Extending the Rocchio Relevance Feedback Algorithm to Provide Contextual Retrieval

Chris Jordan and Carolyn Watters

Dalhousie University, Faculty of Computer Science, 6050 University Avenue, Halifax, Nova Scotia, Canada, B3H 1W5
{cjordan, watters}@cs.dal.ca

Abstract. Contextual retrieval supports differences amongst users in their information seeking requests. The Web, which is very dynamic and nearly universally accessible, is an environment in which it is increasingly difficult for users to find documents that satisfy their specific information needs. This problem is amplified as users tend to use short queries. Contextual retrieval attempts to address this problem by incorporating knowledge about the user and past retrieval results in the search process. In this paper we explore a feedback technique based on the Rocchio algorithm that significantly reduces demands on the user while maintaining comparable performance on the Reuters-21578 corpus.

1 Introduction

Vast amounts of information are now widely accessible on the Web. At the same time, users are finding it harder and harder to satisfy specific information needs. Search engines provide a one-stop approach and are heavily relied on to access this information [2]. A shortcoming of the generic search engine is that it does not use knowledge about its users or about the data sources it covers. Users, however, may have significantly different preferences and needs even when using similar queries due to their education, interests, and previous experiences. Environmental factors such as these form a context [12] from which a query is issued. Incorporating this information into the search process will improve retrieval results at the individual level.

Typical search engines [2] rely solely on the user's query to select information for them. This is challenging as users normally use queries that consist of 1 to 3 terms and as such typically contain very little contextual information [11]. Not surprisingly, these queries are prone to ambiguity thus resulting in the retrieval of irrelevant documents for the user. Using characteristics about the user or specific feedback from them to modify the search query creates a context for those queries which can help remove ambiguities.

The Rocchio algorithm [3], which uses user feedback on the relevancy of retrieved documents, has been shown to improve search query results. Terms that occur in the retrieved items that are rated by the user are used to modify the query to, essentially, try to retrieve documents that are more relevant. A drawback of the Rocchio method is that users are required to cooperate in rating retrieved items as relevant or not through multiple iterations.

This paper will examine a new approach to automatic query modification for contextual retrieval, based on the Rocchio relevance feedback algorithm, called the Extended Rocchio algorithm. It will be shown that this approach improves document retrieval performance over standard Vector Space Retrieval (VSR) and has results comparable to the Rocchio algorithm while minimizing user interaction.

2 Background and Related Work

Filtering [3], recommending [14][15], and query modification [11] techniques have been used successfully in document retrieval systems to exploit knowledge about users to improve retrieval performance. Filtering and recommender systems use a profile of user interests to select relevant items from a retrieved document set or to generate a query that will retrieve an appropriate one. Query modification, however, responds to *ad hoc* queries presented by the user to improve the retrieved results for that user. Filtering, recommending and query modification all use profiles and document representations that are defined as sets of terms, weighted or not weighted. In this paper we concentrate on weighted terms consistent with the widely used Vector Space Model (VSM) [3]. Consequently, both profiles and documents are represented by vectors of term weights, where the term weight indicates the importance of that term in that profile or document.

Profiles are typically generated either directly by the user or by an analysis of documents rated by them [9]. User profiles can be most easily built by asking the user to fill out some form of questionnaire regarding their interests. User profiles can also be constructed from the content of documents that have been rated in some type of relevance feedback. These profiles can then be used in the search process to augment the query or to rank the retrieved documents. As well, they can be used in a collaborative process to identify groups of users with similar interests [16].

Profiles of interest often rely on feedback from users to rate the relevancy of items that have been retrieved for them. There are two general ways of doing this: direct and indirect [7]. Direct approaches outright ask users for input giving them the opportunity to rate the relevancy of retrieved items after each query. Indirect approaches gather information by tracking user interactions covertly often using server logs or keystroke analyses. Spink [19] has shown that Web users typically are engaged in three separate information tasks during each query session; profiles for this environment must be capable of handling multiple simultaneous user interests.

Given vectors of term weights for both documents and profiles there are two general approaches to using these vectors in the retrieval process; comparison of term vectors and machine learning. The term vector approach uses the term frequency/inverse document frequency (TF/IDF) value [3]. WebMate [6], for example, uses a single tier architecture where profiles can contain a maximum of N term vectors. Documents that are rated relevant by the user are converted into feedback term vectors. If the number of vectors in the profile is less than N then feedback vectors are add as new term vectors to the profile. Otherwise the two most similar vectors in the profile are merged together. Alipes [20], a similar system, allows negative relevancy

judgments in addition to positive ones, in a three tier architecture representing short term positive, short term negative, and long term interests.

Machine learning techniques are frequently employed to build and use profiles of interest. News Dude [5], for example, uses a two tiered architecture to map short and long term interests respectively in a user profile. The short term interests are derived from the k previously rated documents. News stories are then classified as interesting or not by running a nearest neighbor algorithm on these k documents. If a story can not be classified using the short term tier then it is passed to the long term tier where it is classified using a Naïve Bayesian classifier. Mooney [15] developed a system called Libra to recommend books to users from a snapshot of Amazon's online catalogue using a single tier Naïve Bayesian classifier.

Rocchio's algorithm [3] is a query modification method using relevance judgments and has been used extensively in the retrieval of documents. The user provides relevance feedback on an initial set of retrieved results for a given query. This feedback is used to alter the query to better represent their current information need and return improved retrieval results. The algorithm is generally stated as:

$$Q_{i+1} = \alpha * Q_i + \beta * D_{pos}/n_{pos} - \gamma * D_{neg}/n_{neg} \quad (\mathbf{1})$$

Where Q represents a query, D_{pos} represents the set of returned documents rated as relevant, D_{neg} represents the set of returned documents rated as not relevant, n is the cardinality of D, and α, β, and γ are constants that regulate the weight each component has on the formation of the new query. Typically $\alpha=1$ and $\beta + \gamma = 1$. The shortcoming of this algorithm is that it works only on a per query basis, so each time a user issues a new query a new model has to be constructed. Allan [1] showed, however, that feedback gathered over time will still have positive effects on performance using an incremental relevance feedback approach. In [13] Rocchio was used adaptively to predict what document categories users are interested in. The Extended Rocchio algorithm presented in this paper builds on these results.

The Rocchio algorithm has been used previously for text classification [8][17] on the Reuters-21578 corpus [22]. The variants of the Rocchio algorithm used in these papers indicated that they are useful in text classification even though they are outperformed slightly by machine learning techniques. A significant advantage of the Rocchio algorithm is that it has linear run time complexity. A distinction between text classification and contextual retrieval is that in classification a training set of documents is used to define term vectors for the categories in the corpus; these term vectors are then used to classify the documents in the test set. These learned term vectors can be thought of as derived queries. In contextual retrieval however, the user creates initial queries, which are improved by the incorporation of information based on relevance feedback.

Concept hierarchies have been used to add contextual information to queries. ARCH [18] uses concept term vectors based on the Yahoo concept hierarchy for query modification. Rocchio's algorithm was employed using concepts term vectors instead of document term vectors; this approach does show promise however the reported results are based on a small population. Concept term vectors in [13] are utilized in a general profile that is accessed by all users; an adaptive Rocchio algorithm is also made use of to generate individual user profiles.

3 Approach

The relevance feedback that users give in the Rocchio algorithm is used to fine tune a particular query. We theorize that this feedback represents the context from which the query was issued. In this work we proposed an Extended Rocchio algorithm that preserves this contextual information by constructing a profile of term vectors using the TF/IDF model. In this way, multiple query contexts can be profiled for the user instead of only one as in the traditional Rocchio algorithm. Each term vector represents a particular context that the user may issue queries from. For example, a user having two interests, automobiles and the stock market, would have a profile that contains two term vectors as they represent two different contexts that the user issues queries from. As the user rates documents as relevant or not new contextual information is introduced into the profile by the addition of new term vectors and the modification of existing ones.

There are three steps in this approach:

1. **Query Modification:** When a user issues a query, the query terms are compared to the profile term vectors. The similarity is calculated using the formula for the cosine of the angle between two vectors [3]:

$$Sim(Q,V) = \frac{Q \bullet V}{|Q| \times |V|} \quad \textbf{(2)}$$

   Q and V represent the query and a profile term vector respectively. If there does not exist a term vector with a great enough similarity then no query modification takes place.

   If the most similar term vector, V, in the profile has a similarity greater than a threshold σ, a modified query, Q_{mod}, is generated by combining the initial query with this vector; the average weight is used for the terms that the vector and query have non-zero values for.

2. **Relevance Feedback:** This modified query is used to perform the document retrieval. The user can then give feedback by rating a subset of the retrieved results. The rated documents are used to form three term vectors:
   a) Term vector P contains the average term weights for terms used in the relevant documents that do not occur in the original query, Q.
   b) Term vector N contains the average term weights for terms used in negatively rated documents that do not occur in the query, Q.
   c) Term vector F contains the terms in V that are not in P, N, or Q.

   P and N do not include terms in the original query Q as it assumed that since the user chose those terms they accurately represent the user's information need and should remain in the query. An additional reason for handling query terms separately is that it helps preserve the similarity between V and Q; this is important if this query is reissued by the user in step 1.

3. **Profile Modification:** The term vectors created in step 2 are used to modify the profile. If the original query was not modified in step 1 then a new term vector is added to the profile using the Rocchio relevance feedback algorithm:

$$V_{new} = \alpha*Q + \beta*P - \gamma*N \quad \textbf{(3)}$$

Otherwise, V is replaced using the formula below:

$$V = \alpha * Q_{mod} + \beta * P - \gamma * N + \Delta * F \quad (4)$$

Where α, β, and γ are constants analogous to the ones in the Rocchio algorithm. Δ is a constant that regulates the rate of decay of terms that are not used in a profile. It is important to have some form of decay in the profile to reflect the dynamic aspects of a user's context such as interest drift [10]. Decay allows contextual information that is out of date to be removed from the profile.

For the trials performed in this paper: α is set to 1; β is set to .75; γ is set to .25; σ is set to .25; and Δ is set to .5. These were arbitrarily determined based on trial and error with the data set. To speed up the similarity calculations, a minimum weight of 0.1 was set for terms in the profile and only the 5 most heavily weighted terms from each feedback vector created in step 2 are used in modifying the profile in step 3.

4 Dataset

The dataset used in this paper was the Reuters-21578, Distribution 1.0 [22], a set of Reuters financial news stories. For experimentation purposes, only two subsets of this corpus were examined, the TRAIN LEWISSPLIT subset of 13,625 documents and the TEST LEWISSPLIT subset of 6,188 documents. We used a standard vector space model (VSM) and a standard Rocchio relevance feedback system to provide a baseline for comparison to the Extended Rocchio system. The TRAIN subset was used in the Extended Rocchio approach to generate the user profile and the TEST subset was used for evaluation. Consequently, the comparison between the Extended Rocchio method and the baseline systems was done using only the TEST subset.

The documents are also tagged by simple categorization schemes, including TOPICS and PLACES which were used during the experimentation process. The TOPICS labels reflect subject matter and the PLACES labels are typically country names. Below is an abbreviated sample document from the corpus.

<REUTERS TOPICS="YES" LEWISSPLIT="TRAIN" CGISPLIT="TRAINING-SET" OLDID="5544" NEWID="1">
<DATE>26-FEB-1987 15:01:01.79</DATE><TOPICS><D>cocoa</D></TOPICS>
<PLACES><D>el-salvador</D><D>usa</D><D>uruguay</D></PLACES>
<TEXT><TITLE>BAHIA COCOA REVIEW</TITLE>
<BODY>...<BODY></TEXT></REUTERS>

The feature set of the corpus was first reduced by the use of a standard stop word list [21] followed by basic term stemming using Porter's algorithm [3]. For each document, a standard TF/IDF term vector was created from the BODY and TITLE but not the TOPICS or PLACES fields.

5 Experimental Trials

In the experimental trials we used the TOPICS and PLACES data to judge the relevancy of retrieved documents. We defined 45 sets of documents where all the mem-

bers for a given set have a minimum of one TOPICS label and one PLACES label in common and at least 20 are in the TRAIN subset and 10 are in the TEST subset. Four of these sets are described in Table 1.

Table 1. Sample document sets used in the evaluation.(P = PLACES, T = TOPICS)

Desired P label	Desired T label	TRAIN subset			TEST subset		
		#docs with T	#docs with P	#with both	#docs with T	#docs with P	#with both
Canada	acq	1615	725	93	719	304	44
China	Grain	424	150	31	149	55	11
Usa	Wheat	8130	206	130	3918	71	39
Usa	crude	8130	385	166	3918	189	96

We then created two sets of 45 queries that targeted these document sets for the trials. The first set of queries consists of only a single PLACES label. Consequently there are ambiguous queries in this set as a PLACES label may be related to different TOPICS labels in the documents. For example, consider the last two documents sets described in the above table. The third set is described by "PLACES=Usa, TOPICS=wheat" and fourth by "PLACES=Usa, TOPICS=crude". The simple query "Usa" targets documents in both of these sets. This first set of queries is designed to be missing significant contextual information. Queries in the second set contain both a PLACES and a TOPICS label; this reduces the ambiguity by incorporating more contextual information.

The test queries have been purposely designed to be short for two reasons. The first is that it allows documents to have their relevancy judge based on their set defining labels. The second reason is that users traditionally only issue queries that consist of 1 to 3 terms [11]. Although it has been acknowledged that longer queries tend to improve retrieval performance [4], the use of short queries makes these trials representative of actual user behavior.

For these experimental trials an automatic method for relevance feedback was developed. When no feedback is given for a particular query the results are equivalent to a VSM system. Depending on the query, there may not be any relevant documents in the top 10 retrieved results. While judging only negative documents does help improve the results, more significant improvements are attained when a document is rated to be positive because there are typically far fewer relevant documents than irrelevant ones. The approach adopted in this work is as follows:

1. The first document retrieved from the search that meets the PLACES and TOPICS label criteria is marked as relevant.
2. The search is performed again using the term vector of this document as feedback.
3. The top five results from this search are then judged as relevant or not based on the label criteria. The query is modified by the system.
4. The search is executed one final time.

For the Rocchio benchmark, the affect of the Rocchio relevance feedback system is measured after this fourth step. For the Extended Rocchio system a profile is created and modified from feedback on the TRAIN subset. A new profile is used for every query to remove the effect of the order that the queries are in during the experiment. The retrieval performance of the Extended Rocchio system is measured by using

these profiles in combination with their respective initial query on the TEST subset; relevance feedback is only given on the TRAIN subset.

6 Results

Trail 1: The queries in the first set are simply PLACES labels. A few examples of the queries in this set are "Canada", "China", and "Usa". Table 2 shows the average precision in the top 5 and top 10 retrieval results for each query.

Table 2. Retrieval performances using a PLACES label as the query (average precision).

TRAIN subset				TEST subset					
VSM		Rocchio		VSM		Rocchio		Ext-Rocchio	
Top 5	Top 10	Top 5	Top 10	Top 5	Top 10	Top 5	Top 10	Top 5	Top 10
12.0	12.2	54.6	46.0	14.6	14.0	49.4	36.4	31.6	26.9

The results from a randomized block design, shown in Table 3, on both the performances over the Top 5 and Top 10 retrieval results indicate there is a significant difference between the systems with $p < .001$.

Table 3. Results from the randomized block design on the results from first set of queries.

	degrees of freedom	F	p value
Top 5	44	42.819	< .001
Top 10	44	8.706	< .001

Given the significant difference between the systems, a series of paired sample t tests were conducted. The outcome of these tests is shown in Table 4. These results indicate that the VSM system was significantly outperformed by both the Rocchio and Extended Rocchio systems. As well, the results also indicate that the Rocchio system significantly outperforms the Extended Rocchio system on the simple query set.

Table 4. Paired sample t tests on the results from first set of queries (VSM – Vector Space Model, R – Rocchio, ER – Extended Rocchio) degrees of freedom = 44

	Paired Differences					T	2 tailed p value
	mean	std. dev.	mean std. err.	95% CI of diff. Lower	Upper		
Top 5: R–VSM	1.73	1.50	.22	1.28	2.18	7.76	< .001
Top 5: ER–VSM	.84	1.26	.19	.47	1.22	4.49	< .001
Top 5: R–ER	.89	1.87	.28	.33	1.45	3.18	.003
Top 10: R–VSM	2.24	2.27	.34	1.56	2.93	6.64	< .001
Top 10: ER–VSM	1.29	2.11	.31	.66	1.92	4.10	< .001
Top 10: R–ER	.96	3.07	.46	.034	1.88	2.09	.042

Trail 2: The queries in the second set included both a PLACES and a TOPICS label, with any abbreviations expanded. A few examples of queries in this set are "Can-

ada acquisitions mergers", "China grain", and "Usa wheat". As in the previous trial, the feedback for the Extended Rocchio system is given only on the TRAIN subset.

Table 5. Retrieval performances using more complex queries (average precision).

TRAIN subset				TEST subset					
VSM		Rocchio		VSM		Rocchio		Ext-Rocchio	
Top 5	Top 10	Top 5	Top 10	Top 5	Top 10	Top 5	Top 10	Top 5	Top 10
41.4	42.2	72.0	58.7	40.8	37.1	59.2	46.9	47.6	43.1

The randomized block design shown in Table 6 illustrates that there is a significant difference in performance between systems.

Table 6. Randomized block design on the results from the second set of queries

	degrees of freedom	F	p value
Top 5	44	15.152	< .001
Top 10	44	8.437	< .001

Paired sample t tests indicated that the VSM system was significantly outperformed by both the Rocchio and Extended Rocchio systems for the more complex query set. However, there is no significant difference between the Rocchio and Extended Rocchio in their performance on the Top 10 results with p = .142.

Table 7. Paired sample t tests for second set of queries (VSM – Vector Space Model, R – Rocchio, ER – Extended Rocchio) degrees of freedom = 44

	Paired Differences					t	2 tailed p value
	mean	std. dev.	mean std. err.	95% CI of diff. Lower	Upper		
Top 5: R–VSM	.91	1.20	.18	.55	1.27	5.08	< .001
Top 5: ER–VSM	.33	.98	.15	.0398	.63	2.29	.027
Top 5: R–ER	.58	1.18	.18	.22	.93	3.29	.002
Top 10: R–VSM	.98	1.74	.26	.46	1.50	3.77	< .001
Top 10: ER–VSM	.60	1.37	.20	.19	1.01	2.93	.005
Top 10: R–ER	.38	1.70	.25	-.13	.89	1.49	.142

7 Discussion

Both the Rocchio and the Extended Rocchio algorithms significantly outperform the standard VSM for both the simple and complex queries. The Rocchio and Extended Rocchio are not, however, significantly different. The Extended Rocchio feedback system was able to achieve comparable performance to a traditional Rocchio system without requiring any user feedback after the training period. Furthermore, the Extended Rocchio system supports multiple simultaneous user interests.

The retrieval performances of the VSM and the Rocchio systems on the TRAIN subset are similar to their respective performances on the TEST subset indicating that the observed behaviors are not due to any abnormalities in the subsets. This strengthens the observation that the Extended Rocchio system maintains a positive effect on retrieval performance without requiring user feedback.

The feedback and evaluation scheme used in this paper was done in a very controlled environment. Obviously, there are some articles that represent a relevant document set better than others.

8 Conclusion

Query modification based on user feedback has been shown to be an effective means of improving retrieval performance. In this paper we show that the Extended Rocchio approach, which only uses feedback during a training phase, provides a performance improvement that is comparable to the traditional Rocchio algorithm, which requires user feedback at query time. The Extended Rocchio approach reduces the amount of relevance feedback that a user needs to provide. Furthermore, the Extended Rocchio method builds a profile that provides query modification for multiple user contexts.

The results from this initial experiment are promising. This will need to be followed up by studies using real users and other profile based query modification techniques to better understand the strengths and weaknesses of specific approaches. Using stereotypic information about the user such as age, occupation, and location should also be explored; contextual information of this type is static. This will provide a foundation for developing a hybrid. In the experimental trials reported in this paper, the Δ constant for contextual information decay was not tested although clearly retrieval systems based on profiles of user contexts must respond to both short and long term changes to those contexts as user preferences and interests change over time.

The Extended Rocchio algorithm provides contextual retrieval through query modification of the original query based on the user's profile. This technique helps overcome the short query problem by adding contextually relevant terms contained within the profile. Although this approach shows promise, there is still much work to be done before it can be incorporate into a hybrid system.

Acknowledgements

We would like to thank Qigang Gao and Wade Blanchard, both at Dalhousie University, for their input into this work.

References:

1. Allan, J.: Incremental Relevance Feedback for Information Filtering. Proc. of the 19th Annual Int. ACM SIGIR Conf. on Research and Development in Info. Retri. (1996) 270 – 278

2. Arasu, A., Cho, J., Garcia-Molina, H., Paepcke, A., Raghavan S.: Searching the Web. ACM Transactions on Internet Technology, Vol. 1, No. 1 (2001) 2–43
3. Baeza-Yates, R., Ribeiro-Neto, B.: Modern Information Retrieval. Addison Wesley (1999)
4. Belkin, N.J., Cool, C., Kelly, D., Kim, G., Kim, J.-Y., Lee, H.-J., Muresan, G., Tang, M.-C., Yuan, X.-J.: Query Length in Interactive Information Retrieval. Proc. of the 26th Annual Int. ACM SIGIR Conf. on Research and Development in Info. Retri. (2003) 205–212
5. Billsus, D., Pazzani, M.: A hybrid user model for news story classification. Proceedings of the Seventh International Conference on User Modeling (1999) 99–108
6. Chen, L., Sycara, K.: WebMate: A Personal Agent for Browsing and Searching. Proceedings of the Second International Conference on Autonomous Agents (1998) 132–139
7. Chan, P. K.: Constructing Web User Profiles: A non-invasive Learning Approach. Revised Papers from the Int. Workshop on Web Usage Analysis and User Profiling (1999) 39–55
8. Joachims, T.: A Probabilistic Analysis of the Rocchio Algorithm with TFIDF for Text Categorization. Proceedings of ICML-97 (1997) 143–146
9. Kobsa, A.: User modeling: Recent work, prospects and hazards. In: Schneider-Huchmidt, M., Kiihme, T., Malinowski, U.(ed.): Adaptive User Interfaces: Principles and Practice (1993) 111–128
10. Koychev, I., Schwab, I.: Adaptation to drifting user's interests. Proceedings of ECML2000/MLnet Workshop: Machine Learning in New Information Age (2000) 39–46
11. Kruschwitz, U.: An Adaptable Search System for Collections of Partially Structured Documents. IEEE Intelligent Systems, July/August (2003) 44–52
12. Lawrence, S.: Context in Web Search. IEEE Data Engineering Bulletin, Vol. 23, No. 3 (2000) 25-32
13. Liu, F., Yu, C., Meng, W.: Personalized web search by mapping user queries to categories. Proc. of the 11th Int. Conf. on Information and knowledge management (2002)
14. Middleton, S., DeRoure, D., and Shadbolt, N.: Capturing knowledge of user preferences: Ontologies in recommender systems. Proc. of the 1st Int. Conf. on Knowledge Capture (2001)
15. Mooney, R. J., Roy, L.: Content-based book recommending using learning for text categorization. Proceedings of the Fifth ACM Conference on Digital Libraries. (2000) 195 – 204
16. Resnick, P., Iacovou, N., Suchak, M., Bergstrom, P., Riedl, J.: GroupLens: An open architecture for collaborative filtering of Netnews. Proc. of ACM Conf. on Computer-Supported Cooperative Work (1994) 175–186
17. Schapire, R., Singer, Y., Singhal, A.: Boosting and Rocchio applied to text filtering. Proc. of the 11th Int. Conf. on Research and Development in Information Retrieval (1998) 215 – 223
18. Sieg, A., Mobasher, B., Lytinen, S., Burke, R.: Concept based query enhancement in ARCH. To appear in Proc. of the Int. Conf. on Internet Computing (2003)
19. Spink, A., Ozmutlu, H.C., Ozmutlu, S.: Multitasking information seeking and searching processes. Journal of the American Society for Information Science and Technology, Vol. 53, No. 8 (2002) 639–652
20. Widyantoro, D.H., Ioerger, T.R., Yen, J.: An Adaptive Algorithm for Learning Changes in User Interests. Proc. of the 8th Int. Conf. on Information and Knowledge Management (1999) 405–412
21. Glasgow IDOM - IR linguistic utilities: Stop word list. http://www.dcs.gla.ac.uk/idom/ir_resources/linguistic_utils/. Access: Oct 24, 2003
22. UCI KDD Archive: Reuters-21578 Text Categorization Collection. http://kdd.ics.uci.edu/databases/reuters21578/reuters21578.html. Access: Oct 24, 2003