Improving the Filtering Quality of Selective Dissemination of Information by Observing User Task Behavior (Undergrad SCS senior thesis paper 2005)

Sue Yi Chew
Carnegie Mellon University
5000 Forbes Avenue
Pittsburgh PA 15213 USA
syc@andrew.cmu.edu

Anthony Tomasic
Carnegie Mellon University
5000 Forbes Avenue
Pittsburgh PA 15213 USA
tomasic@cs.cmu.edu

ABSTRACT

A selective dissemination of information (SDI) service alerts users to latest documents in their field of interest. SDI helps users cope better with streams of incoming information by filtering out uninteresting documents. Existing SDI filtering mechanisms typically use user feedback based on a binary 'interesting/not interesting' decision per document. However, in some situations the user will use information from the document in a structured way to complete a task, for example to fill out a form. Observing how users use information provides additional feedback to the SDI system. In this work we describe a method of observing user task behavior to improve the quality of SDI filtering. Our method creates, via machine learning, information extraction models of the information that the user is interested in. We then use these models on incoming documents to extract parts of the document that are likely to be of interest to the user. The results of the extraction are added as features into the filtering algorithm. We describe some experimental results of this method that demonstrate improved filtering performance of the SDI system using our method. We then describe the conditions where this method may be applied.

1. INTRODUCTION

Nowadays we have access to much more electronic information than we need. Selective Dissemination of Information (SDI) (also known as Content-Based Dissemination) systems enable users to cope with the large amounts of information by filtering only documents which are relevant to the particular user. Selective dissemination is similar to search in that the documents filtered are based on a profile, but in this case the user's need persists over some time. This persistence allows for methods of gradually improving the filtering mechanism specific to a certain profile. Some applications of Selective Dissemination of Information (SDI) systems are in newswire for financial analysts, and filtering recent publications to researchers, and spam filtering.

Previous work on SDI has been roughly in two categories. The first is on improving the efficiency of the filtering mechanism from a systems perspective [TG94] [TG99] [DFFT02]. The second is on improving the quality of the filtering, referring to the degree of relevancy between the information sent to a user and the user's interest. Here, there has been two approaches, the first to improve the filtering algorithm, and another is to improve the profile. Improving the profile can be accomplished in batch filtering or adaptive filtering scenario. In batch filtering, the system begins with a large sample of evaluated training documents. In adaptive filtering, the system begins with only with a topic statement and a small number of positive examples. The TREC Filtering Track Report [RS01] [RS02] summarizes the state of the art of these approaches, which include using Rocchio's algorithm [A96] [ZXC03] [ABLMNK02] [OC03], k-nearest neighbor [AY01], language modelling [OC03], support vector machines [MPM02], clustering, neural-networks, EM.

Positive examples for training are usually obtained via user feedback. Existing filtering mechanisms typically use a binary 'interesting/not interesting' feedback per training document as training examples. The filtering mechanism, for example a classification algorithm, typically uses the count of each word from each incoming document as features for classification. However, there are certain tasks in which the user uses information from the document in a structured way. For these tasks, the user can implicitly provide more feedback to the filtering mechanism than simply 'interesting/not interesting' without doing extra labeling work, as the labeling is already embedded in the task that the user performs. Financial analysts extract information about mergers and acquisitions from documents by copying the names of the companies involved into a form or into columns in a spreadsheet. The copied information is implicit feedback about companies of interest to the user. Another example would be personal shopping for a major investment – someone who is looking to purchase a used vehicle may read documents about vehicles for sale and copy the information that is relevant to him into a spreadsheet, for comparison purposes. For example, the person may be reading a newsgroup such as cmu.misc.market (which is a marketplace and general discussion newsgroup local to Carnegie Mellon University), or craigslist [C]. In the newsgroup, everytime the user reads a document about a car for sale, the user copies the relevant information into a table on for comparison. A diagram of this activity is shown in Figure 1, with just 4 spans that is useful to the user shown.

In this paper we describe algorithms and experimental evidence that demonstrate the impact of this implict information on SDI performance. We experiment with using information that is gathered by observing user behavior in this form to improve filtering quality. We hypothesize that if we add these additional features representing the parts of a document that is interesting to the user, to the classification algorithm, the filtering quality will improve. The parts of the document that is interesting to the user

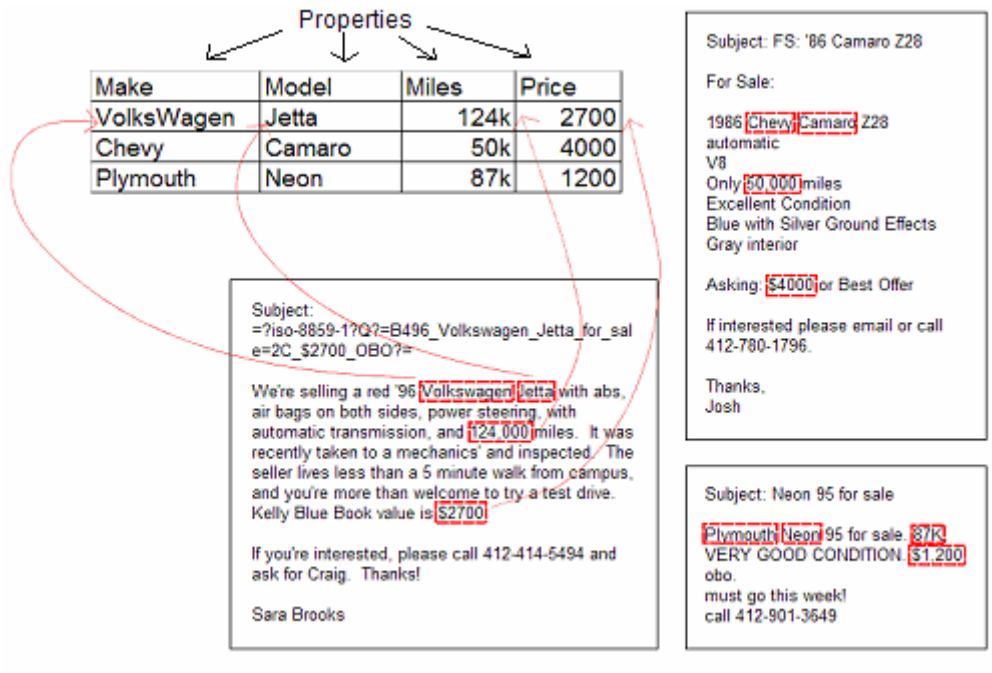

Figure 1: Example of target user model

(The user may also be probably interested in the year the car was produced, the contact phone number, the color of the car, and so on.)

are to be identified by running an information extraction technique on the incoming documents, where the extraction model is trained on the spans that the user labelled as interesting.

A system diagram of the information flow is shown in Figure 2. Previous SDI systems typically use feedback in a way that is corresponding with the dotted box in the figure. The user views documents in a document browser and occasionally marks documents as interesting or not interesting. The marked documents are sent as training examples to a classification learner. The learner improves the relevance of the documents that the document browser displays to the user's interest. We augment this process with the rest of the loop shown in the system diagram. The user copies the parts of the document that makes it interesting to him to an excel spreadsheet. We convert these text spans into mark-up which we feed into an extraction learner. Conversion is accomplished with a heuristic algorithm [TCFZKMHMH04] that labels the substring in the document with the closest match to the copied text span. The context in which the user is interested in a particular span is called its property. We use extraction learners to learn models of the properties. [LMP01] Then, for each incoming document to the SDI system, these models are run on the document to predict which parts of the document is interesting to the user. These predicted spans are used as additional features to the classification learner, in addition to the original features.

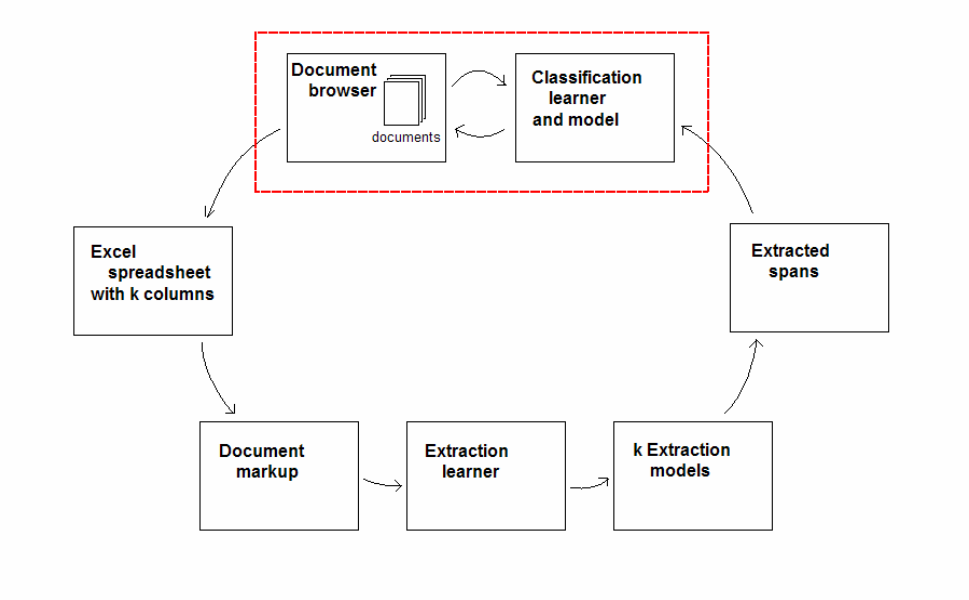

Figure 2: System diagram

2. EXPERIMENTAL METHOD

We tested empirically whether adding the extra information in this form would improve filtering quality. Our courpus is documents from cmu.misc.market, which is a local Carnegie Mellon newsgroup. We hand-classified the documents from cmu.misc.market into 22 categories.

The results listed in Table 1 are obtained from 600 consecutive emails to the newsgroup from January, which covers roughly 3 weeks (11 January to 1 Feb 2005). These 600 documents covered 19 of the 22 categories in cmu.misc.market. There was a definite skew in this dataset. 199/500 (almost two-fifths) of the documents were related to buying or selling books, 21/500 documents were related to buying or selling furniture, and 143/500 were in a miscellaneous category (which is mostly general discussion and flamewars). Figure 3 lists a complete breakdown of the 22 categories and the number of documents in each.

Our experimental method consists of the Control test and the Competitor test. The classification results of each were then compared using precision, recall, F1 and error rate metrics. The definitions of these metrics are as follows -- Precision is the fraction of retrieved documents that are relevant, while recall is the fraction of relevant documents that are retrieved = P(retrieved|relevant). F1 is a combination of the precision and recall measures. Error rate is simply the number of errors made by the classifier (false positives and false negatives) over the number of testing documents.

Control test:

Given a set of documents from cmu.misc.market labeled by category, a classification model was built using multinomial Naïve Bayes to distinguish their categories, where the features to Naïve Bayes are the count of each word in the documents. The generated classification model is then run on a set of testing documents from the corpus.

Competitor test:

Using the same set of documents, we hand-labeled the parts of these documents with the categories listed in Figure 3. An example of the documents we hand-labeled for books documents is shown in Figure 4. When we use the term properties, we mean one of the types of spans that we labeled. The properties that we felt were useful with respect to books were “title”, “course_number”, “course_title”, “price”, “bookstore_new_price” (selling price of a new copy of this book at the CMU bookstore), and “bookstore_used_price”. (selling price of a book) These properties were chosen for the frequency in which they appeared in documents about selling books, in the cmu.misc.market dataset. For documents about selling furniture, we labelled the properties corresponding to “fur_item” (Name of a item being sold), “fur_phonenum” (Contact phone number of the seller), and “fur_price” (Price of a furniture item being sold).

As depicted in the system diagram in Figure 2, we use an CRF Learner to create models for each of the properties. The extraction models are then run on the testing documents to extract spans of text in these documents that are predicted to be interesting to the user. The results of this extraction are added as features into multinomial Naïve Bayes, in addition to the features in the control experiment i.e. count of each word. For each annotator, we use an indicator feature variable to indicate if the annotator detected a span in a particular document.

The Naïve Bayes algorithm for classification and Condition Random Fields (CRF) learner for extraction were run from the Minorthird package. [M]

Description	Symbol	Number of documents
Books – offer	BK_O	65
Books – wanted	BK_W	6
Furniture – offer	FUR_O	23
Furniture – wanted	FUR_W	7
Transport – offer	RIDE_O	1
Transport – wanted	RIDE_W	2
Tutoring – offer	TUT_O	1
Job – offer	JOB_O	4
Lost	LOST	8
Found	FOUND	3
Tickets – offer	TIC_O	4
Tickets – wanted	TIC_W	0
Experiment – offer	EXP_O	15
Apartment – offer	APT_O	12
Apartment – wanted	APT_W	3
Survey	SUR	0
Others	OT	140
Computers – offer	COM_O	67
Computers – wanted	COM_W	5
Event	EVENT	31
Vehicle – offer	VEH_O	3
Vehicle – wanted	VEH_W	0

Figure 3: Categories of documents in cmu.misc.market

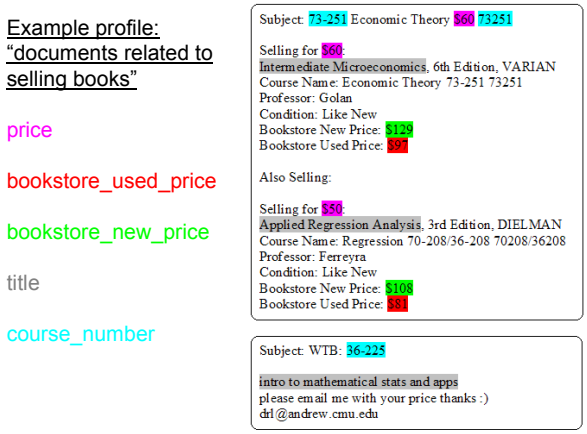

Figure 4: Example labels for a book document

Table 1 : Extraction quality

	Training		Extraction test set (100 documents)									Labels fed to classification test	
Property name	# documents	# true labels	# documents	# true labels	# generated labels	Token P	Token R	Token F1	Span P	Span R	Span F1	# documents	# generated labels
book_price	200	127	100	89	117	0.903	0.866	0.884	0.885	0.865	0.875	400	248
book_bookstore_used_price	200	28	100	18	14	1	0.76	0.864	1	0.778	0.875	400	14
book_bookstore_new_price	200	21	100	13	0	0	0	0	0	0	0	400	0
book_title	200	152	100	110	83	0.878	0.771	0.821	0.8	0.582	0.674	400	113
book_course_number	200	219	100	176	159	0.99	0.909	0.948	0.974	0.852	0.909	400	230
Fur_item	200	48	100	44	0	0	0	0	0	0	0	400	0
Fur_phonenum	200	7	100	3	0	0	0	0	0	0	0	400	0
Fur_price	200	61	100	56	0	0	0	0	0	0	0	400	0

Table 2: Comparison of binary classification between the Control and the Competitor filtering mechanism

(Highlighted rows show where classification performance increased)

	# examples by class		CONTROL					COMPETITOR					F1 Percentage Increase
Category	POS	NEG	Error rate	Recall	Precision	F1	Kappa	Error rate	Recall	Precision	F1	Kappa	
BOOKS	71	329	0.0525	0.704	1	0.826	0.797	0.03	0.831	1	0.908	0.89	9.927
FUR	30	370	0.075	0.1	0.5	0.167	0.145	0.075	0.1	0.5	0.167	0.145	0
OTHERS	268	132	0.1575	0.974	0.823	0.892	0.607	0.15	0.974	0.831	0.897	0.628	0.561
BK_O	65	335	0.055	0.692	0.957	0.804	0.773	0.0275	0.862	0.966	0.911	0.894	13.3
BK_W	6	394	0.015	0	NaN	NaN	0	0.015	0	NaN	NaN	0	NaN
FUR_O	23	377	0.06	0.087	0.4	0.143	0.125	0.06	0.087	0.4	0.143	0.125	0
FUR_W	7	393	0.015	0.143	1	0.25	0.247	0.015	0.143	1	0.25	0.247	0
RIDE_O	1	399	0.0025	0	NaN	NaN	0	0.0025	0	NaN	NaN	0	NaN
RIDE_W	2	398	0.005	0	NaN	NaN	0	0.005	0	NaN	NaN	0	NaN
TUT_O	1	399	0.0025	0	NaN	NaN	0	0.0025	0	NaN	NaN	0	NaN
JOB_O	4	396	0.005	0.5	1	0.667	0.664	0.005	0.5	1	0.667	0.664	0
LOST	8	392	0.02	0.125	0.5	0.2	0.194	0.02	0.125	0.5	0.2	0.194	0
FOUND	3	397	0.0075	0	NaN	NaN	0	0.0075	0	NaN	NaN	0	NaN
TIC_O	4	396	0.01	0	NaN	NaN	0	0.01	0	NaN	NaN	0	NaN
TIC_W	0	400	0	NaN	NaN	NaN	NaN	0	NaN	NaN	NaN	NaN	NaN
EXP_O	15	385	0.015	0.667	0.909	0.769	0.762	0.015	0.667	0.909	0.769	0.762	0
APT_O	12	388	0.0125	0.583	1	0.737	0.731	0.0125	0.583	1	0.737	0.731	0
APT_W	3	397	0.0025	0.667	1	0.8	0.799	0.0025	0.667	1	0.8	0.799	0
SUR	0	400	0	NaN	NaN	NaN	NaN	0	NaN	NaN	NaN	NaN	NaN
OT	140	260	0.14	0.821	0.788	0.804	0.695	0.135	0.821	0.799	0.81	0.705	0.746
COM_O	67	333	0.07	0.657	0.898	0.759	0.719	0.0675	0.6716418	0.9	0.769	0.731	1.32
COM_W	5	395	0.02	0	0	NaN	-0.00946	0.02	0	0	NaN	-0.00946	NaN
EVENT	31	369	0.05	0.419	0.867	0.565	0.542	0.05	0.419	0.867	0.565	0.542	0
VEH_O	3	397	0.0025	0.667	1	0.8	0.799	0.0025	0.667	1	0.8	0.799	0
VEH_W	0	400	0	NaN	NaN	NaN	NaN	0	NaN	NaN	NaN	NaN	NaN

Table 3 : Comparison of multicategory classification between the Control and Competitor filtering mechanisms

Control	Competitor
Error rate	Error rate
0.245	0.2425

3. RESULTS

3.1 Extraction Effectiveness

Table 1 lists the effectiveness of the extraction learning. The CRF Learner algorithm was trained with the first 200 documents from the January dataset, which were labeled with the properties in the table. It was tested with the second 100 documents from this dataset. Note that the quality of extraction is relatively poor due to the dirtiness of the data set. However, the last column of the table shows that the extractor provided a large amount of evidence to the filtering classifier for some properties in the books category, where there was a lot of training data for. (see third column -- # true labels in training data)

3.2 Binary classification

Table 2 shows a comparison of classification quality between the control test and the competitor. The first 200 documents were used for training and the last 400 consecutive documents were used for testing. In this table, the BOOKS category was created as a union of BK_O, and BK_W (descriptions of these labels were given in Figure 3), and the FURNITURE category was created as a union of FUR_O and FUR_W. These categories were created simply as tests as what would happen if similar categories were joined together for training. In the third row, the OTHERS category was a union of all categories except BK_O, BK_W, FUR_O and FUR_W.

The rest of the rows were tests of binary classification of documents as either belonging to the stated category or not. The competitor classification model has the same features as the control model, and in addition indicator feature variables for each of the properties book_price, book_bookstore_used_price, book_title, and book_course_number, which produced extracted spans in the testing set. As indicated in the last column of Table 2, there were no predicted spans of the properties book_bookstore_new_price, fur_item, fur_phonenum and fur_price and hence these were not included as features to the classifier. We see that the competitor outperforms Naïve Bayes with 13% increase in F1 on the 'books for sale' category (BK_O). The reason for this is that the additional features added was most helpful in the books category – upon inspection, taking the first partition of the 3-way cross-validation as example, book_bookstore_used_price is the feature with fourth highest absolute weight among the 5133 features.

As shown in the table, the additional information helped the classification rate in the case of three categories – BK_O, OT and COM_O. Significantly, the OT and COM_O categories are not directly related to the extracted spans that were added as features. Also, while the additional information did not help in every category, it did not hurt the classification rate in any of the other categories.

3.3 Multi-category Classification

Table 3 shows a comparison of multi-category classification error rate between the control test and the competitor, between the 19 categories of the 400 document testing set. The error rate improved slightly. Table 4 shows the confusion matrices of the control and the competitor test. The yellow (light colored) cells along the diagonal show correctly classified documents, while the red (dark colored) cells in the competitor confusion matrix highlight the differences between the results of the competitor classification with respect to the control. The error rate decreased because two more BK_O documents (books offered) were classified correctly compared to the control, and one less COM_O document (computer offered) document was classified correctly.

4. ISSUES AND FUTURE DIRECTION

In this dataset, for each property that is related to a category (for example "books_title"), we have labeled all parts of the document that are related to it. A person actually using a SDI system in the model described earlier may not. The person may simply be filling out relevant information in a form, where mapping the contents of the form to the words in the document which derived it is a machine learning problem in itself.

There is the issue that some documents in this dataset actually deserve more than one category label for example, selling both furniture and computer equipment. They are not many. Based on the 22 category classification, only 7 documents out of 600 should have had more than one label. However the OT (others) category was also very large -- this is partly because a lot of items for sale (such as of ping-pong paddles, or metal detectors, homemade jewelry) simply get classified into the OT category as there is no explicit category that matches it, in addition to general discussion, arguments, test posts and spam.

A major issue is that a user in the model described would probably only give feedback for documents that are useful to them, and only until they found what they wanted – these don't make a complete document category. As a result, it may take very long for the system to become useful for the user and thus, we may want to speed it up by combining user activities on similar tasks for labeling spans. Here we would encounter the issue of dealing with users having varying behavior over similar tasks.

5. CONCLUSION

Motivated by realizing that SDI systems have not taken advantage of many aspects of user behavior, and that user behavior can give a lot of information about why a particular document is interesting, we described a system to use this information. We created models of the user-extracted spans, and used them to identify the word spans in incoming documents that could make the document interesting to the user. Using the simple Naïve Bayes algorithm for classifying documents, we compared filtering quality when the algorithm was used with and without these additional spans as features. Our results show empirically that additional information in this form does help filtering quality. We

conclude that the method we described is potentially useful for SDI systems where the user tasks involve form-filling or similar systematic use of the documents.

Table 4: Confusion matrices of the Control and Competitor tests on multi-category classification, after 3-cross validation.

CONTROL CONFUSION MATRIX

Predicted classes

	APT_0	APT_0W	BK_O	BK_W	COM_O	COM_W	EVENT	EXP_O	FOUND	FUR_O	FUR_W	JOB_O	LOST	OT	RIDE_O	RIDE_W	TIC_O	TUT_O	VEH_O
APT_0	11									1									
APT_W		2												1					
BK_O			62				1	1						1					
BK_W			3											3					
COM_O					55									12					
COM_W					4					1									
EVENT					1		14	1		1				14					
EXP_O								11						4					
FOUND														3					
FUR_O					3	2		2		11				5					
FUR_W	1				1	1		1			1			2					
JOB_O							1					2		1					
LOST							1						2	5					
OT	1		1		5		1			2			1	129					
RIDE_O														1					
RIDE_W														2					
TIC_O														4					
TUT_O														1					
VEH_O					1														2

COMPETITOR CONFUSION MATRIX

Predicted classes

	APT_0	APT_W	BK_O	BK_W	COM_O	COM_W	EVENT	EXP_O	FOUND	FUR_O	FUR_W	JOB_O	LOST	OT	RIDE_O	RIDE_W	TIC_O	TUT_O	VEH_O
APT_0	11									1									
APT_W		2												1					
BK_O			64					1											
BK_W			3											3					
COM_O					54									13					
COM_W					4					1									
EVENT					1		14	1		1				14					
EXP_O								11						4					
FOUND														3					
FUR_O					3	2		2		11				5					
FUR_W					1	1		1			1			2					
JOB_O	1						1					2		1					
LOST							1						2	5					
OT	1		1		5		1			2			1	129					
RIDE_O														1					
RIDE_W														2					
TIC_O														4					
TUT_O														1					
VEH_O					1														2

6. REFERENCES

[A96] Allan, J. Incremental relevance feedback for information filtering, Proceedings of the 19th annual international ACM SIGIR conference on Research and development in information retrieval, 1996.

[ABLMNK02] Anghelescu, A., Boros E., Lewis D., Menkov V., Neu, B., Kantor, P. Rutgers Filtering Work at TREC 2002: Adaptive and Batch. NIST Special Publication SP 500-251, The Eleventh Text Retrieval Conference (TREC 2002)

[AY01] Ault, T. , Yang, Y. kNN, Rocchio and Metrics for Information Filtering at TREC-10. NIST Special Publication SP 500-250, The Tenth Text Retrieval Conference (TREC 2001)

[C] http://www.craigslist.org

[DFFT02] Diao, Y., Fischer, P., Franklin, M. J., and To, R. YFilter: Efficient and Scalable Filtering of XML Documents. Proceedings of the 18th International Conference on Data Engineering (ICDE 2002), page 341, 2002.

[LMP01] Lafferty, J., McCallum, A., Pereira, F. Conditional Random Fields: Probabilistic Models for Segmenting and Labeling Sequence Data. Proceedings of 18th International Conf. on Machine Learning, 2001

[M] http://minorthird.sourceforge.net

[MPM02] McNamee, P., Piatko, C., Mayfield, J. JHU/APL at TREC 2002: Experiments in Filtering and Arabic Retrieval. NIST Special Publication: SP 500-255, The Twelfth Text Retrieval Conference (TREC 2002)

[OC] Ogilvie, P., Callan, J. Combining Structural Information and the Use of Priors in Mixed Named-Page and Homepage Finding. NIST Special Publication: SP 500-255, The Twelfth Text Retrieval Conference (TREC 2003)

[RS01] Robertson, S., and Soboroff, I. The TREC 2001 Filtering Track Report, NIST Special Publication: SP 500-250, The Tenth Text Retrieval Conference (TREC 2001)

[RS02] Robertson, S., and Soboroff, I. The TREC 2002 Filtering Track Report, NIST Special Publication: SP 500-251, The Eleventh Text Retrieval Conference (TREC 2002)

[TCFZKMHMH04] Tomasic, A., Cohen, W., Fussell, S., Zimmerman, J., Kobayashi, M., Minkov, E., Halstead, N., Mosur, R., and Hum, J. Learning to Navigate Web Forms. Proceedings of IIWeb 2004

[TG94] Yan, T., and Garcia-Molina, H. Distributed selective dissemination of information. Proceedings of the Third International Conference on Parallel and Distributed Information Systems, pages 89-98, 1994

[TG99] Yan, T., and Garcia-Molina, H. The SIFT Information Dissemination System. TODS, 24(4), page 529-565, Dec. 1999

[ZXC03] Zhang, Y., Xu, W., and Callan, J. Exploration and Exploitation in adaptive filtering based on Bayesian active learning. Proceedings of the Twentieth International Conference on Machine Learning (ICML-2003)