# USING THE COSINE MEASURE IN A NEURAL NETWORK FOR DOCUMENT RETRIEVAL

Ross Wilkinson
Philip Hingston

Department of Computer Science
Royal Melbourne Institute of Technology
GPO Box 2476V
Melbourne, VIC 3001
Australia

July 11, 1991

**Abstract**

The task of document retrieval systems is to match one natural language query against a large number of natural language documents. Neural networks are known to be good pattern matchers. This paper reports our investigations in implementing a document retrieval system based on a neural network model. It shows that many of the standard strategies of information retrieval are applicable in a neural network model.



## 1 Introduction

A key requirement of large organizations is to manage large volumes of natural language text. Typically, one might wish to locate relevant documents from a collection of one million documents. To manually index, and then retrieve the documents may well be economically infeasible. Thus there is considerable interest in automatically indexing and retrieving documents from such collections. By describing each document by a set of terms derived from the document, we can index the document, and we are then confronted with the task of comparing a query with these sets of terms.

Neural networks can perform very well at matching a given pattern against a large number of possible templates. We use this organisation for selecting relevant documents.

Since it is not possible to compare a given query with one million raw text documents in an acceptable time, documents are indexed prior to query time and then the query is transformed and matched against the indexed terms. The standard way of indexing documents is to select key words or all significant words in a document.

These techniques have a well developed history and literature [Salton89]. The effectiveness of a text retrieval system can be measured in terms of recall and precision. To obtain these measures, an expert examines all documents in the database against a given query and classifies documents in the database as relevant or not. The retrieval system performs the same task.

$$Recall = \frac{(No.\ relevant\ documents\ retrieved)}{(No.\ relevant\ documents)}$$

$$Precision = \frac{(No.\ relevant\ documents\ retrieved)}{(No.\ retrieved\ documents)}$$

Other important measurements include the time taken to index documents and the speed with which documents are matched against the query and retrieved.

In section two we briefly describe the vector space model. In section three we discuss measuring the effectiveness of relevance feedback. In section four we describe neural nets and our model. In the next section we explain how we use the model, and follow it with a comparison with previous work. In section seven we give our results. Finally we describe our plans and conclusions.

## 2 The Vector Space Model

Documents are usually described by a set of terms. A common automatic indexing strategy is to take the set of all words found in the document, remove the most common words such as "the" and the uninteresting terms like "thing", and stem the remaining terms to get "tim" from "timing" and "times". The remaining items constitute the set of terms. This list might be extended by using a thesaurus, or by generating pairs of words that are either adjacent or syntactically related. The thesaurus would widen the possible matches for a document, and the pairs would allow for more refined matching.

The vector space model creates a space in which both documents and queries are represented by vectors. A vector is obtained for each document and query from sets of terms with associated weights. The document and query representatives are considered as vectors in $t$ dimensional space, where $t$ is the number of unique terms in the document collection, then a vector similarity function, such as the inner product or the *cosine measure* can be used to compare document and query representatives.

In order to provide the term weights used for these processes the following parameters are required.

(i) *inverse document frequency* measured by $log(N/f_j)$ where $N$ is the number of documents in the database and $f_j$ is the number of documents that contain term $t_j$.

(ii) *within document frequency*, $tf_{ij}$, being the number of occurrences of term $t_j$ in document $i$.

With these measures, we can express the cosine similarity measure, $sim(Q, D_i)$ between a query $Q = (q_1, q_2, ..., q_t)$ and document $D_i = (d_{i1}, d_{i2}, ..., d_{it})$ as

$$sim(Q, D_i) = \frac{\sum_{j=1}^{t} q_j d_{ij}}{\left(\sum_{j=1}^{t} q_j^2\right)^{1/2} \left(\sum_{j=1}^{t} d_{ij}^2\right)^{1/2}}$$

where the documents weights

$$d_{ij} = tf_{ij} \cdot log(N/f_j)$$

and the query weights

$$q_j = \begin{cases} log(N/f_j) & if\ term\ t_j\ appears\ in\ the\ query \\ 0 & otherwise \end{cases}$$

Documents are then ranked in order of their similarity to the query. See [Salton89] for further discussion of the vector space model.

## 3 Relevance Feedback

Relevance feedback techniques provide for the automatic reformulation and improvement of the original

search request based on information obtained from the user about the relevance of documents retrieved. In general, once a document is presented to a user, the user provides a judgment of whether this document is relevant. Having done so, the original query is modified to incorporate all terms that appear in relevant documents, and the weights of all query terms are modified based on relevant and irrelevant documents.

Suppose 20 documents have been ranked and that they have rank ordering:

1 [2] [3] 4 5 6 [7] 8 9 10 11 [12] 13 14 15 16 17 18 19 [20]

Suppose that documents 2 3 7 12 and 20 are relevant. The precision at 20%, 40%, 60%, 80% and 100% is 0.50, 0.67, 0.43, 0.33 and 0.25 respectively. This gives an average precision of 0.44. Suppose that the first 5 documents are viewed and judged. Thus documents 1, 4 and 5 are judged to be irrelevant, so should have lowest rank. Thus, the documents may be re-ordered given this information, into the sequence:

[2] [3] 6 [7] 8 9 10 11 [12] 13 14 15 16 17 18 19 [20] 1 4 5

What happens now? The effect of changing the rank of some of the documents will, without any further modification and without any benefit to the user, improve the average precision to 0.70. This is known as the **ranking effect**. If as a result of using the relevance judgments, the weights of the terms are modified to produce a new ordering:

[2] [3] [7] 6 8 10 11 9 [12] 13 15 [20] 16 17 14 18 19 1 4 5

The new average precision is 0.77. This improvement is known as the **feedback effect**. Thus, the overall improvement from 0.44 to 0.77 is mainly due to an effect that makes no difference to the order in which the user sees the documents. This problem needs to be addressed.

There are four strategies discussed in [Chang]. They are known as the "full freezing", the "partial rank freezing", the "residual collection" and the "test and control" methods. Briefly, the full freezing strategy means that the rank of a document remains unchanged once it is viewed. Partial rank freezing means that only relevant documents' rank are frozen. The residual collection method approach is to discard all viewed documents both from the collection and the list of relevant documents once they have been viewed. Results are calculated for the remaining documents. The test and control method is to split a collection into two parts, using the first part for feedback, with a consequent modification of term weights, and the second part for comparing retrieval performance.

We believe that the ranking effect is significant in the partial rank freezing method and that the collections that we are using are sufficiently small to make the test and control method a difficult one to implement.

We give results using both of the other techniques as appropriate. Both techniques have disadvantages, but neither displays a ranking effect. The full freezing method limits the improvement obtainable by relevance feedback, since recall/precision figures are dominated by the first few documents - ones that may have already been viewed. The residual collection method concentrates on the part of the collection that we are interested in but may make comparison of different strategies impossible. For instance, if we wish to see whether it is preferable to view 5 documents, re-rank, view another 5 documents and re-rank, versus view 10 documents and re-rank, there may be different collections that are being examined after 10 judgments, so that figures obtained are incomparable.

# 4 The Neural Network Model

Neural network models consist of a collection of simple processing nodes and the connections between them. At each moment in time, each node has a certain *activation level.* Each connection between pairs of nodes has a certain *connection weight.* Nodes communicate by sending

signals to their neighbours via these connections, signals whose strength depends on the current activation level of the sending node. The activation strength of a node at the next moment in time depends on its current activation level, the strengths of the signals being sent to it, and the weights of the connections along which the signals are being sent.

Thus each node affects the activation levels of its neighbours, producing a dynamic pattern of activations over the network. It is possible for particular nodes to be "clamped", which means that their activation levels remain fixed. For a fuller description of the basic concepts of neural network models, see e.g. [McLelland]

We use a neural network model that encapsulates the relationships between documents in the database and terms that they contain. Thus, if we have a database of 5 documents where the sentences "Cats and dogs eat.", "The dog has a mouse.", "Mice eat anything.", "Cats play with mice and rats." and "Cats play with rats." are in documents D1 to D5 respectively, given the query, "Do cats play with mice?" we have the following network (Figure 1).

The network has one node for each term in the documents, one for each document in the database, and one node for each query term. There is a bidirectional connection between each query term node and the corresponding document term node, if it exists. The weight of this connection for term $j$ is denoted $wq_j$. There is a bidirectional connection between a document node and each of the term nodes corresponding to terms in the document. The weight of the connection between the $j$th term node and the $i$th document node is denoted $w_{ij}$. There are no connections between document term nodes or between document nodes. Thus the nodes are divided into three distinct *pools*. (A fourth pool is created if relevance feedback is used.)

In our model, both activation levels and connection weights are real numbers in the range -1.0 to 1.0. The possibility of developing learned connection weights based on a large number of queries, and the desired doc-

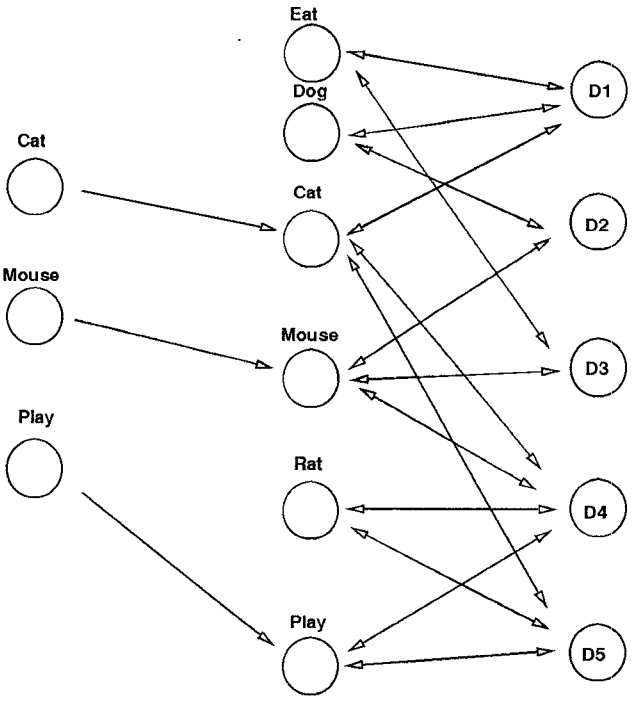


Figure 1: Sample Document Network

ument weights is infeasible given the size of the network. The connection weights are determined using techniques developed for information retrieval and are fixed. The connection between a term node and a document node, reflects the relative significance of the term in the document. If a term appears several times in the document, it has a higher weight than if it appears just once. If a term appears in few documents it has higher weight than a term that is in many documents. Finally, the connection weight should reflect how many terms are connected to a document. Thus the connection weight in terms of the parameters described in the previous section is is given by:

$$w_{ij} = \frac{d_{ij}}{\left(\sum_{j=1}^{t} d_{ij}^2\right)^{1/2}}$$

Signals are received from one pool of nodes at a time from surrounding pools. Each pool is successively reactivated. Thus if at one moment signals are sent from the term nodes to the document nodes, modifying the activation levels of the document nodes, then at the

next moment signals are sent from the document nodes to the term nodes, and vice versa.

A node generates a signal if its activation has reached a threshold. The signal from a node to a connected node is its current activation level. The strength of the signal when it arrives at the connected node is the product of the sending node's current activation with the weight of the connection. The activation level of a node at the next moment in time is obtained solely from the input from connecting nodes, except that this value is then "clipped" to a value in the interval [-1.0, 1.0].

# 5 Usage

The document ranking system described above is very flexible. Firstly, it allows for standard document ranking, as determined by the cosine measure described earlier. Secondly, it allows for the system to find words that appear to be relevant on the basis of initial ranking, and use those words to refine the document ranking. Finally, it allows for a simple incorporation of relevance feedback in the vector space model.

The standard cosine measure is used as follows. A query is converted into a set of terms, which may be represented by the vector $Q = (q_1, q_2, ..., q_t)$. For each term we create a query node and a connection to the corresponding document term node if it exists. The weight of the query nodes is fixed at 1.0. The connection weights are given by

$$wq_j = \frac{q_j}{\left(\sum_{j=1}^{t} q_j^2\right)^{1/2}}$$

Thus only connections with non-zero weight are created. Now when the activation of the document nodes is calculated for the first time, only document term nodes connected to query term nodes will be activated, and so their activation will be the sum of the query node activations times the corresponding document term node - document node connection weights. This sum is

$$\frac{\sum_{j=1}^{t} q_j d_{ij}}{\left(\sum_{j=1}^{t} q_j^2\right)^{1/2} \left(\sum_{j=1}^{t} d_{ij}^2\right)^{1/2}}$$

Thus the standard cosine measure is used.

The process does not have to stop here however. The current activation of the document term node activations represents *initial activations* only. Thus, having generated activations for the documents, these activations may used to calculate the new activations for the document term nodes, in conjunction with the query nodes. Consequently a term that appears in the most activated documents, despite not appearing in the original query, may become active and may activate other documents.

This represents a form of in-built thesaurus. It is not like a conventional thesaurus however, in that terms are grouped on the basis of common documents rather than common meaning. This means that in a set of documents concerning graphs, "node" and "edge" are more likely to be linked than "node" and "vertex". The hope is that two terms that are common to a number of documents discuss the same *subject*.

Finally, relevance feedback can be very easily incorporated into this system. Whenever a document is found to be the most highly activated not yet displayed document, it may be displayed to the user. The user may give a rating to the document. When a rating is given a new node and a connection between the new node and the document node are created with the connection weight being 1.0 and the user rating being transformed into an activation for the new node. This activation is "clamped", so that it may not change in subsequent calculations, and the associated document may not be shown again. This activation will, if it is high, further activate terms in that document, and if it is low, dampen the activation of those terms.

# 6 Previous Work

An early implementation of a document retrieval system using neural networks is described in [Mozer]. As with our model, the network consists of document and term nodes. There are weighted connections between

document and term nodes. As well, there are inhibitory connections between document nodes, and a general decay of activation. This model was implemented and extended in [Bein]. Experiments were carried out to test its robustness with respect to queries. The AIR system is described in [Belew]. In this system attribute nodes are used that include both term and author nodes. The weights of the connections are chosen so as to conserve the overall activity of the network. It also introduces a learning scheme to improve its performance as it models the behaviour of its user population. In [Kwok] a similar system is described that uses a probabilistic model of retrieval.

While not a neural network, a highly parallel retrieval system of note is implemented on a Connection Machine [Stanfill]. In this model, the terms and documents may be regarded as similarly connected. However the weights of the connections are all 1. The query terms are assigned weights based on inverse document frequencies, and the query is augmented by terms in documents selected via relevance feedback.

All of these contributions note that using such a system, it is easy to allow query by example and incorporate relevance feedback. None of these systems, however, describe their performance in comparison to other matching algorithms such as the standard cosine measure.

# 7 Results

This system was tested by using a standard document collection developed by Salton. It consists of 3,204 abstracts of the Communications of the ACM. A total of 52 queries have been generated and judgments on the relevance of each document to each query have been made. This allows recall and precision figures to be obtained. (Confirmation of these results were obtained using the Cranfield database and a set of articles from TIME Magazine.)

These documents result in a network with 9,000 nodes and 130,000 connections. The network is implemented using no secondary storage on an Encore Multimax 520. A full iteration, where terms activate documents, and vice versa, takes about 3 seconds.

A number of tests were used examining which of the adjustments to the node weights were most appropriate. We do not feel that we have the definitive formula! We are more confident of the initial connection weights due to their comprehensive testing in the information retrieval environment.

Our first tests were based on considerations of neural network strategies that had been successfully used in other situations. Having fixed the weights of the connections as described above, we allowed activation to flow backwards and forwards between the term nodes and the document nodes, examining the induced ordering of the documents based on the corresponding node activations at each stage. We found an initial modest improvement and then all nodes became increasingly activated leading to random ordering.

It was not possible to deal with this general increase in activation using inhibitory connections, since we were not trying to obtain a single document but an ordering. Thus we damped the activation at each stage, meaning that unless a node received continuing input its activation tended to 0. This did not help, since the general spread of activation was diminished in size but not in generality. A number of other tactics were tried that made minor differences but gave no substantial performance gains.

At this stage, we went back to the information retrieval literature to ask what is the ideal that we are aiming for. Given that we are basing this work on a vector space model, the answer is provided in [Rocchio]. In this paper he describes an optimal query and how to successively approximate this query via relevance feedback. At this stage we were not using relevance feedback, however we had some idea of the relevance of documents, since the cosine measure provides a rough approximation. Thus we sought to incorporate his ideas in the network.

First we decided to introduce a threshold before a node

could influence its neighbours. This notion has often been used in other neural network simulation and is justified here on the ground that only documents with a significant activation are good candidates for modifying the query term activations. Next we had to determine how much the original query terms could be modified, and to what extent new terms would be activated. Again we were guided by [Rocchio] and subsequent work. He suggested that if D is a document collection and $REL(D)$ is the subset that have been judged relevant and $IRR(D)$ is the subset that have been judged relevant, then a new query is created from the old as follows:

$$Q_{new} = Q_{old} + \frac{1}{\mid REL(D) \mid} \sum_{D_i \in REL(D)} D_i - \frac{1}{\mid IRR(D) \mid} \sum_{D_i \in IRR(D)} D_i$$

We came up with the following formula for the activation of the $i^{th}$ term.

$$q_i + \alpha \times \frac{\sum_{j \in Pos} a_j d_{ij}}{\sum_{j \in Pos} d_{ij}} + \beta \times \frac{\sum_{j \in Neg} a_j d_{ij}}{\sum_{j \in Neg} d_{ij}}$$

where $a_j$ is the activation of the $j^{th}$ document, Pos is the set of j's such that $a_j > T$, and Neg is the set of j's such that $a_j < -T$, where T is a threshold value between 0 and 1.

This has fairly grave consequences for the neural net, since the $a_j$'s are not available to the $i^{th}$ term node in the original architecture. Thus a new connections between each document node and each term node of weight 1 must be inserted to make this information available. The results however justify this modification. Letting T = 0.2, $\alpha = 0.25$ and $\beta = 0.05$, we show the results of three tests. In the first test we reproduce the standard cosine measure ranking. Following this, we give the result of letting the activation spread for one and two iterations. (The network stabilises very quickly.) Table 1 shows the average precision figures for the 52 queries that we have obtained at 10 different recall levels.

| Recall | Cosine | 1 Iter. | 2 Iter. | 20 Iter. |
|---|---|---|---|---|
| 10% | 0.4892 | 0.5601 | 0.6036 | 0.5987 |
| 20% | 0.4240 | 0.4678 | 0.4961 | 0.5074 |
| 30% | 0.3714 | 0.3918 | 0.4104 | 0.4074 |
| 40% | 0.2994 | 0.3397 | 0.3478 | 0.3488 |
| 50% | 0.2368 | 0.2794 | 0.2774 | 0.2648 |
| 60% | 0.1884 | 0.2194 | 0.2266 | 0.2090 |
| 70% | 0.1610 | 0.1750 | 0.1758 | 0.1627 |
| 80% | 0.1288 | 0.1424 | 0.1363 | 0.1228 |
| 90% | 0.0913 | 0.0950 | 0.0886 | 0.0878 |
| 100% | 0.0796 | 0.0797 | 0.0713 | 0.0733 |
| Av. | 0.2470 | 0.2750 | 0.2834 | 0.2783 |

Table 1.

We found that the values for T, $\alpha$ and $\beta$ significantly affected the results and confirmed other studies' findings with regard to the value for $\alpha$. We also found that allowing many iterations meant that very gradually performance deteriorated. Having optimised the performance for the CACM collection, we performed similar tests on the Cranfield and TIME collections. A similar pattern of quick improvement and the gradual deterioration was observed. However the level of improvement was lower. There was a 4% improvement for the Cranfield collection and a 2% improvement for the TIME collection.

Next we examined the network's ability to deal with relevance feedback. First we give the recall and precision figures using the residual collection method of calculation after viewing 5 documents and before performing any re-ordering. This gives us a base case. Next we give the results of applying the classical feedback formula [Salton89] p.320 having viewed 5 documents. Thus, only the terms in viewed documents modify the weight of the terms. Following this, we give the result of letting the activation spread for one and two iterations using the formula given above instead of using the classical feedback formula. In the case of the viewed documents, $a_j = 1$ if the document is relevant and -1 if it is not. The results are given in table 2.

| | Residual | Rocchio | 1 Iter. | 2 Iter. |
|---|---|---|---|---|
| Av. | 0.1525 | 0.1814 | 0.1935 | 0.2052 |

Table 2.

Given that feedback is useful, the issue of when to perform the feedback and when to let the activation flow becomes significant. Is it better to look at 1 document then let activation flow once, 10 times, or is it better to look at 5 documents then let activation flow once, 2 times? In this study we need to give the figures using the full freezing technique for reasons outlined earlier.

| | Cosine | 10×1 doc. | 5×2 docs | 1×10 docs |
|---|---|---|---|---|
| Av. | 0.2470 | 0.2776 | 0.2769 | 0.2739 |

Table 3.

Notice that the average precision figures given in table 3 may be compared with those in table 1. We see that using relevance feedback provides less improvement, than simply allowing activation to flow with no user input.

Other investigations were carried out. Given the threshold for document nodes firing, we investigated introducing a threshold for term nodes. There was no corresponding advantage to be obtained. Also, we investigated whether introducing minimum and/or maximum frequencies for terms to be used in the network was considered. No advantage was obtained.

To give an idea of the significance of these results, many other ways have been suggested for improving recall and precision figures. A quite successful strategy is to introduce pairs of words as terms for describing the documents. The introduction of pairs generates an improvement of the order of 15%. Note that our figures show even greater improvement. (We could easily incorporate the strategy of using pairs as terms if desired.)

# 8 Further Work

A simple refinement is to introduce a layer of new nodes that represent groups of terms appearing in several documents. Connections between the term and the corresponding documents would be deleted, with connections between the term and the group, and the group and the document replacing them.

A similar refinement would be to develop *subject* nodes which would be connected to documents that are related to the subject, and terms that are used describing the subject.

These methods would probably require extensive training sets to determine appropriate connection weights.

A most important further development would be to incorporate this model as the ranking portion of a Boolean based system. A number of authors have suggested that a system based solely on ranking may be computationally infeasible. However, a Boolean query can be used to isolate, say, a 1,000 document subset for the purposes of ranking. This neural network structure may be appropriate in this context.

# 9 Conclusions

We have shown that a neural net structure can be used for ranking documents in a flexible fashion that allows for a variety of inputs to influence the final ranking.

The standard cosine measure of classical information retrieval may be used. We have seen that by allowing the activation to spread through related terms, retrieval performance may improve.

We have given a new method of incorporating relevance feedback in the vector space model, giving similar performance gains. Controversially then we offer the tentative conclusion that the user interaction required for relevance feedback is unnecessary, since the performance gains are no better than that provided by the spreading activation described.

A finding that we believe to be significant is that all major improvements to the performance of the network were based on strategies that have been shown to be of value already in implementing the vector space model.

## References


[Bein] J. Bein, P. Smolensky *Application of the Interactive Activation Model to Document Retrieval* Technical Report CU-CS-405-88, University of Colorado, Boulder, Colorado, 1988

[Belew] R. K. Belew, *Adaptive Information Retrieval* 12th International Conference on Research and development in Information Retrieval, Cambridge, Massachusetts, 1989 pp. 11-20

[Chang] Y. K. Chang, C. Cirillo and J. Razon, *Evaluation of Feedback Retrieval Using Modified Freezing, Residual Collection, and Test and Control Groups* in The SMART RETRIEVAL SYSTEM, G. Salton Ed. Prentice Hall, New Jersey, 1971, pp. 355-370

[Kwok] K. L. Kwok, *A Neural Network for Probabilistic Information Retrieval* 12th International Conference on Research and development in Information Retrieval, Cambridge, Massachusetts, 1989 pp. 21-30

[McLelland] J. L. McLelland and D. E. Rumelhart, *Parallel Distributed Processing* MIT Press, Massachusetts, 1986

[Mozer] M. Mozer, *Inductive Information Retrieval using Parallel Distributed Computation*, Technical Report, ICS, UCSD, La Jolla, California, 1984

[Rocchio] J. J. Rocchio, *Relevance Feedback in Information Retrieval* in The SMART RETRIEVAL SYSTEM, G. Salton Ed. Prentice Hall, New Jersey, 1971, pp. 355-370

[Salton85] G. Salton, E. A. Fox and E. Voorhees, *Advanced Feedback Methods in Information Retrieval*, Journal of the American Society for Information Science, Vol. **36**, No. 3, 1985, pp. 200-210

[Salton89] G. Salton *Automatic Text Processing.* Addison-Wesley, Reading, Massachusetts, 1989

[Stanfill] C. Stanfill and B. Kahle, *Parallel Free-Text Search on the Connection Machine System*, Communications of the ACM, Vol. **29**, No. 12, 1986, pp. 1229-1239