tend to be disadvantageous, i.e., when the number of terms associated with documents becomes too great. However, a more easily applied rule can be formulated: this will be given in Section 5.

The set of *feasible NNs* of document i, FEAS(NN(i)), is defined as the set of documents i' whose positions are, at best, within a certain radius of document i:

$$\text{FEAS}(\text{NN}(i)) \subset \text{POSS}(\text{NN}(i))$$
$$\text{FEAS}(\text{NN}(i)) = \{i' : \text{LB}(d(i, i')) < r\}$$

where LB is the lower bound on the dissimilarity between document i and candidate NN document i'; d is a dissimilarity, i.e., a real-valued function with properties: $d(i, i') = d(i', i)$ and $d(i, i) \geqslant 0$ (the more common use of similarities may be easily catered for by reversing the inequality and replacing LB with UB, an upper bound); r is a radius, equal to the dissimilarity between document i and the current NN (initially ∞; or 0 in the case of similarities).

A feasible NN of document i is thus a document whose least possible dissimilarity with i is less than the current NN dissimilarity. Consequently there is a good chance that $d(i, i')$ is in fact less than the current NN dissimilarity.

The advantage of specifying whether or not a document belongs to the set FEAS at any stage lies in the fact that some information regarding the documents can be made easily accessible. If set FEAS is small, then very few full (dis)similarity calculations will be necessary. If a good current NN is selected at an early stage, membership in FEAS is made more restrictive, thus enhancing the performance of the algorithm.

All the (dis)similarity measures discussed in the next section are functions of some or all of the following:

CARD(i): the number of terms associated with the given document i.
CARD(i'): the number of terms associated with a candidate NN.
COMM(i, i'): the number of terms common to i and i'.

It is assumed that the CARD values of all n documents can be kept in memory throughout. The data required to calculate COMM(i, i') on the other hand must normally be held on some direct-access device, due to the size of the document collection.

The lower bounding of the dissimilarity (upper bounding of the similarity) is achieved by upper bounding COMM. Let us consider the case where the NN of document i is sought, and where at some stage in the execution, document i' is being examined. This situation arises via the following steps:

1. Access the terms associated with document i.
2. For each term in turn, consider the set of documents which are indexed by the term, and which have not been examined before. (The inverted file is used here, and the set POSS is thereby built up in stages.)
3. Consider the kth such term, and let document i' be indexed by this term. Document i' cannot have been met with before. Therefore at most CARD(i)$-k+1$ terms are common to document i and document i' (i.e., the most favourable supposition is made that *all* remaining terms in document i are common to document i'). On the other hand, from the point of view of document i', CARD(i') is the maximum number of terms which can be common to i and i'.

Consequently we arrive at the lower bound estimate for COMM(i, i') as Min (CARD(i'), CARD(i)$-k+1$). This is used to give an estimated least value of $d(i, i')$. If it is less than the current NN dissimilarity, the terms associated with i' are accessed and an exact calculation of $d(i, i')$ is carried out.

In the experimentation, it was found that the bound on COMM(i, i') could be simplified to CARD(i)$-k+1$ with very few extra calculations, and in some cases with small improvements in CPU time.

3. EXPERIMENTAL RESULTS

The well-known National Physical Laboratory's (NPL) test collection was used. The NNs of 93 test queries were determined. The relevant statistics of the collection and of queries are:

Number of documents:	11429
Number of terms:	7491
Average terms/document:	19.9
Average documents/term:	30.4
Average terms/query:	7.1

Using the notation of the previous section, the (dis)similarity functions used were:

Hamming:	CARD(i) + CARD(i') $-$ 2.COMM(i, i').
Simple:	COMM(i, i').
Ivie:	COMM(i, i')/(CARD(i).CARD(i')).

Dice: $2.\mathrm{COMM}(i, i')/(\mathrm{CARD}(i) + \mathrm{CARD}(i'))$.
Cosine: $\mathrm{COMM}(i, i')^{**}2/(\mathrm{CARD}(i).\mathrm{CARD}(i'))$.
Jaccard: $\mathrm{COMM}(i, i')/(\mathrm{CARD}(i) + \mathrm{CARD}(i') - \mathrm{COMM}(i, i'))$.
Overlap: $\mathrm{COMM}(i, i')/\mathrm{Min}(\mathrm{CARD}(i), \mathrm{CARD}(i'))$.

where the first function is a distance, and all others are similarities. The Hamming distance is also known as the 'city-block', least-moves, symmetric difference, or Minkowski 1-distance. It is also the squared Euclidean or Minkowski 2-distance in the case of binary data.

However unlike the other (similarity) measures above, a minimum Hamming distance between two documents could have a zero value for COMM, i.e., it is not always necessary for an index term to be common to both. In other words the NN document might not be a member of set POSS, and *a fortiori* of set FEAS. It may be easily established that the NN, using the Hamming distance, will be *either* the NN which shares at least one term, *or* the document of least CARDinality which has no term in common (and hence $\notin$ POSS). This can be checked for, at little extra computational cost. It might on the other hand be deemed desirable that a NN document have at least one term in common with a given document or query. This was done in the programmed version of the algorithm, and by a slight abuse of terminology we will refer to this as the Hamming distance in what follows.

The results obtained using 93 test queries and the various (dis)similarity functions defined above are shown in Table 1. The first column gives the results obtained on the same data by Smeaton and van Rijsbergen (1981), and the second column gives the results obtained by the algorithm under discussion in this article. Both sets of results refer to the average numbers of full calculations of (dis)similarities per query. Column 3 gives the CPU time required, for determining the NNs of all 93 queries, on a DEC-2060 machine; these are intended as being indicative only.

Table 1. Results obtained

	*Number of determinations of terms common to query and documents**		
	Smeaton and van Rijsbergen (1981)	*Present method*	*CPU times* (min:s)†
Hamming	—	101	4:40
Simple	—	307	8:03
Ivie	1755	148	5:32
Dice	1591	307	7:43
Cosine	1876	349	8:51
Jaccard	—	307	7:51
Overlap	—	312	8:23

* Average per query.
† For all 93 NN searches.

In all cases, an average of 4078 documents would have been obtained by using the documents indexed by a given query's terms; of these, 3162 on average were distinct, i.e., POSS contained an average of 3162 documents. In the case of the Hamming distance, we find on average that:

1. POSS reduced the total number of documents to be examined, i.e., 11429, to 3162.
2. FEAS further reduced this to 101.

Overall, then, less than 1 per cent of the documents had to be fully examined when searching for the NN of the average query.

The NNs of the first 500 documents in the NPL collection were also obtained. The average number of terms per document was almost three times the average number of terms per query (cf., statistics at the beginning of this section), and consequently many more documents had to be examined. POSS in this case contained an average of 4807 documents, and FEAS further reduced this to 767 documents. Thus the full Hamming distance had to be calculated on average with less than 7 per cent of the possible 11429 documents in order to determine a NN.

4. THE SUITABILITY OF DISSIMILARITIES FOR USE IN PROPOSED ALGORITHM

The results obtained differ for the various (dis)similarity functions used. The question arises as to whether or not any particular (dis)similarity function can be recommended when using this bounding strategy. We are therefore led to propose a suitability axiom for (dis)similarity functions used in the bounding strategy. As will be seen, this axiom is verified only by the Hamming distance.

With reference to Figure 1, we have a given document i with CARD(i) associated terms; we have a current NN document i', of CARD(i') terms; and we have, finally,

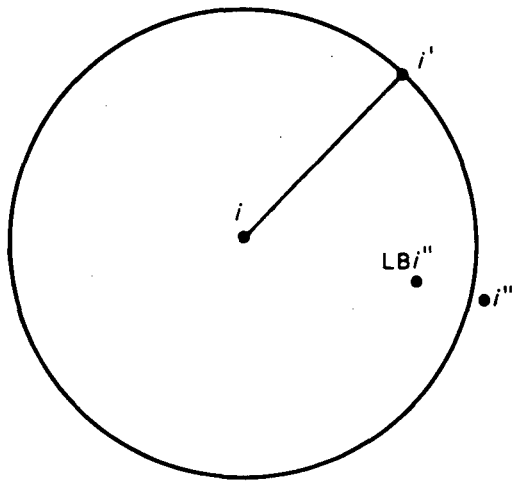

FIG. 1. Schematic view at any stage of finding of NN. i = Document or query for which NN is desired; i' = current NN document; i'' = candidate NN document; LBi'' = vector such that: $d(i, \text{LB}i'') = \text{LB}(d(i, i''))$

a candidate NN document i'', of CARD(i'') terms. It is assumed that we have immediate access to the value of CARD(i''). In order to decide whether or not to determine the number of terms common to i and i'', i.e., COMM(i, i''), we compare the current NN dissimilarity, $d(i, i')$, with the lower bounded dissimilarity between i

and i'': $\mathrm{LB}(d(i, i'')) = d(i, \mathrm{LB}(i''))$. (What we have done here is to interpret the bounded distance, $\mathrm{LB}(d(i, i''))$, as a distance between i and some other vector, called $\mathrm{LB}(i'')$). We will abbreviate $\mathrm{LB}(i'')$ to $\mathrm{LB}i''$. It may be noted that more than one vector, $\mathrm{LB}i''$, will ordinarily satisfy

$$\mathrm{LB}(d(i, i'')) = d(i, \mathrm{LB}i'')$$

In what follows, where we will be discussing $d(i'', \mathrm{LB}i'')$, we will assume that $\mathrm{LB}i''$ is always chosen in such a way that this dissimilarity is minimum (an arbitrary $\mathrm{LB}i''$ is chosen if there is more than one that satisfies this condition).

The axiom of suitability is now defined as:

$$d(i'', \mathrm{LB}i'') + d(i, \mathrm{LB}i'') = d(i, i'') \qquad \text{(AS)}$$

i.e., if the vector $\mathrm{LB}i''$ is close to candidate NN vector i'', then the lower bound estimate of $d(i, i'')$ is close to the real value of $d(i, i'')$, and vice versa. In the context of a metric space, equality (AS) may be interpreted as the fact that vector $\mathrm{LB}i''$ lies on the shortest path connecting vectors i and i''. The axiom of suitability is therefore proposed as a basic condition of 'well-definedness' when implementing a bounding strategy. Two lemmas are now proved, from which the advantage of the Hamming distance immediately follows.

Lemma 1: In the case of the Hamming distance, equality (AS) is satisfied.

Proof: As a preliminary to the proof, we note that vector $\mathrm{LB}i''$ has the same number of ones as does i'', and differs only in that the location of these ones has changed.

In the case of the Hamming distance we have:

$$\begin{aligned} d(i, \mathrm{LB}i'') &= \mathrm{CARD}(i) + \mathrm{CARD}(i'') - 2.\mathrm{COMM}(i, \mathrm{LB}i'') \\ d(i, i'') &= \mathrm{CARD}(i) + \mathrm{CARD}(i'') - 2.\mathrm{COMM}(i, i'') \\ d(i'', \mathrm{LB}i'') &= \mathrm{CARD}(i'') + \mathrm{CARD}(i'') - 2.\mathrm{COMM}(i'', \mathrm{LB}i'') \end{aligned}$$

We claim that:

$$\mathrm{COMM}(i'', \mathrm{LB}i'') = \mathrm{CARD}(i'') - \mathrm{COMM}(i, \mathrm{LB}i'') + \mathrm{COMM}(i, i'')$$

If that is true, it follows that

$$d(i, i'') = d(i, \mathrm{LB}i'') + d(\mathrm{LB}i'', i'')$$

The claim that

$$\mathrm{COMM}(i'', \mathrm{LB}i'') = \mathrm{CARD}(i'') - \mathrm{COMM}(i, \mathrm{LB}i'') + \mathrm{COMM}(i, i'')$$

can be proved by distinguishing two cases.

$$\textit{Case I}: \quad \mathrm{COMM}(i, \mathrm{LB}i'') = \mathrm{CARD}(i'')$$

In this case, *all* the one values of vector $\mathrm{LB}i''$ overlap with the one values of vector i. Only $\mathrm{COMM}(i, i'')$ ones of vector i'' overlap with the ones of vector i (and $\mathrm{COMM}(i, i'') \leqslant \mathrm{COMM}(i, \mathrm{LB}i'')$ always). Therefore $\mathrm{COMM}(i, i'')$ ones have not

been displaced in the creation of LBi'' from i'', and so COMM(i'', LBi'') = COMM(i, i'').

Case II: $\text{COMM}(i, \text{LB}i'') < \text{CARD}(i'')$

In this case, there are $\text{CARD}(i'') - \text{COMM}(i, \text{LB}i'')$ ones in vector i'' which do not overlap with one values in vector i. In other words, these one values have not been displaced in creating LBi'' from i'', and they therefore must contribute to the term COMM(i'', LBi''). In addition there are COMM(i, i'') ones among the COMM(i, LBi'') ones which have also had no need to be relocated.

This completes the proof of the claim, and of Lemma 1.

Lemma 2: In the case of all other (dis)similarity functions listed in Section 2, equality (AS) is not satisfied.

Proof: Taking as an example the simple matching coefficient, and converting it to a dissimilarity we get:

$$d(i, \text{LB}i'') = M - \text{COMM}(i, \text{LB}i'')$$
$$d(i, i'') = M - \text{COMM}(i, i'')$$
$$d(i'', \text{LB}i'') = M - \text{COMM}(i'', \text{LB}i'')$$

where M is a sufficiently large constant. Substituting for COMM(i'', LBi'') as found in Lemma 1, we find that the equality

$$d(i, i'') = d(i, \text{LB}i'') + d(\text{LB}i'', i'')$$

is false.

Lemma 2 may be proved for other coefficients in a similar manner.

Theorem: Of all (dis)similarities listed in Section 2, the Hamming distance alone satisfies the axiom of suitability, i.e.,

$$d(i'', \text{LB}i'') = d(i, i'') - \text{LB}(d(i, i''))$$

Proof: Lemma 1 shows that this condition is realized only for the Hamming distance, and Lemma 2 shows that it is not valid in any other metric.

5. INDEXING EXHAUSTIVITY AND THE USE OF THE INVERTED FILE

Document/index term data used in information retrieval are characterized by extreme sparsity, i.e., the number of document/term associations would normally be no more than a few per cent. Could the algorithm proposed here work for other, less sparse, binary (zero-one) data?

Croft (1977), who first proposed using the inverted term/document file in order to take full advantage of the sparsity of the data, did not specify the condition under which the procedure was viable. Harding and Willett (1980) addressed themselves to the problem of repeatedly coming across the same document as index terms associated with the initially-given document were processed. Their analysis was in the context of automatic classification, requiring an $n \times n$ matrix. The problem examined here is the condition under which the set POSS will substantially reduce the original set of n documents. (In our programmed implementation of the

bounding algorithm, a Boolean vector indicated when a POSSible NN document was processed, and thus allowed subsequent checking for re-occurrences of this document.) In the following simple analysis, we will revert temporarily to taking POSS as a multiset (cf., Section 2).

Let the average number of documents associated with an index term be denoted by *nn*, and the average number of terms per document by *mm*. Thus, since there are *n* documents and *m* index terms in the collection, the number of document/term dependencies is $n \times mm$ or $m \times nn$. Consider now the processing of an index term associated with the initially-given document. We want all documents associated in turn with this term. On average, *mm* such sets of documents will need to be examined, and there will be *nn* documents in each such set. Consequently, ignoring the repetition of occurrences of documents, POSS will consist on average of $nn \times mm$ documents. If then

$$n > nn \times mm$$

it follows that searching through $nn \times mm$ documents will be more efficient than searching through *n* documents. If this relation is satisfied, i.e., if the number of documents in the collection is greater than the average number of documents per term, multiplied by the average number of terms per document, then the strategy which uses the inverted file can definitely be recommended. In fact, some of the $nn \times mm$ documents will not need to be examined as they will have already been met with. Note, finally, that in the case of searching for the NNs of queries, *nn* and *mm* will be the average number of documents per (query) term and the average number of terms per query, respectively, and these will generally have different values from the case in which the NNs of documents are sought.

6. CONCLUSION

We have described in this article a general approach for determining NNs. A bounding strategy has been used to estimate in advance what more costly calculations are required. Results obtained are considerably better than anything reported on, so far, in the information retrieval area. We have also proved that a particular 'closeness' measure is most suited to this general strategy. Finally, and very importantly for practical applications, a specific recommendation was made as to when the proposed approach would be effective.

In a more general perspective on the NN problem—outside the information retrieval area—it is unlikely that the bounding strategy described could be used. This is due to the result obtained in Section 5.

The approach to the NN problem described here is a very efficient $O(n)$ algorithm. A divide-and-conquer approach would lessen the $O(n)$ dependency of a NN search (see, for example, Weiss, 1981, who describes initial work in such a framework). However it is not at all clear that such an approach would prove more practical than the powerful, easily implementable, exact search algorithm presented.

ACKNOWLEDGEMENT

This article would not have been possible without many discussions with A. F. Smeaton. I am indebted to him for the use of his programs. My thanks also to P. Willett for a number of very helpful comments.

REFERENCES

Croft, W. B. (1977) Clustering large files of documents using the single-link method. *Journal of the American Society for Information Science 28*, 341–344.

Harding, A. F. and Willett, P. (1980) Indexing exhaustivity and the computation of similarity matrices. *Journal of the American Society for Information Science 31*, 298–300.

Smeaton, A. F. and van Rijsbergen, C. J. (1981) The nearest neighbour problem in information retrieval: an algorithm using upperbounds. *Proceedings of the Fourth International Conference on Information Storage and Retrieval*, Oakland, California, 31 May–2 June, pp. 83–87.

Weiss, S. F. (1981) A probabilistic algorithm for nearest neighbour searching. In *Information Retrieval Research.* (R. N. Oddy, S. E. Robertson and C. J. van Rijsbergen, eds.) pp. 325–333. London: Butterworths.

Willett, P. (1981) A fast procedure for the calculation of similarity coefficients in automatic classification. *Information Processing and Management 17*, 53–60.