Integration of News Content into Web Results

Fernando Diaz
Yahoo! Labs Montreal
1000 Rue de la Gauchetiere
Suite 2400
Montreal, QC H3M4W5
diazf@yahoo-inc.com

ABSTRACT

Aggregated search refers to the integration of content from specialized corpora or *verticals* into web search results. Aggregation improves search when the user has vertical intent but may not be aware of or desire vertical search. In this paper, we address the issue of integrating search results from a news vertical into web search results. News is particularly challenging because, given a query, the appropriate decision—to integrate news content or not—changes with time. Our system adapts to news intent in two ways. First, by inspecting the dynamics of the news collection and query volume, we can track development of and interest in topics. Second, by using click feedback, we can quickly recover from system errors. We define several click-based metrics which allow a system to be monitored and tuned without annotator effort.

Categories and Subject Descriptors

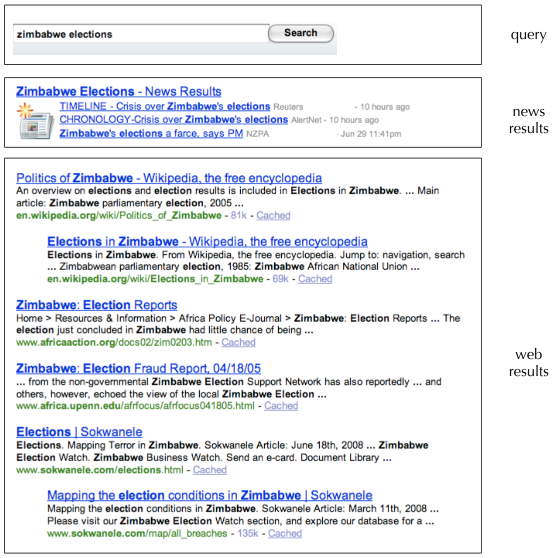
H.3.5 [**Information Storage and Retrieval**]: Online Information Services—*Web-based services*

General Terms

Algorithms

Keywords

Distributed information retrieval, click prediction, news search, query similarity



zimbabwe elections

Zimbabwe Elections - News Results
TIMELINE - Crisis over **Zimbabwe's elections** Reuters - 10 hours ago
CHRONOLOGY-Crisis over **Zimbabwe's elections** AlertNet - 10 hours ago
Zimbabwe's elections a farce, says PM NZPA · Jun 29 11:41pm

Politics of **Zimbabwe** - Wikipedia, the free encyclopedia
An overview on **elections** and **election** results is included in **Elections** in **Zimbabwe**. ... Main article: **Zimbabwe** parliamentary **election**, 2005 ...
en.wikipedia.org/wiki/Politics_of_**Zimbabwe** - 81k - Cached

Elections in **Zimbabwe** - Wikipedia, the free encyclopedia
Elections in **Zimbabwe**. From Wikipedia, the free encyclopedia. Jump to: navigation, search ... Zimbabwean parliamentary **election**, 1985: **Zimbabwe** African National Union ...
en.wikipedia.org/wiki/**Elections**_in_**Zimbabwe** - 69k - Cached

Zimbabwe: **Election** Reports
Home > Resources & Information > Africa Policy E-Journal > **Zimbabwe**: **Election** Reports ... The **election** just concluded in **Zimbabwe** had little chance of being ...
www.**africaaction.org**/docs02/zim0203.htm - Cached

Zimbabwe: **Election** Fraud Report, 04/18/05
... from the non-governmental **Zimbabwe Election** Support Network has also reportedly ... and others, however, echoed the view of the local **Zimbabwe Election** ...
www.**africa.upenn.edu**/afrfocus/afrfocus041805.html - Cached

Elections | Sokwanele
Elections. Mapping Terror in **Zimbabwe**. Sokwanele Article: June 18th, 2008 ... **Zimbabwe Election** Watch. **Zimbabwe** Business Watch. Send an e-card. Document Library ...
www.**sokwanele.com**/**elections**.html - Cached

Mapping the **election** conditions in **Zimbabwe** | Sokwanele
Mapping the **election** conditions in **Zimbabwe**. Sokwanele Article: March 11th, 2008 ... Please visit our **Zimbabwe Election** Watch section, and explore our database for a ...
www.**sokwanele.com**/map/all_breaches - 135k - Cached

Figure 1: Integrating news content into web results.

1. INTRODUCTION

The web is comprised of data covering a variety of media, sources, and topics. The ease of classifying certain content types such as news or images has motivated the construction of specialized sub-collections or *verticals*. Despite the existence of many vertical search engines, searchers may still use a portal search engine even when the query is handled better by a vertical search engine. In these cases, the searcher may express explicit intent for vertical content (e.g. "zimbabwe election news", "border collie pictures"); or, the searcher's intent may be implicit (e.g. "zimbabwe election", "border collie"). When a general web search engine has access to or maintains vertical search engines, one important task becomes the detection and presentation of relevant vertical results. This process is referred to as *aggregated search* [23].

In this paper, we address the issue of integrating search results from a news vertical into web search results. News results are presented above the top web result in a small box we refer to as the *news display* or view. We present an example news display in Figure 1. In response to a news display, a searcher may either *click* on a displayed link to news content or *skip* the display without clicking a displayed link to news content.

News is particularly challenging because, given a query, the appropriate decision—to integrate news content or not—changes with time. Changes occur in two places. In the news index, topics emerge and decay *with respect to content production*. In query logs, news intent emerges and decays *with respect to content demand*. A system which only models evolving topics in the news index may waste modeling effort on topics which searchers never request or, even worse, topics which searchers do not believe are newsworthy. A system which only models evolving query volume will not be able to separate queries requiring news displays from those which are merely popular.

We present a system which integrates both massive document and query approaches to modeling events. Specifically, we will train a classifier to distinguish between newsworthy and non-newsworthy queries.

When training a classifier for any task, one requires a training set. For our task, such a data set would consist of queries manually classified as deserving a news display or not. For humans, making this decision requires knowledge about topical events being queried for as well as topical events being written about *at the time when the query was issued*. While not impossible, such a classification task for a modest number of days would be extremely expensive and potentially unreliable.

To address this, we define several click-based metrics which allow a system to be monitored and tuned without annotator effort. We will demonstrate that this feedback is sufficiently related to manual labels so as to allow it to be used as a surrogate for training.

Our system adapts to the dynamics of the news integration problem. We present modules for (1) assessing the newsworthiness of a query, (2) recovering from system errors, and (3) supporting query similarity.

2. RELATED WORK

In information retrieval, *distributed information retrieval* (DIR) refers to the situation where a searcher's query is satisfied by retrieving content from various sub-collections [7]. DIR can be divided into three subtasks: resource description, resource selection, and results merging. The news integration problem is essentially a DIR task with two subcollections and a fixed results merging scheme (i.e. the news display above the first web result). Our work deviates from prior work in DIR because of non-stationary relevance, massive click feedback, and the small number of subcollections.

Query classification is a common task in information retrieval [4]. Classifiers have been used for the improvement of retrieval [16] and advertising [6]. *Vertical selection* refers to classifying queries into one or more relevant verticals. Li *et al.* present an algorithm for expanding a training set for shopping and job verticals [19]. While Li *et al.* address the general problem of vertical selection, our work focuses on the particularities—for example non-stationarity—of the news vertical.

As mentioned earlier, one approach to detecting events focuses on documents. Topic detection and tracking (TDT) addresses the modeling of bursts of newsworthy documents [1]. TDT systems attempt to predict the onset of news events and topics. In information filtering, given a query or seed document, the system must decide which documents to present the user [5]. Strategies usually involve modeling a topic and presenting documents predicted to be relevant. In some situations, a system intentionally presents a document predicted to be non-relevant in order to refine its topic model [34]; this is part of a more general body of work in active learning [21]. In an information retrieval situation, Diaz and Jones inspected the temporal distribution of retrieved documents in order to predict system performance [11].

Another approach to event detection focuses on queries. Jones and Diaz construct search volume time series and attempt to predict whether a query refers to a single, multiple, or no events [14]. From the data mining community, there is some work on mining bursts in query logs [31]. This is part of a more general body of work in burst detection [33].

Click information provides a noisy feedback on the relevance of documents to a certain query. This information can be used for evaluation [12, 8], offline training [19], and online news personalization [10]. The substantial information in click histories has resulted in the development of several formal models of user click behavior [2, 13].

Because of the redundancy in queries being issued to search engines, a few approaches have applied online learning techniques to adjust models throughout the lifetime of the system. Radlinski *et al.* use a bandit model to adapt retrieval results and evaluate on synthetic click data [26]. Contextual and search advertising refers to deciding which advertisements to place on a webpage or search results page. Bandit approaches have also applied to advertisement placement [24, 25, 17]. These approaches, because placement companies accumulate revenue per-click, optimize the number of clicks accumulated. There are two differences between the ad placement problem and news integration. First, every action in ad placement results in feedback. Whereas if we decide not to present a news display, we receive no feedback. Second, bandit solutions assume reward accumulation micro-averaged over all queries. Instead, we focus on macro-averaged performance, benefiting systems which do well on many queries as opposed to popular queries.

Table 1: Queries binned by empirical click-through rate

bin	breaks	Spring 2007	Winter 2008
1	(0.721,1.000]	8	19
2	(0.553,0.721]	30	40
3	(0.432,0.553]	58	100
4	(0.337,0.432]	104	135
5	(0.260,0.337]	160	212
6	(0.194,0.260]	274	303
7	(0.137,0.194]	458	491
8	(0.086,0.137]	1067	1203
9	(0.041,0.086]	2596	2873
10	[0.000,0.041]	5926	7401
all		10681	12777

3. MOTIVATION

We hypothesize that there is a strong correlation between the click-through rate of a news display and its newsworthiness. We test this hypothesis using two experiments.

We collected two data sets over the course of two weeks in Spring 2007 and two weeks in Winter 2008. Data was collected using a small percentage of search traffic. For users in this sample, we presented a news display if there was *any* hit in the title of articles in the news index. We presented a small box (Figure 1) containing up to three article titles, recording query times, display clicks, and display skips. After the data collection, we first removed query events which were not presented a news display. We then removed queries occurring fewer than 15 times and those consisting of a single character. Queries were normalized by down-casing and removing punctuation. For Spring 2007, this resulted in a total of 1,621,140 views represented as query-time-click triplets (12,777 unique queries). For Winter 2008, this re-

Table 2: Mean newsworthiness grade in each bin. Twenty samples from each bin of Winter 2008 queries were judged to be newsworthy or not. Treating newsworthy queries as having value 1 and 0 otherwise, we computed the mean and standard deviation of grades in each bin. We also present a third column which averages the mean bin values for both users

bin	user 1	user 2	mean
1	0.889 ± 0.314	1.000 ± 0.000	0.944
2	0.722 ± 0.448	0.944 ± 0.229	0.833
3	0.765 ± 0.424	0.944 ± 0.229	0.855
4	0.556 ± 0.497	1.000 ± 0.000	0.778
5	0.600 ± 0.490	0.941 ± 0.235	0.771
6	0.400 ± 0.490	0.706 ± 0.456	0.553
7	0.368 ± 0.482	0.647 ± 0.478	0.508
8	0.111 ± 0.314	0.412 ± 0.492	0.261
9	0.100 ± 0.300	0.222 ± 0.416	0.161
10	0.000 ± 0.000	0.200 ± 0.400	0.100

sulted in 1,856,996 views (10,681 unique queries). For each unique query, we computed the empirical click-through rate using the all view and click information in the data set. We then broke the click-through rate range into the logarithmic subranges presented in Table 1 and counted the number of queries in each bin. Our experiments in this section only use the Winter 2008 data.

We are interested in the click-through rate of queries manually judged newsworthy. To investigate this, we used the following experiment. On a single day, a total of 184,016 displays were presented, receiving a total of 11,742 clicks. During this day, we asked annotators to generate a list of queries which they believed should generate a news display. We gave annotators access to various news media sources including the web as well as query log information. The annotators generated a list of 336 unique queries. For each of these queries, we aggregated the number of clicks and views on this day and computed the click-through rate. The click-through rate was determined to be 0.249. These queries therefore tend to be drawn from the first six bins in Table 1. However, the majority of possible displays is far below this click-through rate.

This result demonstrates that queries manually classified as newsworthy tend to have high click-through rates. We would also like to confirm that queries with high click-through rate tend to be newsworthy. In order to test this, we constructed the following experiment. For the Winter 2008 data, we selected 20 queries from each of the bins in Table 1, forming a set of 199 queries. We repeated this process twice and asked two annotators to label queries as deserving of a news display or not. Assigning the value 1 for newsworthiness and 0 for non-newsworthiness, we computed the mean value for queries in each bin. We present these means and standard deviations in Table 2. We note that, indeed, queries in high click-through rate bins tend to be judged newsworthy by our annotators. Averaging across users, bins with click-through rates above 0.194 tend to be judged newsworthy.

4. OUR APPROACH

In this paper, determining newsworthiness relies on predicting the probability of a user clicks on the news display of a query. We will use the following notation in this paper,

Table 3: Base contextual features. We computed web search query log, news vertical query log, and news index features for each query. We also included various combinations of features to compute rate features

feature	description
`query-last-k`	how many of the last k queries were q
`query-last-k-yesterday`	yesterday, how many of the last k queries were q
`news-last-k`	how many of the last k queries on the news vertical were q
`news-last-k-yesterday`	yesterday, how many of the last k queries on the news vertical were q
`doc-last-k`	how many of the last k documents were retrieved by q
`doc-last-k-yesterday`	yesterday, how many of the last k documents were retrieved by q
`weight-mean-age`	weighting by relevance, how old is the average retrieved document
`weight-stddev-age`	weighting by relevance, what is the standard deviation of retrieved documents

p_q^t	true probability of a user clicking a news display given the query q was issued at time t
$\mathcal{C}_q^t$	clicks observed for query q for views before t
$\mathcal{V}_q^t$	displays presented for query q before t
$\mathcal{S}_q^t$	skips observed for query q (i.e. $\mathcal{V}_q^t - \mathcal{C}_q^t$)
$\tilde{p}_q^t$	predicted probability of a user click
τ	threshold on $\tilde{p}_q^t$ for presenting a display

4.1 Estimation

The simplest way to estimate p_q^t for a news display is by presenting the display to a large number of users over a large period of time. Mathematically,,

$$\tilde{p}_q^t = \frac{\mathcal{C}_q^t}{\mathcal{V}_q^t} \tag{1}$$

Given enough data and assuming p_q^t is stationary, we should be able to compute a decent estimate of p_q^t.

Unfortunately, the assumptions required for a maximum likelihood estimation often do not hold. We often have to estimate p_q^t after only having seen it a few times, with or without direct display click information. Furthermore, p_q^t is unlikely to be stationary. For example, the query "yahoo" will have a higher p_q^t on days when some company-related event happens than on days when nothing happens.

One way to address this situation is to compute an initial guess at the p_q^t from the query's *context*. A query's context refers to the various non-lexicographic information we have about the query when it is submitted. This includes information such as whether the query exists as part of a burst of queries (in either the web or news vertical query log), and whether top ranked news documents retrieved by a query were all published in the last few minutes. We list several candidate predictors in Table 3. We prefer features which are not lexicographic because they allow us to learn a prior belief on the click-through rate based solely on properties of the query, not the query itself.

We would like to model the relationship between these features and clicks. To this end, we can gather data by presenting a large number of displays to users. Because we are using query-independent features, clicks on different queries with the same context will provide meaningful evidence for a model to generalize from. In our experiments, we model the probability of click using logistic regression. We refer to this estimate as π_q^t. It is important to mention that our features change with time, implying that π_q^t may not be equal to π_q^{t+1}.

Recall that Equation 1 suffered from insufficient data for small numbers of views. However, Equation 1 becomes more accurate with large increasing data. We will incorporate π_q^t into our estimate $\tilde{p}_q^t$ using a Beta prior over p_q^t. Mathematically,

$$p_q^t \sim \text{Beta}(a, b) \tag{2}$$

where we set the Beta parameters such that,

$$a = \mu\pi_q^t \qquad b = \mu(1 - \pi_q^t) \tag{3}$$

where μ is a hyperparameter of our model.

Assume we have click and skip information for a query. Then, the posterior, given data from displays presented is also a Beta distribution,

$$p_q^t | \mathcal{C}_q^t, \mathcal{S}_q^t \sim \text{Beta}(a + \mathcal{C}_q^t, b + \mathcal{S}_q^t) \tag{4}$$

And the posterior mean,

$$\begin{aligned} \tilde{p}_q^t &= \int p \frac{\Gamma(a + \mathcal{C}_q^t + b + \mathcal{S}_q^t)}{\Gamma(a + \mathcal{C}_q^t)\Gamma(b + \mathcal{S}_q^t)} p^{a + \mathcal{C}_q^t - 1}(1-p)^{b + \mathcal{S}_q^t - 1} dp \\ &= \frac{\mathcal{C}_q^t + \mu\pi_q^t}{\mathcal{V}_q^t + \mu} \end{aligned} \tag{5}$$

Note here that we can gain an intuition for μ. For small values of μ, the model will be very sensitive to early feedback from the user. For large values, the model will rely on π_q^t more than feedback.

We have demonstrated how to use historic clicks and skips for a query to estimate p_q^t. However, a query's probability of being clicked is also likely to be related to the clicks and skips of topically related queries. For example, consider a situation where we have seen many query events for "olympics opening ceremony" and have presented a direct display for these cases, providing a reliable click-through rate estimate. Then, given the new query "opening ceremony", we have information about the click-through rate if we know that these two queries are related.

There are several methods for detecting the relationship between queries. In this work, we adopt a corpus-based similarity measure using language models of retrieved results. Specifically, we create a query language model by interpolating the top retrieved document language models weighted by their retrieval scores [18, 9]. Formally,

$$P(w|\theta_Q) = \frac{1}{\mathcal{Z}} \sum_{d \in \mathcal{R}} P(w|\theta_d) P(Q|\theta_d) \tag{6}$$

where $P(Q|\theta_d)$ is the query likelihood score for the document d and $\mathcal{Z} = \sum_{d \in \mathcal{R}} P(Q|\theta_d)$. This provides a query language model for each query. We compare two queries by comparing their associated language models using the Bhattacharyya correlation,

$$\mathcal{B}(q_i, q_j) = \sum_{w \in \mathcal{V}} \sqrt{P(w|\theta_{q_i}) P(w|\theta_{q_j})} \tag{7}$$

This provides a similarity score between 0 and 1. Similar methods have been used for retrieval and ad placement tasks [3, 32, 27, 22]. We note that while the Bhattacharyya similarity measure uses only $P(w|\theta_Q)$, other similarity measures based on time or term overlap can be used or combined with $\mathcal{B}$.

We incorporate information from related queries as pseudo-clicks. Specifically, define the aggregated information for a query as,

$$\tilde{\mathcal{C}}_q^t = \mathcal{C}_q^t + \sum_{q'} \mathcal{B}(q, q') \mathcal{C}_{q'}^t \tag{8}$$

$$\tilde{\mathcal{V}}_q^t = \mathcal{V}_q^t + \sum_{q'} \mathcal{B}(q, q') \mathcal{V}_{q'}^t \tag{9}$$

$$\tilde{\mathcal{S}}_q^t = \mathcal{S}_q^t + \sum_{q'} \mathcal{B}(q, q') \mathcal{S}_{q'}^t \tag{10}$$

We can use these modified click and view counts in the same way we used the original counts in Equation 5.

One potential drawback to this approach is maintaining a collection of dense language model vectors (Equation 6) of previously seen queries. Even worse, given a new query, we would have to compute Equation 7 for each of these previously seen queries. In practice, we can avoid much of this cost by using only the top terms from Equation 6 and using inverted indices for storing previously seen queries.

4.2 Classification

Given $\tilde{p}_q^t$, we need to make a decision whether to present a display or not. We will make this decision based on the relative importance of predicting clicks and skips. Assume that we know the true labels for all queries, regardless of whether they were actually presented a display. One way to evaluate our system is by its accuracy of predicting these clicks and skips,

$$\mathcal{A}_\alpha = \frac{\alpha\mathcal{C}_q^+ + \mathcal{S}_q^+}{\alpha\mathcal{C}_q^\star + \mathcal{S}_q^\star} \tag{11}$$

$\mathcal{C}_q^+$	correctly predicted clicks
$\mathcal{S}_q^+$	correctly predicted skips
$\mathcal{C}_q^\star$	total clicks, seen and unseen
$\mathcal{S}_q^\star$	total skips, seen and unseen

where $\alpha \geq 1$ and controls the importance we place on detecting clicks.

Given this measure of accuracy and a value for α, we can compute the expected incremental accuracy for the judgment on one query, $\mathcal{A}_\alpha^q$, by integrating with respect to the distribution in Equation 4,

$$\begin{aligned} \mathbf{E}[\mathcal{A}_\alpha^q | v_q^t] &> \mathbf{E}[\mathcal{A}_\alpha^q | \overline{v}_q^t] \\ \alpha\tilde{p}_q^t &> 1 - \tilde{p}_q^t \\ \tilde{p}_q^t &> \frac{1}{\alpha + 1} \end{aligned} \tag{12}$$

where v_q^t is a binary variable indicating the decision to present a display for query q at time t. Equation 12 provides a method for setting a threshold probability above which a display will be presented; we refer to this value as τ.

Sometimes, we would like to present a display even though a query is below threshold. The motivation for this may be to gather a small amount of valuable feedback without devastating system performance. One naïve method is to present the direct display if the query has not been issued recently. In our work, we will look at presenting the first k occurrences of a query, regardless of $\tilde{p}_q^t$.

The naïve approach samples from all queries in a similar fashion. However, we may wish finer control on the degree of sampling our system performs. Therefore, we can also present queries below threshold randomly with some probability. In the ϵ-greedy approach, if $p_q^t < \tau$, with probability ϵ, we randomly choose to present a display solely in order to get feedback [30].

The ϵ-greedy approach while providing a control on the amount of sampling, does not incorporate any information from a query's click history. We may want to explore only when we have presented few displays. That is, we might make ϵ a function of $\mathcal{V}_q$. To achieve this, we exploit the fact that Equation 4 defines a distribution from which we can sample p_q^t. Assume for some query, $\tilde{p}_q^t < \tau$. We then sample a p_q^t from Equation 4. If this sample is above threshold, we present a display to the user. If we have seen few or no samples, the variance of the posterior will be high, allowing queries with $\tilde{p}_q^t$ below threshold to be displayed. As we accumulate samples, this variance falls, ensuring that queries below threshold will no longer be presented. This process is similar to approaches used in reinforcement learning [28].

4.3 Summary

It is worth pointing out how our system responds to false positives and false negatives. False positives, displayed queries which do not receive clicks, are addressed by the click feedback model. We present an example of the effect of 10 clicks and 10 skips in Figure 2. After 10 skips, the posterior mean falls below threshold (i.e. $\tilde{p}_q^t < \tau$). False negatives, clicked queries which are not presented a display, are addressed by sampling. We present an example of the probability of sampling a below threshold query in Figure 3. With a prior probability of 0.15, the query falls below threshold. However, with probability 0.272, the query will be presented to the searcher.

The hyperparameter μ plays two roles. In Equation 5, μ can be interpreted as pseudo-counts provided by the prior. Therefore, for false positives, a small μ makes the model sensitive to feedback. A larger μ results in a model which is requires more data to change the posterior. For false negatives, a large μ results in a concentrated posterior and a lower probability of queries below threshold being sampled (Figure 3). A small μ results in a higher probability of queries below threshold being sampled. The hyperparameter μ encodes the tradeoff between exploration and exploitation.

We can decouple the exploration and exploitation effects of μ by introducing a new parameter which controls the sensitivity of the model to feedback, independent of exploration. We accomplish this by allowing each observation to contribute a count greater than 1. Assume that each observed click adds a count of w to the $\mathcal{C}_q$; similarly for skips and views. The posterior mean becomes,

$$\tilde{p}_q^t = \frac{w\mathcal{C}_q + \mu\pi_q^t}{w\mathcal{V}_q + \mu} \tag{13}$$

We can, then, for a fixed μ control the contribution of feedback using w.

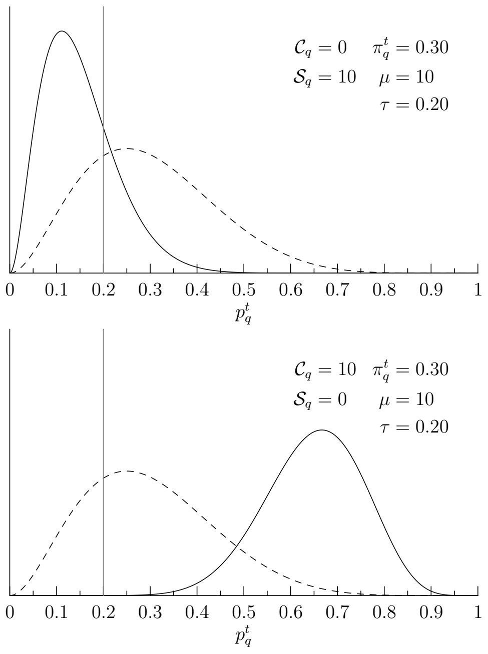


Figure 2: The effect of click information on the posterior. The prior is shown as a dashed line.

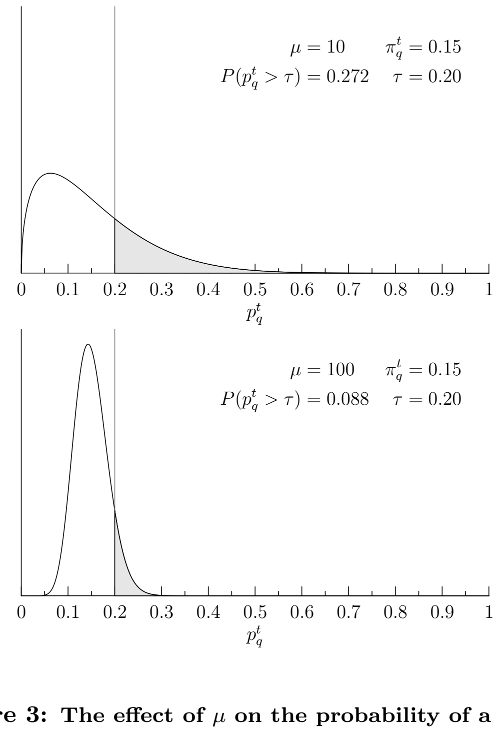


Figure 3: The effect of μ on the probability of a below-threshold query being presented.

5. EVALUATION

We evaluate our performance using two suites of metrics. If we treat clicks as the desired items to be retrieved, we can compute precision and recall for a given threshold τ. Therefore, we sweep values of τ in order to generate precision-recall curves. These curves are micro-averaged; we consider the totality of clicks retrieved as opposed to the clicks retrieved per query. This is because we are likely to compute extremely low precision and recall for the large portion of queries with low click-through rate. Because these queries dominate the data set, subtle differences in precision for queries with high click-through rates is unlikely to be noticeable.

In order to address this issue, we also use the accuracy of a system's classifications (Equation 11). Unlike click precision and recall, we can compute this measure for each query and macro-average the query level accuracies. In our experiments, we set $\alpha = 4$ which leads to a $\tau = 0.20$ and is consistent with the results from our manual annotations. In addition to macro-averaging over all queries, we also macro-average performance within the bins described in Table 1. This will provide us with an indication of which queries are affected by performance changes.

We also define an oracle classifier which computes the stationary, maximum likelihood p_q^t for each query given all data in the data set. We use this oracle classifier in order to provide practical upper bound on performance. We present the performance of this oracle in Table 4. In presentation, we will normalize all scores with respect to this oracle classifier.

Table 4: Accuracy for each bin if we classify each query by its true click-through rate

bin	Spring 2007	Winter 2008
1	0.935	0.947
2	0.870	0.858
3	0.786	0.792
4	0.712	0.710
5	0.625	0.630
6	0.531	0.536
7	0.563	0.562
8	0.671	0.670
9	0.802	0.801
10	0.958	0.962
all	0.855	0.863

6. EXPERIMENTS

We use the two data sets described in Section 3 as ground truth for clicks and skips. We simulate a running system by using these streams of queries with click and skip feedback when the system classifies a query as deserving a display. Because we have the true click and skip information, we can compute the metrics in Section 5.

In addition to query volume information present in the data set, the contextual model needs news collection information. For each set of queries, we gathered three months of news articles found on the web. The Spring 2007 corpus consists of 10,884,958 documents and the Winter 2008 corpus consists of 9,123,972 documents. We indexed the corpora using the indri retrieval engine [29].

We use the `liblinear` package to model the contextual prior [20]. The model trained on the Spring 2007 data was used to compute π_q^t for the Winter 2008 data; similarly, the model trained on the Winter 2008 data was used to compute π_q^t for the Spring 2007 data.

We performed three non-sampling runs which included only using the contextual model (baseline), using feedback without similar queries (history), using feedback with similar queries (similarity). All sample based runs used feedback and similarity information. Results for algorithms which include random sampling were averaged over 100 separate runs. Except for the sampling runs, we fix $\mu = 10$. When using the model with weighted observations, we always use $w = \frac{\mu}{10}$.

7. RESULTS

We present click precision-recall curves for our contextual models (baseline) and feedback (history) in Figure 4. Comparing the baseline performance to that of a model trained on the same data demonstrates that generalizing from Winter 2008 data to Spring 2007 seems to perform better than generalizing in the other direction. The reduced performance appears to be confined to points of low recall. At higher recall levels, models achieve comparable precision. We should also note that across all recall levels, click feedback improves performance over the baseline. In fact, for low recall levels, the feedback model outperforms the contextual model trained on the same set.

Table 5: Normalized accuracy for non-sampling methods. Bold numbers represent statistically significant improvements over the baseline; italicized numbers represent significant decreases compared to the baseline. Numbers superscripted by $\star$ represent statistically significant improvements over history; numbers superscripted by $\circ$ represent statistically significant decreases compared to history

Spring 2007

	baseline	history		similarity	
1	0.297	0.356	19.73 %	0.449	51.25 %
2	0.326	**0.408**	25.22 %	**0.460**	41.18 %
3	0.364	**0.440**	20.87 %	**0.540***	48.40 %
4	0.457	**0.516**	12.85 %	**0.559***	22.19 %
5	0.624	0.624	0.02 %	0.631	1.05 %
6	0.898	0.898	-0.07 %	0.903	0.56 %
7	0.998	0.998	0.03 %	0.996	-0.20 %
8	0.996	**0.998**	0.18 %	0.997°	0.05 %
9	1.000	1.000	0.03 %	1.000°	-0.01 %
10	1.000	1.000	0.01 %	1.000°	-0.02 %
all	0.980	**0.981**	0.16 %	**0.982***	0.26 %

Winter 2008

	baseline	history		similarity	
1	0.358	**0.421**	17.51 %	**0.615***	71.58 %
2	0.576	0.636	10.33 %	**0.820***	42.31 %
3	0.598	**0.680**	13.60 %	**0.776***	29.61 %
4	0.620	**0.654**	5.55 %	**0.738***	19.02 %
5	0.722	0.731	1.36 %	**0.796***	10.32 %
6	0.901	*0.885*	-1.78 %	0.896*	-0.54 %
7	0.978	0.982	0.38 %	0.979	0.10 %
8	0.964	**0.984**	2.08 %	**0.977**$^\circ$	1.32 %
9	0.973	**0.991**	1.94 %	**0.987**$^\circ$	1.46 %
10	0.985	**0.995**	1.00 %	**0.992**$^\circ$	0.68 %
all	0.964	**0.978**	1.35 %	**0.978**	1.37 %

In Table 5, we present the $\mathcal{A}_4$ accuracy of our non-sampling runs. We find that using click feedback improves performance for all query bins, indicating that feedback provides a method not only for dealing with false positives but also,

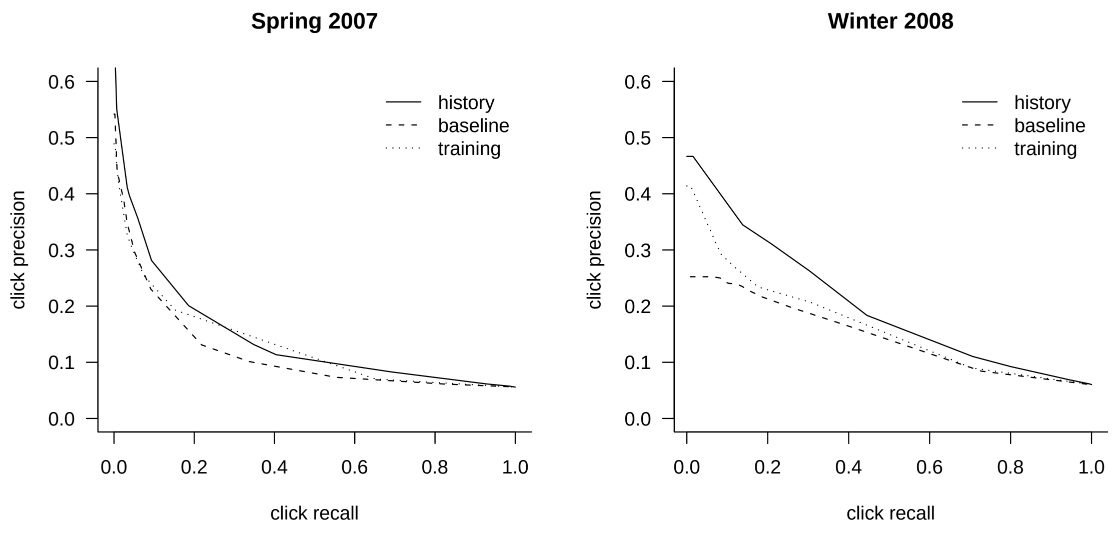


Figure 4: Click precision and recall. The performance of baseline models on the training set is shown as a dotted line.

because π_q^t may change with time, for preventing true positives from becoming false negatives in the future. When we consider similarity information, average performance improves over the basic feedback model in the Spring 2007 data set. However, for both data sets, we see an improvement for queries with high click-through rate and a reduction in queries with low click-through rate. This implies that similarity information tends to classify more queries as newsworthy on average.

We first inspect the performance of our naïve sampling scheme. In Table 6, we compare sampling the first k occurrences of all queries. Since we are increasing the number of queries which are presented displays, an improvement for queries with high click-through rates is expected. However, we also see substantial drops in the accuracy for queries with lower click-through rate.

We introduced the ϵ-greedy approach in order to provide fine-grained control on exploration. Based on the results in Table 7, however, it is unclear whether we see a benefit to this approach. Comparing columns where $\epsilon = 0.25$ to comparably performing columns for the naïve algorithm, we see that the average performance for the ϵ-greedy approach is inferior. For example, in the Winter 2008 data set, performance for bins 1-6 is comparable between $\epsilon = 0.75$ and $k = 1$ but the naïve performs better for queries with lower click-through rates. The attractive property of the ϵ-greedy approach is the ability to control exploration. As ϵ approaches 0, performance will become comparable with "similarity". This granularity is not possible with naïve sampling.

One of the drawbacks of the ϵ-greedy approach is in its uniform treatment of below threshold queries. In Table 8, we present the performance for sampling from the posterior. For $\mu = 10$, we observe a greater improvement in queries with high click-through rate and smaller reduction in queries with low click-through rate compared to both of the other sampling methods. For example, in the Spring 2007 data set, sampling from the posterior achieves statistically significant improvements compared to naïve sampling ($k = 1$) for bins 3, 4, 8, 9, and 10 with no significant differences in performance for other bins. Similarly, for the same data set, sampling from the posterior achieves statistically significant improvements compared to ϵ-greedy sampling ($\epsilon = 0.05$) for bins 7, 8, 9, and 10 with no significant differences in performance for other bins.

If we attempt to reduce exploration by increasing μ, we begin to degrade performance on queries with high click-through rate. This is results from the coupled effects of μ. Reducing exploration results in more concentrated posteriors which in turn require more data to be adjusted. In Table 9, we present results for our model with weighted observations. In this case, increasing μ for a fixed w results in a more graceful change in performance. In fact, at $\mu = 100$, we get significant improvements in performance for queries with high click-through rate without hurting average performance.

8. CONCLUSION

We have presented a system for aggregating news search and web search which is able to adapt to new events in response to document volume, query volume, and click feedback. We found that click feedback could provide evidence to adaptively improve a non-lexicographic baseline model. We also demonstrated a method for incorporating query similarity to improve performance on queries with high click-through rate. Finally, we proposed several algorithms for opportunistically gathering user feedback for low-performing queries.

We believe the approaches described here are general enough to apply to other verticals with dynamic content and interest such as blog or financial data. Even within the context of news integration, there is room for future work. First, we assume that non-stationarity is dealt with in our model through the contextual model features and click feedback. Alternatively, we can use signals in the traffic to detect when it is appropriate to significantly revise our estimate of p_q^t [15].

Secondly, we incorporated information from similar queries through click evidence. Alternatively, we might incorporate feature values from similar queries.

9. ACKNOWLEDGMENTS

Rajesh Parekh, Jignashu Parikh, and Pavel Berkhin originally proposed a news vertical selection algorithm and influenced the features used in this paper; patent under review ("Predicting Newsworthy Queries Using Combined Online and Offline Models", Attorney Docket No. YAH1P140, Y04164US00). Jaime Arguello, Hugues Bouchard, Jean-François Crespo, Remi Kwan, Frode Lundgren, Donald Metzler, and Desislava Petkova provided useful discussions and support.

10. REFERENCES

[1] J. Allan, editor. *Topic Detection and Tracking: Event-based Information Organization*, volume 12 of *The Information Retrieval Series*. Springer, 2002.

[2] H. Becker, C. Meek, and D. M. Chickering. Modeling contextual factors of click rates. In *AAAI*, pages 1310–1315, 2007.

[3] D. Beeferman and A. Berger. Agglomerative clustering of a search engine query log. In *KDD 2000*, pages 407–416, 2000.

[4] S. M. Beitzel, E. C. Jensen, O. Frieder, D. Grossman, D. D. Lewis, A. Chowdhury, and A. Kolcz. Automatic web query classification using labeled and unlabeled training data. In *SIGIR 2005*, pages 581–582, 2005.

[5] N. J. Belkin and W. B. Croft. Information filtering and information retrieval: two sides of the same coin? *CACM*, 35(12):29–38, 1992.

[6] A. Z. Broder, M. Fontoura, E. Gabrilovich, A. Joshi, V. Josifovski, and T. Zhang. Robust classification of rare queries using web knowledge. In *SIGIR 2007*, pages 231–238, 2007.

[7] J. Callan. Distributed information retrieval. In W. B. Croft, editor, *Advances in Information Retrieval*. Kluwer Academic Publishers, 2000.

[8] B. Carterette and R. Jones. Evaluating search engines by modeling the relationship between relevance and clicks. In *NIPS*, 2007.

[9] S. Cronen-Townsend, Y. Zhou, and W. B. Croft. Predicting query performance. In *SIGIR 2002*, pages 299–306, 2002.

[10] A. S. Das, M. Datar, A. Garg, and S. Rajaram. Google news personalization: scalable online collaborative filtering. In *WWW 2007*, pages 271–280, 2007.

[11] F. Diaz and R. Jones. Using temporal profiles of queries for precision prediction. In *SIGIR 2004*, pages 18–24, 2004.

[12] G. Dupret, V. Murdock, and B. Piwowarski. Web search engine evaluation using clickthrough data and a user model. In *Query Log Analysis: Social And Technological Challenges. A workshop at the 16th International World Wide Web Conference (WWW 2007)*, May 2007.

[13] G. E. Dupret and B. Piwowarski. A user browsing model to predict search engine click data from past observations. In *SIGIR 2008*, pages 331–338, 2008.

[14] R. Jones and F. Diaz. Temporal profiles of queries. *ACM Trans. Inf. Syst.*, 25(3):14, July 2007.

[15] N. K. Jong and P. Stone. Bayesian models of nonstationary markov decision processes. In *The IJCAI-2005 Workshop on Planning and Learning in A Priori Unknown or Dynamic Domains*, 2005.

[16] I.-H. Kang and G. Kim. Query type classification for web document retrieval. In *SIGIR 2003*, pages 64–71, 2003.

[17] R. Kleinberg, A. Slivkins, and E. Upfal. Multi-armed bandits in metric spaces. In *STOC 2008*. 2008.

[18] V. Lavrenko and W. B. Croft. Relevance based language models. In *SIGIR 2001*, pages 120–127, 2001.

[19] X. Li, Y.-Y. Wang, and A. Acero. Learning query intent from regularized click graphs. In *SIGIR 2008*, pages 339–346, 2008.

[20] C.-J. Lin, R. C. Weng, and S. S. Keerthi. Trust region newton method for large-scale logistic regression. *Journal of Machine Learning Research*, 9:627–650, 2008.

[21] A. K. McCallum and K. Nigam. Employing EM in pool-based active learning for text classification. In *ICML 1998*, pages 350–358, 1998.

[22] D. Metzler, S. T. Dumais, and C. Meek. Similarity measures for short segments of text. In *ECIR*, pages 16–27, 2007.

[23] V. Murdock and M. Lalmas, editors. *Proceedings of the SIGIR Workshop on Aggregated Search*, 2008.

[24] S. Pandey, D. Agarwal, D. Chakrabarti, and V. Josifovski. Bandits for taxonomies: A model-based approach. In *SDM*, 2007.

[25] S. Pandey, D. Chakrabarti, and D. Agarwal. Multi-armed bandit problems with dependent arms. In *ICML 2007*, pages 721–728, 2007.

[26] F. Radlinski, R. Kleinberg, and T. Joachims. Learning diverse rankings with multi-armed bandits. In *ICML 2008*, pages 784–791, 2008.

[27] M. Sahami and T. D. Heilman. A web-based kernel function for measuring the similarity of short text snippets. In *WWW 2006*, pages 377–386, 2006.

[28] M. J. A. Strens. A bayesian framework for reinforcement learning. In *ICML 2000*, pages 943–950, 2000.

[29] T. Strohman, D. Metzler, H. Turtle, and W. B. Croft. Indri: A language model-based search engine for complex queries. In *Proceedings of the International Conference on Intelligence Analysis*, 2004.

[30] R. Sutton and A. Barto. *Reinforcement Learning*. MIT Press, 1998.

[31] M. Vlachos, C. Meek, Z. Vagena, and D. Gunopulos. Identifying similarities, periodicities and bursts for online search queries. In *SIGMOD 2004*, pages 131–142, 2004.

[32] J.-R. Wen, J.-Y. Nie, and H.-J. Zhang. Query clustering using user logs. *ACM Trans. Inf. Syst.*, 20(1):59–81, 2002.

[33] X. Zhang. *Fast Algorithms for Burst Detection*. PhD thesis, New York University, 2006.

[34] Y. Zhang, W. Xu, and J. P. Callan. Exploration and exploitation in adaptive filtering based on bayesian active learning. In *ICML 2003*, pages 896–903, 2003.

Table 6: Naïve sampling of queries. The parameter k controls the number of queries we automatically present before using the model

Spring 2007

	similarity	$k = 1$		$k = 2$		$k = 3$		$k = 4$		$k = 5$	
1	0.449	0.545	21.46 %	**0.780**	73.76 %	**1.004**	123.53 %	**1.003**	123.24 %	**1.001**	122.95 %
2	0.460	**0.602**	30.67 %	**0.736**	59.88 %	**0.837**	81.75 %	**0.872**	89.35 %	**0.955**	107.38 %
3	0.540	**0.653**	20.93 %	**0.743**	37.48 %	**0.794**	46.99 %	**0.875**	61.99 %	**0.906**	67.76 %
4	0.559	**0.627**	12.19 %	**0.699**	25.11 %	**0.768**	37.44 %	**0.788**	41.01 %	**0.819**	46.60 %
5	0.631	**0.704**	11.66 %	**0.757**	19.98 %	**0.797**	26.39 %	**0.822**	30.29 %	**0.856**	35.66 %
6	0.903	0.915	1.29 %	**0.928**	2.73 %	**0.929**	2.88 %	**0.930**	2.92 %	**0.929**	2.82 %
7	0.996	*0.982*	-1.40 %	*0.958*	-3.80 %	*0.938*	-5.81 %	*0.925*	-7.07 %	*0.923*	-7.28 %
8	0.997	*0.953*	-4.39 %	*0.912*	-8.53 %	*0.865*	-13.22 %	*0.833*	-16.39 %	*0.798*	-19.96 %
9	1.000	*0.934*	-6.52 %	*0.873*	-12.63 %	*0.815*	-18.45 %	*0.759*	-24.02 %	*0.708*	-29.12 %
10	1.000	*0.910*	-8.96 %	*0.828*	-17.21 %	*0.751*	-24.91 %	*0.679*	-32.02 %	*0.615*	-38.52 %
all	0.982	*0.915*	-6.85 %	*0.852*	-13.25 %	*0.792*	-19.40 %	*0.736*	-25.05 %	*0.685*	-30.23 %

Winter 2008

	similarity	$k = 1$		$k = 2$		$k = 3$		$k = 4$		$k = 5$	
1	0.615	**0.840**	36.66 %	**0.966**	57.14 %	**0.991**	61.26 %	**1.000**	62.68 %	**1.000**	62.68 %
2	0.820	**0.902**	10.00 %	**0.934**	13.90 %	**0.951**	15.95 %	**0.957**	16.60 %	**0.966**	17.80 %
3	0.776	**0.849**	9.49 %	**0.877**	13.12 %	**0.891**	14.87 %	**0.923**	18.98 %	**0.917**	18.26 %
4	0.738	**0.793**	7.53 %	**0.850**	15.26 %	**0.893**	21.01 %	**0.912**	23.61 %	**0.925**	25.42 %
5	0.796	**0.832**	4.54 %	**0.857**	7.62 %	**0.889**	11.62 %	**0.901**	13.19 %	**0.903**	13.39 %
6	0.896	**0.918**	2.45 %	**0.926**	3.30 %	**0.929**	3.58 %	**0.934**	4.21 %	**0.934**	4.19 %
7	0.979	*0.958*	-2.12 %	*0.935*	-4.50 %	*0.929*	-5.09 %	*0.919*	-6.12 %	*0.904*	-7.61 %
8	0.977	*0.930*	-4.81 %	*0.889*	-9.01 %	*0.857*	-12.28 %	*0.823*	-15.73 %	*0.788*	-19.38 %
9	0.987	*0.922*	-6.61 %	*0.861*	-12.74 %	*0.804*	-18.53 %	*0.754*	-23.60 %	*0.708*	-28.23 %
10	0.992	*0.907*	-8.56 %	*0.828*	-16.54 %	*0.753*	-24.06 %	*0.683*	-31.08 %	*0.620*	-37.50 %
all	0.978	*0.911*	-6.77 %	*0.849*	-13.16 %	*0.791*	-19.09 %	*0.737*	-24.65 %	*0.686*	-29.86 %

Table 7: ϵ-greedy. The parameter ϵ is equal to the probability of showing a display if a query is below threshold. Performance is averaged across 100 runs of the algorithm

Spring 2007

	similarity	$\epsilon = 0.25$		$\epsilon = 0.15$		$\epsilon = 0.05$	
1	0.449	**0.806**	79.55 %	**0.719**	60.17 %	**0.557**	24.08 %
2	0.460	**0.777**	68.72 %	**0.702**	52.45 %	**0.570**	23.73 %
3	0.540	**0.785**	45.28 %	**0.729**	35.01 %	**0.630**	16.66 %
4	0.559	**0.768**	37.44 %	**0.715**	27.98 %	**0.624**	11.68 %
5	0.631	**0.781**	23.88 %	**0.744**	17.95 %	**0.684**	8.36 %
6	0.903	**0.940**	4.10 %	**0.929**	2.81 %	**0.913**	1.12 %
7	0.996	*0.936*	-6.05 %	*0.958*	-3.81 %	*0.983*	-1.28 %
8	0.997	*0.868*	-12.90 %	*0.919*	-7.82 %	*0.970*	-2.70 %
9	1.000	*0.811*	-18.87 %	*0.886*	-11.37 %	*0.961*	-3.81 %
10	1.000	*0.761*	-23.85 %	*0.856*	-14.32 %	*0.952*	-4.78 %
all	0.982	*0.797*	-18.91 %	*0.872*	-11.26 %	*0.946*	-3.71 %

Winter 2008

	similarity	$\epsilon = 0.25$		$\epsilon = 0.15$		$\epsilon = 0.05$	
1	0.615	**0.873**	41.96 %	**0.822**	33.76 %	**0.708**	15.13 %
2	0.820	**0.910**	10.90 %	**0.887**	8.17 %	**0.851**	3.76 %
3	0.776	**0.888**	14.56 %	**0.861**	10.98 %	**0.817**	5.29 %
4	0.738	**0.858**	16.33 %	**0.834**	13.05 %	**0.794**	7.61 %
5	0.796	**0.873**	9.65 %	**0.853**	7.18 %	**0.825**	3.60 %
6	0.896	**0.926**	3.35 %	**0.918**	2.41 %	**0.907**	1.21 %
7	0.979	*0.922*	-5.78 %	*0.942*	-3.80 %	*0.964*	-1.50 %
8	0.977	*0.853*	-12.75 %	*0.901*	-7.84 %	*0.950*	-2.82 %
9	0.987	*0.802*	-18.71 %	*0.875*	-11.30 %	*0.949*	-3.86 %
10	0.992	*0.756*	-23.78 %	*0.850*	-14.29 %	*0.944*	-4.80 %
all	0.978	*0.790*	-19.15 %	*0.866*	-11.47 %	*0.940*	-3.83 %

Table 8: Sampling from the posterior. The parameter μ controls the concentration of the posterior around the π_q^t. Performance is averaged across 100 runs of the algorithm

Spring 2007

	similarity	$\mu = 10$		$\mu = 25$		$\mu = 50$		$\mu = 100$	
1	0.449	**0.639**	42.38 %	0.486	8.29 %	0.454	1.10 %	0.387	-13.81 %
2	0.460	**0.671**	45.74 %	**0.510**	10.84 %	0.445	-3.36 %	*0.410*	-11.04 %
3	0.540	**0.714**	32.14 %	**0.585**	8.25 %	*0.494*	-8.60 %	*0.439*	-18.79 %
4	0.559	**0.679**	21.62 %	**0.581**	3.99 %	*0.528*	-5.50 %	*0.504*	-9.85 %
5	0.631	**0.719**	13.94 %	**0.661**	4.74 %	0.641	1.65 %	0.633	0.41 %
6	0.903	**0.924**	2.33 %	**0.910**	0.78 %	0.906	0.25 %	0.903	-0.03 %
7	0.996	*0.980*	-1.55 %	0.994	-0.16 %	0.997	0.08 %	0.997	0.14 %
8	0.997	*0.976*	-2.08 %	*0.992*	-0.50 %	*0.995*	-0.19 %	0.995	-0.12 %
9	1.000	*0.979*	-2.05 %	*0.996*	-0.31 %	0.999	-0.02 %	1.000	0.01 %
10	1.000	*0.979*	-2.10 %	*0.996*	-0.32 %	0.999	-0.02 %	1.000	0.01 %
all	0.982	*0.968*	-1.49 %	*0.981*	-0.19 %	*0.982*	-0.07 %	*0.981*	-0.13 %

Winter 2008

	similarity	$\mu = 10$		$\mu = 25$		$\mu = 50$		$\mu = 100$	
1	0.615	**0.863**	40.46 %	**0.721**	17.25 %	0.656	6.67 %	0.611	-0.61 %
2	0.820	**0.914**	11.48 %	**0.862**	5.07 %	0.798	-2.72 %	*0.726*	-11.53 %
3	0.776	**0.886**	14.28 %	**0.838**	8.08 %	0.778	0.34 %	*0.722*	-6.88 %
4	0.738	**0.850**	15.21 %	**0.812**	10.05 %	0.763	3.42 %	0.713	-3.39 %
5	0.796	**0.870**	9.30 %	**0.846**	6.22 %	**0.815**	2.34 %	0.785	-1.37 %
6	0.896	**0.918**	2.44 %	**0.916**	2.14 %	**0.915**	2.11 %	**0.910**	1.48 %
7	0.979	*0.937*	-4.24 %	*0.956*	-2.34 %	*0.967*	-1.19 %	0.974	-0.54 %
8	0.977	*0.908*	-7.06 %	*0.931*	-4.69 %	*0.947*	-3.07 %	*0.957*	-2.02 %
9	0.987	*0.907*	-8.05 %	*0.935*	-5.21 %	*0.955*	-3.22 %	*0.966*	-2.13 %
10	0.992	*0.908*	-8.43 %	*0.940*	-5.25 %	*0.963*	-2.92 %	*0.976*	-1.62 %
all	0.978	*0.908*	-7.15 %	*0.934*	-4.50 %	*0.952*	-2.67 %	*0.961*	-1.73 %

Table 9: Sampling from the posterior with weighted observations. The parameter μ controls the concentration of the posterior around the $\tilde{p}_q^t$. Performance is averaged across 100 runs of the algorithm

Spring 2007

	similarity	$\mu = 10$		$\mu = 25$		$\mu = 50$		$\mu = 100$	
1	0.449	**0.639**	42.38 %	**0.547**	21.77 %	0.487	8.50 %	0.476	5.90 %
2	0.460	**0.671**	45.74 %	**0.580**	26.04 %	**0.512**	11.16 %	**0.478**	3.91 %
3	0.540	**0.714**	32.14 %	**0.651**	20.60 %	**0.603**	11.72 %	**0.573**	6.12 %
4	0.559	**0.679**	21.62 %	**0.619**	10.86 %	**0.582**	4.18 %	0.567	1.53 %
5	0.631	**0.719**	13.94 %	**0.683**	8.29 %	**0.660**	4.58 %	**0.646**	2.43 %
6	0.903	**0.924**	2.33 %	**0.914**	1.22 %	0.909	0.60 %	0.906	0.33 %
7	0.996	*0.980*	-1.55 %	*0.990*	-0.61 %	*0.993*	-0.27 %	0.995	-0.11 %
8	0.997	*0.976*	-2.08 %	*0.990*	-0.68 %	*0.994*	-0.27 %	*0.995*	-0.15 %
9	1.000	*0.979*	-2.05 %	*0.995*	-0.47 %	*0.998*	-0.13 %	*0.999*	-0.06 %
10	1.000	*0.979*	-2.10 %	*0.995*	-0.41 %	*0.999*	-0.07 %	*0.999*	-0.03 %
all	0.982	*0.968*	-1.49 %	*0.981*	-0.17 %	0.983	0.02 %	0.983	0.01 %

Winter 2008

	similarity	$\mu = 10$		$\mu = 25$		$\mu = 50$		$\mu = 100$	
1	0.615	**0.863**	40.46 %	**0.808**	31.47 %	**0.748**	21.64 %	**0.700**	13.87 %
2	0.820	**0.914**	11.48 %	**0.899**	9.61 %	**0.887**	8.18 %	**0.873**	6.45 %
3	0.776	**0.886**	14.28 %	**0.863**	11.28 %	**0.841**	8.46 %	**0.817**	5.34 %
4	0.738	**0.850**	15.21 %	**0.834**	13.07 %	**0.816**	10.57 %	**0.795**	7.77 %
5	0.796	**0.870**	9.30 %	**0.858**	7.77 %	**0.847**	6.38 %	**0.834**	4.73 %
6	0.896	**0.918**	2.44 %	**0.914**	2.01 %	**0.910**	1.56 %	**0.906**	1.03 %
7	0.979	*0.937*	-4.24 %	*0.951*	-2.82 %	*0.962*	-1.77 %	*0.968*	-1.08 %
8	0.977	*0.908*	-7.06 %	*0.933*	-4.51 %	*0.950*	-2.82 %	*0.961*	-1.61 %
9	0.987	*0.907*	-8.05 %	*0.941*	-4.67 %	*0.961*	-2.60 %	*0.974*	-1.32 %
10	0.992	*0.908*	-8.43 %	*0.946*	-4.56 %	*0.968*	-2.36 %	*0.981*	-1.10 %
all	0.978	*0.908*	-7.15 %	*0.940*	-3.89 %	*0.958*	-2.01 %	*0.969*	-0.93 %