Temporal Feature Modification for Retrospective Categorization

Robert Liebscher and **Richard K. Belew**
Department of Cognitive Science
University of California, San Diego
{rliebsch|rik}@cogsci.ucsd.edu

Abstract

We show that the intelligent use of one small piece of contextual information–a document's publication date–can improve the performance of classifiers trained on a text categorization task. We focus on academic research documents, where the date of publication undoubtedly has an effect on an author's choice of words. To exploit this contextual feature, we propose the technique of *temporal feature modification*, which takes various sources of lexical change into account, including changes in term frequency, associative strength between terms and categories, and dynamic categorization systems. We present results of classification experiments using both full text papers and abstracts of conference proceedings, showing improved classification accuracy across the whole collection, with performance increases of greater than 40% when temporal features are exploited. The technique is fast, classifier-independent, and works well even when making only a few modifications.

1 Introduction

As they are normally conceived, many tasks relevant to Computational Linguistics (CL), such as text categorization, clustering, and information retrieval, ignore the *context* in which a document was written, focusing instead on the lexical *content* of the document. Numerous improvements have been made in such tasks when context is considered, for example the hyperlink or citation structure of a document collection (Cohn and Hofmann, 2001; Getoor et al., 2001). In this paper, we aim to show that the intelligent use of another dimension of context–a document's publication date–can improve the performance of classifiers trained on a text categorization task.

Traditional publications, such as academic papers and patents, have histories that span centuries. The World Wide Web is no longer a new frontier; over a decade of its contents have been archived (Kahle, 2005); Usenet and other electronic discussion boards have been around for several decades. These forums continue to increase their publication rates and show no signs of slowing. A cursory glance at any one of them at two different points in time can reveal widely varying content.

For a concrete example, we can ask, "What is Computational Linguistics *about*?" Some topics, such as machine translation, lie at the heart of the discipline and will always be of interest. Others are ephemeral or have reached theoretical upper bounds on performance. It is thus more appropriate to ask what CL is about at some point in time. Consider Table 1, which lists the top five unigrams that best distinguished the field at different six-year periods, as derived from the odds ratio measure (see Section 3.2) over the full text of the ACL proceedings.

1979-84	1985-90	1991-96	1997-02
system	phrase	discourse	word
natural	plan	tree	corpus
language	structure	algorithm	training
knowledge	logical	unification	model
database	interpret	plan	data

Table 1: ACL's most characteristic terms for four time periods.

While these changes are interesting in their own right for an historical linguist, we aim to show that they can also be exploited for practical purposes. We focus on a fairly homogeneous set of academic research documents, where the time of publication undoubtedly has an effect both on an author's choice of words and on a field's definition of underlying topical categories. A document must say something novel while building upon what has already been said. This dynamic generates a landscape of changing research language, where authors and disciplines constantly influence and alter the course of one another.

Proceedings of the ACL Workshop on Feature Engineering for Machine Learning in NLP, pages 17–23,
Ann Arbor, June 2005.

1.1 Motivations

Text Categorization (TC) systems are typically used to classify a stream of documents soon after they are produced, based upon a set of historical training data. It is common for some TC applications, such as topic tracking (see Section 5.2), to downweight older features, or the feature vectors of entire documents, while placing more emphasis on features that have recently shown increased importance through changes in frequency and discriminative ability.

Our task, which we call *retrospective categorization*, uses historical data in both the training and test sets. It is retrospective from the viewpoint of a current user browsing through previous writings that are categorized with respect to a "modern" interpretation. Our approach is motivated by three observations concerning lexical change over time, and our task is to modify features so that a text classifier can account for all three.

First, lexical changes can take place within a category. The text collections used in our experiments are from various conference proceedings of the Association of Computing Machinery, which uses a hierarchical classification system consisting of over 500 labels (see Section 2). As was suggested by the example of Table 1, even if classification labels remain constant over time, the terms that best characterize them can change to reflect evolving "meanings". We can expect that many of the terms most closely associated with a category like Computational Linguistics cannot be captured properly without explicitly addressing their temporal context.

Second, lexical changes can occur between categories. A term that is significant to one category can suddenly or gradually become of interest to another category. This is especially applicable in news corpora (see examples in Section 3), but also applies to academic research documents. Terminological "migrations" between topics in computer science, and across all of science, are common.

Third, any coherent document collection on a particular topic is sufficiently dynamic that, over time, its categorization system must be updated to reflect the changes in the world on which its texts are based. Although Computational Linguistics predates Artificial Intelligence (Kay, 2003), many now consider the former a subset of the latter. Within CL, technological and theoretical developments have continually altered the labels ascribed to particular works.

In the ACM's hierarchical Computing Classification System (see Section 2.1), several types of transformations are seen in the updates it received in 1983, 1987, 1991, and 1998.[1] In *bifurcations*, categories can be split apart. With *collapses*, categories that were formerly more fine-grained, but now do not receive much attention, can be combined. Finally, entirely new categories can be inserted into the hierarchy.

[1] http://acm.org/class/

2 Data

To make our experiments tractable and easily repeatable for different parameter combinations, we chose to train and test on two subsets of the ACM corpus. One subset consists of collections of abstracts from several different ACM conferences. The other includes the full text collection of documents from one conference.

2.1 The ACM hierarchy

All classifications were performed with respect to the ACM's Computing Classification System, 1998 version. This, the most recent version of the ACM-CCS, is a hierarchic classification scheme that potentially presents a wide range of hierarchic classification issues. Because the work reported here is focused on temporal aspects of text classification, we have adopted a strategy that effectively "flattens" the hierarchy. We interpret a document which has a *primary*[2] category at a narrow, low level in the hierarchy (e.g., `H.3.3.CLUSTERING`) as also classified at all broader, higher-level categories leading to the root (`H`, `H.3`, `H.3.3`). With this construction, the most refined categories will have fewer example documents, while broader categories will have more.

For each of the corpora considered, a threshold of 50 documents was set to guarantee a sufficient number of instances to train a classifier. Narrower branches of the full ACM-CCS tree were truncated if they contained insufficient numbers of examples, and these documents were associated with their parent nodes. For example, if H.3.3 contained 20 documents and H.3.4 contained 30, these would be "collapsed" into the H.3 category.

All of our corpora carry publication timestamp information involving time scales on the order of one to three decades. The field of computer science, not surprisingly, has been especially fortunate in that most of its publications have been recorded electronically. While obviously skewed relative to scientific and academic publishing more generally, we nevertheless find significant "micro-cultural" variation among the different special interest groups.

2.2 SIGIR full text

We have processed the annual proceedings of the Association for Computing Machinery's Special Interest Group in Information Retrieval (SIGIR) conference from its inception in 1978 to 2002. The collection contains over 1,000 documents, most of which are 6-10 page papers, though some are keynote addresses and 2-3 page poster

[2] Many ACM documents also are classified with additional "other" categories, but these were not used.

Corpus	Vocab size	No. docs	No. cats
SIGIR	16104	520	17
SIGCHI	4524	1910	20
SIGPLAN	6744	3123	22
DAC	6311	2707	20

Table 2: Corpus features

	Unlabeled	Expected
Proceedings	18.97%	7.73%
Periodicals	19.08%	11.54%
No. docs	24,567	8,703

Table 3: Missing classification labels in ACM

summaries. Every document is tagged with its year of publication. Unfortunately, only about half of the SIGIR documents bear category labels. The majority of these omissions fall within the 1978-1987 range, leaving us the remaining 15 years to work with.

2.3 Conference abstracts

We collected nearly 8,000 abstracts from the Special Interest Group in Programming Languages (SIGPLAN), the Special Interest Group in Computer-Human Interaction (SIGCHI) and the Design Automation Conference (DAC). Characteristics of these collections, and of the SIGIR texts, are shown in Table 2.

2.4 Missing labels in ACM

We derive the statistics below from the corpus of all documents published by the ACM between 1960 and 2003. The arguments can be applied to any corpus which has categorized documents, but for which there are classification gaps in the record.

The first column of Table 3 shows that nearly one fifth of all ACM documents, from both conference proceedings and periodicals, do not possess category labels. We define a document's label as "expected" when more than half of the other documents in its publication (one conference proceeding or one issue of a periodical) are labeled, and if there are more than ten total. The second column lists the percentage of documents where we expected a label but did not find one.

3 Methods

Text categorization (TC) is the problem of assigning documents to one or more pre-defined categories. As Section 1 demonstrated, the terms which best characterize a category can change through time, so it is not unreasonable to assume that intelligent use of temporal context will prove useful in TC.

Imagine the example of sorting several decades of articles from the *Los Angeles Times* into the categories `ENTERTAINMENT`, `BUSINESS`, `SPORTS`, `POLITICS`, and `WEATHER`. Suppose we come across the term `schwarzenegger` in a training document. In the 1970s, during his career as a professional bodybuilder, Arnold Schwarzenegger's name would be a strong indicator of a `SPORTS` document. During his film career in the 1980s-1990s, his name would be most likely to appear in an `ENTERTAINMENT` document. After 2003, at the outset of his term as California's governor, the `POLITICS` and `BUSINESS` categories would be the most likely candidates. We refer to `schwarzenegger` as a *temporally perturbed* term, because its distribution across categories varies greatly with time.

Documents containing temporally perturbed terms hold valuable information, but this is lost in a statistical analysis based purely on the average distribution of terms across categories, irrespective of temporal context. This information can be recovered with a technique we call *temporal feature modification* (TFM). We first outline a formal model for its use.

3.1 A term generator framework

One obvious way to introduce temporal information into the categorization task is to simply provide the year of publication as a new lexical feature. Preliminary experiments (not reported here) showed that this method had virtually no effect on classification performance. When the date features were "emphasized" with higher frequencies, classification performance declined.

Instead, we proceed from the perspective of a simplified *language generator* model (e.g. (Blei et al., 2003)). We imagine that the first step in the production of a document involves an author choosing a category `C`. Each term `k` (word, bigram, phrase, etc.) is accorded a unique generator G^k that determines the distribution of `k` across categories, and therefore its likelihood to appear in category `C`. The model assumes that all authors share the same generator for each term, and that the generators do not change over time. We are particularly interested in identifying temporally perturbed lexical generators that violate this assumption.

External events at time *t* can perturb the generator of `k`, causing $\Pr(C|k_t)$ to be different relative to the background $\Pr(C|k)$ computed over the entire corpus. If the perturbation is significant, we want to separate the instances of `k` at time *t* from all other instances.

Returning to our earlier example, we would treat a generic, atemporal occurrence of `schwarzenegger` and the *pseudo-term* "`schwarzenegger`*+2003*" as though they were actually *different* terms, because they were produced by two different generators. We hypothesize that separating the analysis of the two can improve

our estimates of the true Pr(C|k), both in 2003 and in other years.

3.2 TFM Procedure

The generator model motivates a procedure we outline below for flagging certain lexemes with explicit temporal information that distinguish them so as to contrast them with those generated by the underlying atemporal alternatives. This procedure makes use of the (log) odds ratio for feature selection:

$OddsRatio(C,k) = log(\frac{p_k(1-q_k)}{q_k(1-p_k)})$

where p is Pr(k|C), the probability that term k is present, given category C, and q is Pr(k|!C).

The odds ratio between a term and a category is a measure of the associated strength of the two, for it measures the likelihood that a term will occur frequently within a category and (relatively) infrequently outside. Odds ratio happens to perform very well in feature selection tests; see (Mladenic, 1998) for details on its use and variations. Ultimately, it is an arbitrary choice and could be replaced by any method that measures term-category strength.

The following pseudocode describes the process of temporal feature modification:

```
VOCABULARY ADDITIONS:
for each class C:
 for each time (year) t:
  PreModList(C,t,L) = OddsRatio(C,t,L)
  ModifyList(t) =
   DecisionRule(PreModList(C,t,L)
  for each term k in ModifyList(t):
   Add pseudo-term "k+t" to Vocab

DOCUMENT MODIFICATIONS:
for each document:
 t = time (year) of doc
 for each term k:
  if "k+t" in Vocab:
   Replace k with "k+t"
 Classify modified document
```

PreModList(C,*t*,L) is a list of the top L terms that, by the odds ratio measure, are highly associated with category C at time *t*. (In our case, time is divided annually, because this is the finest resolution we have for many of the documents in our corpus.) We test the hypothesis that these come from a perturbed generator at time *t*, as opposed to the atemporal generator G^k, by comparing the odds ratios of term-category pairs in a PreModList at time *t* with the same pairs across the entire corpus. Terms which pass this test are added to the final ModifyList(*t*) for time *t*. For the results that we report, DecisionRule is a simple ratio test with threshold factor *f*. Suppose *f* is 2.0: if the odds ratio between C and k is twice as great at time *t* as it is atemporally, the decision rule is "passed".

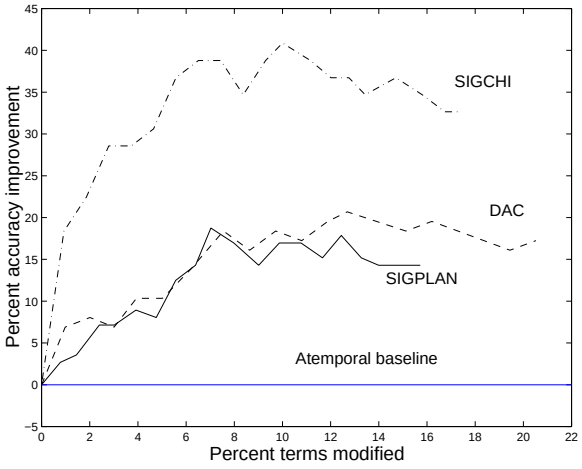

Figure 1: Improvement in categorization performance with TFM, using the best parameter combinations for each corpus.

The generator G^k is then considered perturbed at time *t* and k is added to ModifyList(*t*). In the training and testing phases, the documents are modified so that a term k is replaced with the pseudo-term "k+*t*" if it passed the ratio test.

3.3 Text categorization details

The TC parameters held constant in our experiments are: Stoplist, Porter stemming, and Laplacian smoothing. Other parameters were varied: four different classifiers, three unique minimum vocabulary frequencies, unigrams and bigrams, and four threshold factors *f*. 10-fold cross validation was used for parameter selection, and 10% of the corpus was held out for testing purposes. Both of these sets were distributed evenly across time.

4 Results

Table 4 shows the parameter combinations, chosen by ten-fold cross-validation, that exhibited the greatest increase in categorization performance for each corpus.

Using these parameters, Figure 1 shows the improvement in accuracy for different percentages of terms modified on the test sets. The average accuracies (across all parameter combinations) when no terms are modified are less than stellar, ranging from 26.70% (SIGCHI) to 37.50% (SIGPLAN), due to the difficulty of the task (20-22 similar categories; each document can only belong to one). Our aim here, however, is simply to show improvement. A baseline of 0.0 in the plot indicates accuracy without any temporal modifications.

Figure 2 shows the accuracy on an absolute scale when TFM is applied to the full text SIGIR corpus. Performance increased from the atemporal baseline of 28.85%

Corpus	Improvement	Classifier	n-gram size	Vocab frequency min.	Ratio threshold *f*
SIGIR	33.32%	Naive Bayes	Bigram	2	2.0
SIGCHI	40.82%	TF.IDF	Bigram	10	1.0
SIGPLAN	18.74%	KNN	Unigram	10	1.5
DAC	20.69%	KNN	Unigram	2	1.0

Table 4: Top parameter combinations for TFM by improvement in classification accuracy. *Vocab frequency min.* is the minimum number of times a term must appear in the corpus in order to be included.

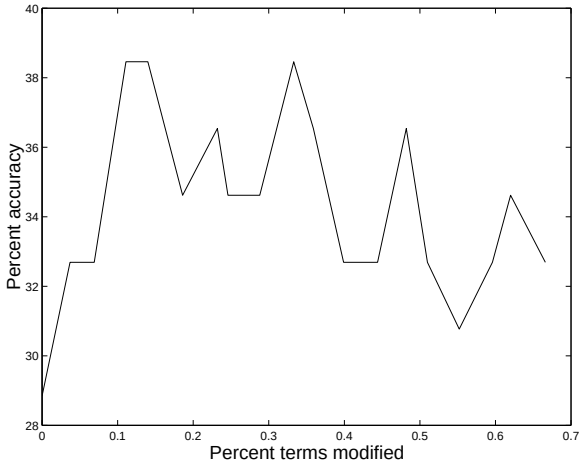

Figure 2: Absolute categorization performance with TFM for the SIGIR full text corpus.

correct to a maximum of 38.46% when only 1.11% of the terms were modified. The *ModifyLists* for each category and year averaged slightly fewer than two terms each.

In most cases, the technique performs best when making relatively few modifications: the left sides of each figure show a rapid performance increase, followed by a gradual decline as more terms are modified. After requiring the one-time computation of odds ratios in the training set for each category/year, TFM is very fast and requires negligible extra storage space. This is important when computing time is at a premium and enormous corpora such as the ACM full text collection are used. It is also useful for quickly testing potential enhancements to the process, some of which are discussed in Section 6.

The results indicate that L in PreModList(C,*t*,L) need not exceed single digits, and that performance asymptotes as the number of terms modified increases. As this happens, more infrequent terms are judged to have been produced by perturbed generators, thus making their true distributions difficult to compute (for the years in which they are not modified) due to an insufficient number of examples.

4.1 General description of results

A quantitative average of all results, using all parameter combinations, is not very meaningful, so we provide a qualitative description of the results not shown in Table 4 and Figures 1 and 2. Of the 96 different parameter combinations tested on four different corpora, 83.33% resulted in overall increases in performance. The greatest increase peaked at 40.82% improvement over baseline (atemporal) accuracy, while the greatest decrease dropped performance by only 8.31%.

5 Related Work

The use of metadata and other complementary (non-content) information to improve text categorization is an interesting and well-known problem. The specific use of temporal information, even if only implicitly, for tasks closely related to TC has been explored through adaptive information filtering (AIF) and topic detection and tracking (TDT).

5.1 Adaptive Information Filtering

There exists a large body of work on information filtering, which "is concerned with the problem of delivering useful information to a user while preventing an overload of irrelevant information" (Lam et al., 1996). Of particular interest here is *adaptive* information filtering (AIF), which handles the problems of concept *drift* (a gradual change in the data set a classifier must learn from) and concept *shift* (a more radical change).

Klinkenberg and Renz test eight different classifiers on their abilities to adapt to changing user preferences for news documents (Klinkenberg and Renz, 1998). They try different "data management techniques" for the concept drift scenario, selectively altering the size of the set of examples (the *adaptive window*) that a classifier trains on using a heuristic that accounts for the degree of dissimilarity between the current batch of examples and previous batches. Klinkenberg and Joachims later abandon this approach because it relies on "complicated heuristics", and instead concentrate their analysis on support vector machines (Klinkenberg and Joachims, 2000).

Stanley uses an innovative approach that eschews the need for an adaptive window of training examples, and

instead relies on a voting system for decision trees (Stanley, 2001). The weight of each classifier's vote (classification) is proportional to its record in predicting classifications for previous examples. He notes that this technique does not rely on decision trees; rather, any combination of classifiers can be inserted into the system.

The concept drift and shift scenarios used in the published literature are often unrealistic and not based upon actual user data. Topic Detection and Tracking, described in the following section, must work not with the behavior of one individual, but with texts that report on real external events and are not subject to artificial manipulation. This multifaceted, unsupervised character of TDT makes it a more appropriate precursor with which to compare our work.

5.2 Topic Detection and Tracking

Franz et al. note that Topic Detection and Tracking (TDT) is fundamentally different from AIF in that the "adaptive filtering task focuses on performance improvements driven by feedback from real-time human relevance assessments. TDT systems, on the other hand, are designed to run autonomously without human feedback" (Franz et al., 2001). Having roots in information retrieval, text categorization, and information filtering, the initial TDT studies used broadcast news transcripts and written news corpora to accomplish tasks ranging from news story clustering to boundary segmentation. Of most relevance to the present work is the *topic tracking* task. In this task, given a small number (1-4) of training stories known to be about a particular event, the system must make a binary decision about whether each story in an incoming stream is about that event.

Many TDT systems make use of temporal information, at least implicitly. Some employ a least recently used (Chen and Ku, 2002) or decay (Allan et al., 2002) function to restrict the lexicon available to the system at any given point in time to those terms most likely to be of use in the topic tracking task.

There are many projects with a foundation in TDT that go beyond the initial tasks and corpora. For example, TDT-inspired language modeling techniques have been used to train a system to make intelligent stock trades based upon temporal analysis of financial texts (Lavrenko et al., 2000). Retrospective *timeline* generation has also become popular, as exhibited by Google's *Zeitgeist* feature and browsers of TDT news corpora (Swan and Allan, 2000; Swan and Jensen, 2000).

The first five years of TDT research are nicely summarized by Allan (Allan, 2002).

6 Summary and Future Work

In this paper, we have demonstrated a feature modification technique that accounts for three kinds of lexical changes in a set of documents with category labels. Within a category, the distribution of terms can change to reflect the changing nature of the category. Terms can also "migrate" between categories. Finally, the categorization system itself can change, leading to necessary lexical changes in the categories that do not find themselves with altered labels. Temporal feature modification (TFM) accounts for these changes and improves performance on the retrospective categorization task as it is applied to subsets of the Association for Computing Machinery's document collection.

While the results presented in this paper indicate that TFM can improve classification accuracy, we would like to demonstrate that its mechanism truly incorporates *changes* in the lexical content of categories, such as those outlined in Section 1.1. A simple baseline comparison would pit TFM against a procedure in which the corpus is divided into slices temporally, and a classifier is trained and tested on each slice individually. Due to changes in community interest in certain topics, and in the structure of the hierarchy, some categories are heavily represented in certain (temporal) parts of the corpus and virtually absent elsewhere. Thus, the chance of finding every category represented in a single year is very low. For our corpora, this did not even occur once.

The "bare bones" version of TFM presented here is intended as a proof-of-concept. Many of the parameters and procedures can be set arbitrarily. For initial feature selection, we used odds ratio because it exhibits good performance in TC (Mladenic, 1998), but it could be replaced by another method such as information gain, mutual information, or simple term/category probabilities. The ratio test is not a very sophisticated way to choose which terms should be modified, and presently only detects the *surges* in the use of a term, while ignoring the (admittedly rare) declines.

In experiments on a Usenet corpus (not reported here) that was more balanced in terms of documents per category and per year, we found that allowing different terms to "compete" for modification was more effective than the egalitarian practice of choosing L terms from each category/year. There is no reason to believe that each category/year is equally likely to contribute temporally perturbed terms.

We would also like to exploit temporal *contiguity*. The present implementation treats time slices as independent entities, which precludes the possibility of discovering temporal trends in the data. One way to incorporate trends *implicitly* is to run a smoothing filter across the temporally aligned frequencies. Also, we treat each slice at annual resolution. Initial tests show that aggregating two or more years into one slice improves performance for some corpora, particularly those with temporally sparse data such as DAC.

Acknowledgements

Many thanks to the anonymous reviewers for their helpful comments and suggestions.

References

J. Allan, V. Lavrenko, and R. Swan. 2002. Explorations within topic tracking and detection. In J. Allan, editor, *Topic Detection and Tracking: Event-based Information Organization*, pages 197–224. Kluwer Academic Publishers.

J. Allan. 2002. Introduction to topic detection and tracking. In J. Allan, editor, *Topic Detection and Tracking: Event-based Information Organization*, pages 1–16. Kluwer Academic Publishers.

David M. Blei, Andrew Y. Ng, and Michael I. Jordan. 2003. Latent dirichlet allocation. *The Journal of Machine Learning Research*, 3:993–1002.

H. Chen and L. Ku. 2002. An nlp & ir approach to topic detection. In J. Allan, editor, *Topic Detection and Tracking: Event-based information organization*, pages 243–264. Kluwer Academic Publishers.

D. Cohn and T. Hofmann. 2001. The missing link: a probabilistic model of document content and hyperlink connectivity. In *Advances in Neural Information Processing Systems*, pages 430–436. MIT Press.

M. Franz, T. Ward, J.S. McCarley, and W. Zhu. 2001. Unsupervised and supervised clustering for topic tracking. In *Proceedings of the Special Interest Group in Information Retrieval*, pages 310–317.

L. Getoor, E. Segal, B. Taskar, and D. Koller. 2001. Probabilistic models of text and link structure for hypertext classification (2001). In *Proceedings of the 2001 IJCAI Workshop on Text Learning: Beyond Supervision*.

Brewster Kahle. 2005. The internet archive. http://www.archive.org/.

Martin Kay. 2003. Introduction. In Ruslan Mitkov, editor, *The Oxford Handbook of Computational Linguistics*, pages xvii–xx. Oxford University Press.

R. Klinkenberg and T. Joachims. 2000. Detecting concept drift with support vector machines. In *Proceedings of the Seventeenth International Conference on Machine Learning (ICML)*, page 11. Morgan Kaufmann.

R. Klinkenberg and I. Renz. 1998. Adaptive information filtering: Learning in the presence of concept drifts. In *AAAI/ICML workshop on learning for text categorization*.

W. Lam, S. Mukhopadhyay, J. Mostafa, and M. Palakal. 1996. Detection of shifts in user interests for personalized information filtering. In *Proceedings of the Special Interest Group in Information Retrieval*, pages 317–326.

V. Lavrenko, M. Schmill, D. Lawrie, P. Ogilvie, D. Jensen, and J. Allan. 2000. Mining of concurrent text and time series. In *6th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Text Mining Workshop*, pages 37–44.

D. Mladenic. 1998. *Machine Learning on non-homogeneous, distributed text data*. Ph.D. thesis, University of Ljubljana, Slovenia.

K.O. Stanley. 2001. Learning concept drift with a committee of decision trees. Computer Science Department, University of Texas-Austin.

R. Swan and J. Allan. 2000. Automatic generation of overview timelines. In *Proceedings of the Special Interest Group in Information Retrieval*, pages 47–55.

R. Swan and D. Jensen. 2000. Timemines: Constructing timelines with statistical models of word usage. In *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*.