# Improving Web Page Classification by Label-propagation over Click Graphs

Soo-Min Kim, Patrick Pantel, Lei Duan, and Scott Gaffney
Yahoo! Labs
701 First Avenue,
Sunnyvale, CA, 94089-0703, US
{soomin, ppantel, leiduan, gaffney}@yahoo-inc.com

## ABSTRACT
In this paper, we present a semi-supervised learning method for web page classification, leveraging click logs to augment training data by propagating class labels to unlabeled similar documents. Current state-of-the-art classifiers are supervised and require large amounts of manually labeled data. We hypothesize that unlabeled documents similar to our positive and negative labeled documents tend to be clicked through by the same user queries. Our proposed method leverages this hypothesis and augments our training set by modeling the similarity between documents in a click graph. We experiment with three different web page classifiers and show empirical evidence that our proposed approach outperforms state-of-the-art methods and reduces the amount of human effort to label training data.

## Categories and Subject Descriptors
H.3.3 [**INFORMATION STORAGE AND RETRIEVAL**]: Information Search and Retrieval – *Relevance feedback, Selection process*

## General Terms
Algorithms, Measurement, Performance, Design, Experimentation

## Keywords
Web page classification, click-through data, page similarity, seed set expansion.

## 1. INTRODUCTION
Web page classification is one of the basic tasks for understanding a web page. It categorizes a web page into various classes, such as news, blog, shopping, adult and review, based on the contents of the page and link information such as in-link (i.e., which pages point to this page) and out-link (i.e., urls to which this page points). Most page classification systems follow supervised learning methods, which rely heavily on large amounts of manually annotated data. The obvious downside of this approach is the large labor cost of annotating the data.


CIKM'09, November 2–6, 2009, Hong Kong, China.


Web page classification is different from typical text classification. Unlike other texts, web pages contain not only textual content, but also html tags, meta-data, images, and in/out-link urls. Web page classifiers which use this additional information outperform classifiers which rely solely on textual features. On the other hand, the textual content of web pages contains not only primary content, but also peripheral content such as advertisement, bread crumbs, and navigation information. Peripheral content tends to confuse a classifier and discerning between it and primary content is challenging. A common way to address this problem is to have very large labeled training data which mask the features of peripheral content.

A hypothesis of this paper is that search engine click logs contain information which links web pages of similar classes. Click data contains information such as which queries search engine users issue to retrieve certain pages and how many times those pages are actually clicked from those queries. Although it is well known that click data suffers from position bias (Craswell, 2008) (e.g., users are likely to click urls presented in first position), the aggregated click data across many users over time provides valuable information. Since click data is relatively inexpensive to obtain, it has been applied to web applications such as relevance ranking and query expansion.

In this paper, we apply click data in web page classification to augment training data by propagating class labels to unlabeled similar documents. Although click data can also be utilized as features of a page classification model, our focus is to expand training data automatically by finding similar pages from user click information. We hypothesize that unlabeled documents similar to our positive and negative labeled documents will tend to be clicked on by the same user queries. We then augment our training set (i.e., seed set) by modeling the similarity between documents in a click graph. Similarity between two pages is computed based on the number of clicks and the queries used to click two pages[1]. An alternative method to identify similar pages would be investigating page content such as text, in/out-link and title. However, in order to obtain such content it requires crawling actual pages and store the content of millions of document. Our goal is to identify similar pages without crawling contents of the web.

[1] Other click data information such as dwell time (i.e., how long users spent on a page) is not considered in our work.

We propose two models for seed set augmentation: *click data model (CDM)* and *query constraint model* (*QC model*). In both models, each url (i.e., page) is represented as a vector of queries that users issued and clicked through to the page. The page similarity between two pages is computed using the cosine similarity between the two vectors. In *CDM*, we augment seed sets based on the cosine similarity scores. The intuition is that a page with high similarity with the seed set should have the same class label as the seed set. In the *QC* model, we consider query patterns as well as similarity scores. It augments the *CDM* model by enforcing that certain query terms are shared by the page and the seed set (e.g., for a review classifier, we would enforce that a page shares with the seed set a query containing words such as *review*).

We also propose an active learning method leveraging click data, which attempts to reduce the number of labeled data required to build a model of similar performance by intelligently (i.e., actively) collecting training data. In our approach, we first label positive and negative seed sets then expand them using the proposed expansions from the *CDM* and *QC* models. Then, we manually label the augmented sets and expand them again iteratively. A model trained with data sets collected by this approach outperforms a model built with three times more labeled data.

We investigate our hypothesis by applying the approach in three different web page classification tasks. Experimental results show promising result and support our claim that we can use much less human annotation effort by leveraging click data.

This paper is organized as following: Section 2 summarizes previous work related to our research and Section 3 describes our proposed approach of calculating page similarity using click data. Section 4 proposes two algorithms for automatically expanding seed sets of labeled urls, using the similarity graph built in Section 3, and proposes our active learning algorithm for obtaining labeled pages. Section 5 reports experimental results of our methods with various classification problems including adult, review, and howto. Finally, Section 6 discusses practical issues and future work and Section 7 concludes this paper.

# 2. RELATED WORK

This paper is closely related to web page classification. Since the main different idea of this work from previous page classification work is leveraging click through data, this section also summarizes previous research on systems and algorithms using clickthrough data.

Text classification is a task of assigning a class label from pre-defined classes to a given document. Binary classification has two pre-defined classes whereas multi-class classification has more than two classes. Text classification tasks typically use bag-of-word approaches (Joachisms, 2001). However, (Chakrabarti et al., 1998) discuss that hypertext categorization is a lot more difficult than text classification. They present that a text classification system without considering link structure of hypertexts performs poorly on hypertext classification task. (Yang et al., 2002) also address the challenge of hypertext categorization and present methodologies using meta-data and hyperlink features of hypertext.

User click data has been applied in many web applications such as web search ranking systems and query classification. (Joachims, 2002) introduced the use of click data for search engine optimization. (Radlinski et al, 2008) investigate the co-relation between click data and search quality and (Dou et al., 2008) leverage click data for a search ranking system by extracting pair-wise relevance preferences. (Chapelle and Zhang, 2009) discuss the search-engine position bias problem of click data (i.e., top ranked result will get more user clicks) and attack this problem by proposing a click model with dynamic Bayesian network.

Click data is also actively applied to query log mining. (Li et al., 2008) build a bipartite-graph of click data and use it for query intent classification. (Antonellis et al., 2008) also use bipartite-graph representation of click logs and apply on query rewriting applications.

# 3. CALCULATING PAGE SIMILARITY USING CLICK DATA

In this section, we describe how the similarity between two pages can be measured using click data. A common way of measuring page similarity between two pages without click information would be comparing the contents (e.g., text, title, and in/out-link) of the two pages. However, in order to obtain such content, it requires crawling actual pages and store the content of documents on the web. Compared to our approach which doesn't require the acquisition of the actual page of a url, the content-comparison approach takes not only more storage but also takes time for crawling.

In our work, we do not look at the contents of pages for similarity computation. Instead, we hypothesize that unlabeled documents similar to positive and negative labeled documents will tend to be clicked on by the same user queries. Section 3.1 describes the click data that we used in this study and Section 3.2 explains our proposed page similarity computation using the click data.

## 3.1 Click Data

We use one month of click data collected during December 2008 from Yahoo! search engine. In order to protect the users' privacy, we removed any identifiable user information (e.g., which user clicked on which page with which query) from the click data representation. For each query, only the top 10 urls are considered for our experiments. Urls with less than 10 clicks are excluded from our experiment. This results in a total of 1.1 million urls. Session information such as what is the next url a user clicked after clicking a particular url or how long users stay in a particular url on average (i.e., dwell time) is not used for our models.

## 3.2 Building a Page Similarity Graph Using Click Data

The hypothesis that we use to compute similarities between pages is: "*Two pages that tend to be clicked by the same user queries tend to be topically similar*". For example, if a page A is clicked by a set of queries like "how to tie a tie", "how to tie a neck tie knots" and "tying a tie" and another page B is also clicked by the same set of queries, we can assume that the contents of page A and B are topically similar (i.e., the two fall into the same topical class). We consider a page as an object and queries that users issue to click this page as features. Each page is represented as a vector whose elements (i.e., features) are queries and whose values are the Pointwise Mutual Information (PMI) statistic. The following equation shows a vector of a page $p_i$.

$$p_i = (q_1 : v_1, q_2 : v_2, q_3 : v_3, \ldots, q_{ni} : v_{ni})$$

where $\{q_1, q_2, q_3, \ldots, q_{ni}\}$ are queries that were issued and resulted in a click to page $p_i$, $\{v_1, v_2, v_3, \ldots, v_{ni}\}$ are PMI scores between $p_i$ and $q$, and $n_i$ is the total number of unique queries that users issued and clicked through to $p_i$. The PMI between a page $p$ and a query $q$ is calculated as follows:

$$PMI(p,q) = \log \frac{P(p,q)}{P(p)P(q)}$$

where $P(p,q)$ is the joint probability that a page $p$ is clicked with query $q$, $P(p)$ is the probability that $p$ is clicked through from any query and $P(q)$ is the probability of a user issuing and clicking through a query $q$. When $p$ and $q$ are totally independent (i.e., $P(p,q) = P(p)P(q)$), PMI becomes 0.

A well-known problem of PMI is that it is biased towards infrequent events. Consider the case where $p$ and $q$ are statistically dependent (i.e., they have maximum association). Then $P(p,q) = P(p) = P(q)$. Hence PMI($p,q$) becomes log ($1/P(p)$) and PMI increases as the probability of $p$ decreases. Several discounting factors have been proposed to alleviate this problem. An example follows (Pantel & Lin, 2002a):

$$\frac{c_{pq}}{c_{pq}+1} \times \frac{\min\left(\sum_{i=1}^{n} c_{pi}, \sum_{j=1}^{m} c_{jq}\right)}{\min\left(\sum_{i=1}^{n} c_{pi}, \sum_{j=1}^{m} c_{jq}\right)+1}$$

where $c_{pq}$ is the frequency of clickthroughs on $p$ from $q$, $n$ is the number queries that had a clickthrough to $p$, and $m$ is the number of pages to which query $q$ clicked through.

With this vector representation of each page, we compute the similarity between two pages using the cosine similarity of their respective feature vectors[2]. Table 1 shows 3 example pages. Page P1 is clicked by 3 queries: "how to tie", "tying a tie" and "tie". The PMI value between P1 and the query "how to tie" is 2.5.

The cosine similarity between *p1* and *p2* (i.e., *sim(p1,p2)*) is greater than *sim(p1,p3)* and *sim(p2,p3)* because *p1* and *p2* share more common queries than with *p3*.

A page, represented as a node, in the similarity graph is identified by its url string. In order to avoid treating two same urls differently due to a small variation in the url, we normalize all the urls by removing "http://" at the beginning of the url and a slash "/" at the end of a url. For example the following four urls are treated as the same even if there is a small variation: "http://www.acm.org", "www.acm.org", "www.acm.org/" and "http://www.acm.org/".

Section 4 explains how we utilize the similarity graph for semi-supervised learning for page classification problems.

[2] Other similarity measures can be also applied. However, investigating which similarity measures work best on our task is not the focus of this paper. It is commonly said in IR that, properly normalized, the difference in retrieval performance using different measures is insignificant (van Rijsbergen, 1979).

**Table 1. Simplified examples of page vectors whose features are queries and feature values are Point-wise Mutual Information (PMI).**

| Page | Vector |
|---|---|
| *P1* | ("how to tie": 2.5, "tying a tie": 1.3, "tie": 0.7 ) |
| *P2* | ("how to tie": 1.9, "tying a tie": 2.3, "how do you tie": 0.9, "tie": 0.6) |
| *P3* | ( "designer tie": 2.4, "tie" 1.2) |

# 4. FINDING SIMILAR PAGES GIVEN SEED SETS

We now discuss how we augment training data for a page classifier given sets of positive and negative labeled data and a page similarity graph built by the algorithm leveraging click data described in Section 3. We call a set of same labeled data a "*seed set*" in the rest of this paper. A page in a seed set has a class label (e.g., *adult* or *non-adult* in the adult page classifier and *review* or *non-review* in review page classifier) judged by humans[3]. A page also has its contents and a url. However, for our approach of finding similar pages, we only use the url, not the contents of the page. In the following sections, we propose two algorithms for seed set expansion. The first considers the similarity score of a page and pages in the seed set whereas the second refines the first by additionally considering query validation patterns.

## 4.1 Expanding Seed Pages with Similarity Score

We can state the problem of expanding the seed set as follows:

Given two seed sets, one for positive instances ($S_{pos}$) and the other for negative instances ($S_{neg}$), and a click graph G ={( $v_{11}$, $v_{12}$, $w_1$), ( $v_{21}$, $v_{22}$, $w_2$), …, ( $v_{k1}$, $v_{k2}$, $w_k$)}, find two expanded sets $Exp_{pos}$ and $Exp_{neg}$ of nodes similar to $S_{pos}$ and $S_{neg}$, respectively, where $v_{ij}$ is a page and $w_k$ is the similarity score between $v_{k1}$ and $v_{k2}$. Table 2 shows the details of the proposed algorithm.

The algorithm is divided into two phases: **updating** score phase and **filtering** phase. In the updating score phase, with a positive seed set, the algorithm reads all the edges ($v_{i1}$, $v_{i2}$, $w_i$) in the given graph and checks if the similarity score $w_i$ is greater than threshold T1. If it is greater than T1 and one of the two urls belongs to the seed $S_{pos}$ set but the other doesn't, add the other to the expanded set $E_{pos}$. With the negative seed set, updating score phase is similar except it multiplies by -1 before adding the weight to the score. In this way, a url that has more similarity score with urls in $S_{pos}$ gets more positive value whereas a url that has more similarity score with urls in $S_{neg}$ gets more negative value.

After finishing the updating score phase, the algorithm goes through all the urls and their scores in the expanded set to eliminate urls with low scores. In the filtering phase, if a url in the expanded set has a lower absolute score value compared with a threshold T2, it gets eliminated from the set. After the filtering

[3] The main focus of classification problem we are considering in this paper is binary classification.

**Table 2. Seed set expansion algorithm**

**Input**: Two seed sets S1 and S2 and a graph G={( $v_{11}$, $v_{12}$, $w_1$), ( $v_{21}$, $v_{22}$, $w_2$), …, ( $v_{k1}$, $v_{k2}$, $w_k$)}
Two threshold parameters: T1 and T2
Initialize: E1 =(Empty), E2 =(Empty)

**Output:** Two sets of similar urls E1 and E2

**(Updating score phase)**
c: a temporary variable
for i=1 to k **do**
  read a node ($v_{i1}$, $v_{i2}$, $w_i$) in the graph
  if $w_k$ >= T1
    if $v_{i1}$ is a member of S1 and $v_{i2}$ is not, set c $v_{i2}$
    else if $v_{i2}$ is a member of S1 and $v_{i1}$ is not, set c $v_{i1}$
    if c is not a member of E1,
      add c to E1 and set score(c)=0
    score(c) = score(c) + $w_i$
    if $v_{i1}$ is a member of S2 and $v_{i2}$ is not, set c $v_{i2}$
    else if $v_{i2}$ is a member of S2 and $v_{i1}$ is not, set c $v_{i1}$
    if c is not a member of E2
      add c to E2 and set score(c)=0
    score(c) = score(c) - $w_i$
end for

**(Filtering phase)**
For each c in E1
  If score(c) < T2, remove c from E1
For each c in E2
  If score(c) >-T2, remove c from E2

phase, the algorithm only keeps urls whose aggregated similarity score in absolute value is higher than T2. In this paper, the two thresholds T1 and T2 are empirically set by performing a grid search of various threshold values on a development dataset (i.e., 0.1<T1<0.6 and 0.6<T2<1.2).

The proposed algorithm considers both the individual similarity score of a node (i.e., by threshold T1) and the aggregated similarity score (i.e., by threshold T2). If a url has a strong link with one of the members in seed set Si, it gets registered to the expanded set *Ei.* The final membership is determined by the aggregated similarity score of a node with all the nodes from the seed set who have similarity links with this node.

## 4.2 Expanding Seed Pages with Added Query Constraints

We propose another seed set expansion algorithm which considers query patterns on top of the similarity scores described in the previous section. The motivation of this algorithm is that queries that users type in for certain classes of pages, such as reviews, have common terms like "review" that strongly indicate the intent of the query as well as the class of the clicked pages. This means that if two review pages are similar, not only they have high similarity score but also their common queries may have a term like "review". Although, these patterns might be captured by algorithm in 4.1, our initial manual investigation on the result of expanded pages by algorithm in 4.1 shows that there are some noise that need to be filtered out.

For query constraint (QC) algorithm, we use the algorithm described in Table 2 with an additional module that checks whether the common queries between two nodes have certain term patterns[4]. The input for this algorithm is the following:

(1) Two seed sets S1 and S2
(2) a graph G={( $v_{11}$, $v_{12}$, $w_1$, $Q_1$ ), ( $v_{21}$, $v_{22}$, $w_2$, $Q_2$), …, ( $v_{k1}$, $v_{k2}$, $w_k$, $Q_k$)} where $Q_i$ is a set of queries that used to click both pages $v_{i1}$and $v_{i2}$.
(3) Query patterns R={r1, r2,…rj} where ri can be any lexical pattern such as a unigram and a bigram

Section 5 reports experimental results of both algorithms described in Section 4.1 and Section 4.2.

## 4.3 Active Learning

One of the main goals of this work is to investigate whether leveraging click data for Web page classification can reduce the total amount of human judgments and reach comparable levels of performance. We now propose an efficient way of selecting unlabeled data to manually annotate in an active learning framework. By carefully selecting which pages are annotated by a human judge, we can greatly reduce the number of judgments required, compared to if they were given randomly selected pages.

We first use two sets of seed urls (i.e., $S_{pos}$ and $S_{neg}$) labeled by humans and automatically identify similar pages from the similarity graph built by click data. Then we collect two expanded sets of urls, positive and negative, for which we have highest confidence (144 urls). The urls are labeled by human judges and are added to the original seed sets. Then we expand the seed sets again with the same procedure. Section 5.3 shows that this method reaches the same performance level with significantly less manually labeled data.

We investigate whether we can reduce the amount of human annotation effort by leveraging the click data. The reason we call this "active learning" is that we build an expansion model with labeled training data and use it to select next round of training data. Instead of randomly collecting more data to label, we *actively* select data from positive and negative candidates suggested by a model (described in 4.1).

# 5. Experimental Results

In this section we present an extensive evaluation of our label-propagation algorithm over the following three classification tasks:

- **`how-to:`** pages that describe procedural knowledge on how to accomplish certain tasks such as fixing computer problems, painting a house, and cleaning leather.
- **`adult:`** pages that contain explicit vulgar, violent or sexual content; and
- **`review:`** pages that display either user or editorial reviews of products or services.

Our experiments focus on two primary questions: 1) the precision of our label propagation algorithms presented in Section 4; and 2)

[4] The query patterns are manually collected by looking at shared queries between pages in positive classes of a separate development set.

the effect of our expanded training sets on both classification accuracy and saved judgments. We answer the first question by manually evaluating the labels assigned to a random sample of instances expanded by our methods, and we answer the second question by comparing the classification performance of a model trained with only manually labeled data with models trained with our added expanded data (using both our passive and active algorithms from Section 4).

We now describe our experimental setup and our experimental results.

## 5.1 Experimental Setup

### *5.1.1 Data Sets*

The training data for each of our three classification tasks consists of 10,000 manually labeled positive and negative examples by a large team of professional editors.

The data was collected in two stages. First, we collect a data pool by selecting a large number of random queries from our search logs, fully anonymized, and then obtain the top-20 search results returned by Yahoo! search engine. These urls were then manually labeled by our editors. In order to increase the number of positive examples, in the second stage, we build a second data pool by manually collecting queries that are likely to surface positive pages, and then obtain and manually label the top-20 search results. For example, for our `review` classifier, we can use queries such as "digital camera reviews" or "baby swing reviews", and for our `how-to` classifier, we can use "how to clean uggs" or "best way to loose weight". The reason we combine these two approaches is that the data pool from the first approach is likely to contain too few positive examples and this will cause our training data to be skewed too much towards negative examples. Again, once the training data is collected, trained professional editors review each url and judge whether the page is positive or negative. A subset of labeled data is annotated by multiple human judges and is used for checking data quality via human agreement.

To test our expansion methods, we sub-sampled from each labeled set four training sets of varying sizes: 1000, 2000, 5000, and 10,000.

Our test data is collected in a similar way. Table 3 describes the number of urls in the test sets for our three classification tasks.

### *5.1.2 Baseline*

As a strong baseline, we employ fully supervised commercial-grade web page classification system used at Yahoo!. For our model, we use a Gradient Boosting Decision Tree (GBDT, Hastie et al. 2001) with a large collection of manually annotated positive and negative training examples. GBDT first trains a weak learner ($f_1$) with a sample of the training data and then re-weights other samples based on the $f_1$ classifier (to overweight difficult examples). GBDT then learns a new model based on the weighted samples and iteratively repeats this procedure. Our preliminary result has shown that GBDT outperforms SVM in web page classification tasks. Although SVM seems the better choice in regular text classification with a bag-of-words mode, GBDT performs better in web page classification with access to non-textual features such as page structure, in-link and out-link patterns, and url patterns (described in 5.1.3). The exact reason why GBDT performs better for this task still needs to be further studied but we believe that it might be due to the fact that GDBT doesn't require feature value normalization. With various groups

**Table 3. The number of positive and negative data in test sets**

| Model Name | # positive data | # negative data |
|---|---|---|
| Howto | 1248 | 2297 |
| Adult | 1048 | 2765 |
| Review | 511 | 903 |

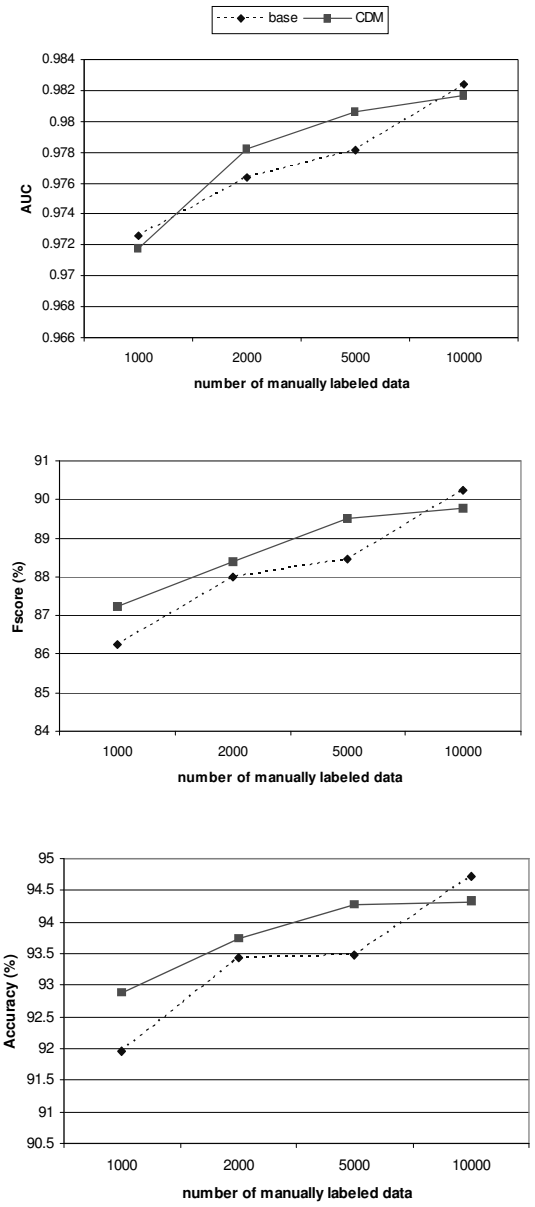
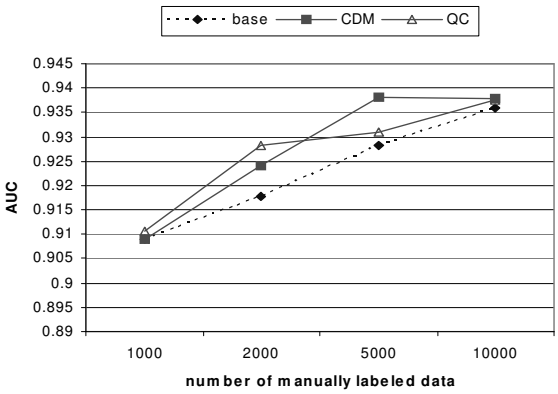
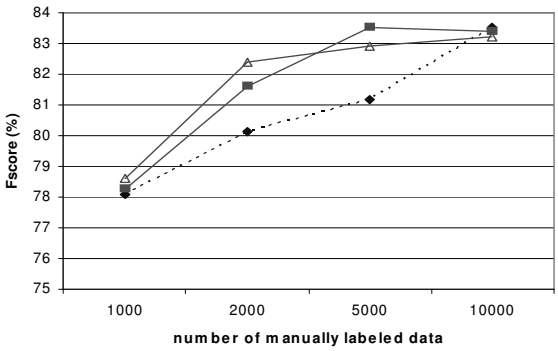



**Figure 1. Adult models' AUC, Fscore and Accuraccy comparison between baseline system and Click Data Model (CDM)**



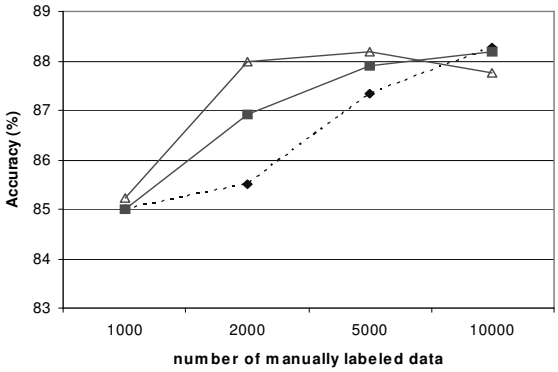
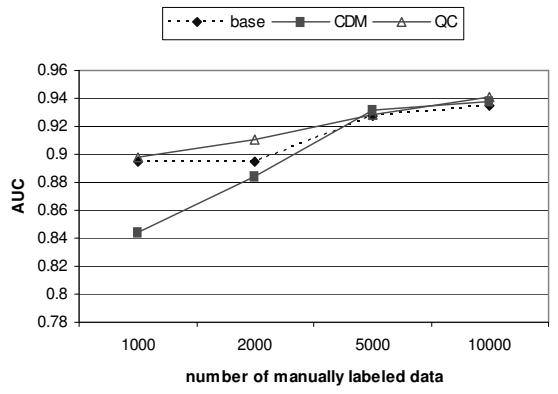
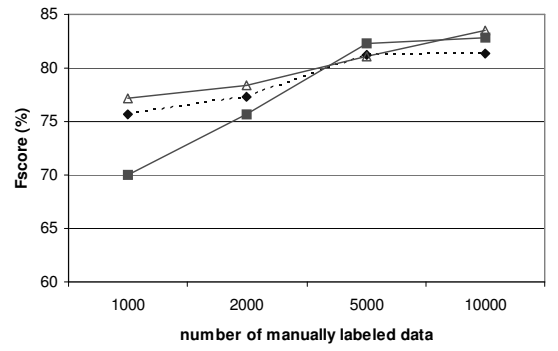




**Figure 2. Review models' AUC, Fscore and Accuraccy comparison between baseline system, Click Data Model (CDM) and Query Constraint Model (QC).**



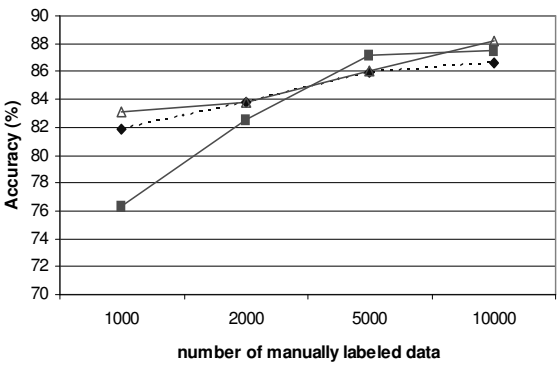




**Figure 3. Howto models' AUC, Fscore and Accuraccy comparison between baseline system ,Click Data Model (CDM) and Query Constraint Model (QC).**

of features whose value ranges are quite diverse, it is a challenge to find optimal way of normalizing values for SVM. GBDT also has much faster training and decoding time compared with SVM and results in much smaller models to be stored in memory, both very desirable properties for a commercial web page classifier used for classifying billions of web pages.

### *5.1.3 Features*

Our classifier uses various groups of features. *Textual features* contain ngram features such as unigrams and bigrams in different sources of a page (e.g., body, title, and anchor text). It also includes the total number of words in a page and the total number of capitalized words. *Link features* of a page represent patterns of out-links (i.e., urls that the page points to) and in-link (i.e., pages that point to this page). *Url features* counts the length of a url and checks if a url contains certain terms. *HTML features* specifically look into html tags and check weather a page contains certain html tags(e.g., <i> and <b>). *Other features* consider if a page contains images, tables, and javascript.

### 5.1.4 Systems

We instantiated our CDM and QC models from Section 4 using fully anonymized click log statistics collected in December 2008 at a commercial search engine, as described in Section 3. Both CDM and QC are modeled in the same way as the *Baseline* using GBDT and the same set of features.

### 5.1.5 Metrics

Section 5.4 presents the evaluation result of how well the proposed methodology correctly identifies similar pages. We measure this by precision of both positive and negative expanded urls. Human judgment is considered as a correct label for a url.

The rest of the section will present how the expanded pages affect the classification performance. We measure our classifiers by three standard metrics: Area Under the ROC Curve (AUC, Fawcett, 2003), Fscore, and Accuracy.

## 5.2 Label Propagation Analysis

This section investigates whether expanding training data using click data improves web page classification performance and whether we can reduce the required number of human judgments. For each classification task, we compare our CDM and QC models against the strong baseline presented in Section 5.1.2.

Figures 1, 2, and 3 illustrate the performance gains of our CDM and QC models compared with our state-of-the-art baseline Web page classifier in terms of AUC, F-score and Accuracy, on the `adult`, `review` and `how-to` classification tasks, respectively. For each classifier, we use different sizes of seed data: 1000, 2000, 5000, and 10,000.

Below we analyze the performance of CDM and QC.

### 5.2.1 CDM Performance Analysis

The amount of expanded data is controlled by two similarity thresholds (i.e., T1 and T2 in Table 2). In our experiments, we experimentally set these thresholds by optimizing them using a grid search over a held out development set of 200 examples, annotated the same way as described in Section 5.1.1. Intuitively, high thresholds result in more precise expansions at the cost of fewer amounts of data. On the other hand, low thresholds increase coverage but result in lower precision. We expect high precision to be most important, however too few training expansions would likely not affect the GBDT modeling.

**`adult`**: For the adult classification task, CDM mostly performs better than the baseline in all three measures except the model with 10,000 manually labeled examples (Figure 1). With large amounts of manual data, the benefit of click data is minimal since newly discovered pages by CDM are subsumed by other manually labeled data. The biggest improvement of CDM is observed with a model using 5000 labeled data as a seed set (+1.07% in F-score, +0.81% in Accuracy and +0.25% in AUC).

**`review`**: Similar to above, CDM outperforms the baseline in all three measures (Figure 2) except when the labeled data is too small (i.e., 1000) or too many (i.e., 10,000). With small amount of labeled data, our expansion method does not discover many new pages and the system performs almost the same as the baseline. With 10,000 labeled pages, again the extra urls added by our method didn't provide useful signal to the classifier. An interesting observation is that CDM with 5000 manually labeled data performs as well as the baseline system using 10,000 manually labeled examples. By leveraging click data with our proposed method, we reduce the manual labor by 50%. Also, CDM with 2000 labeled examples performs similar to the baseline with 5000 manually labeled examples. This demonstrates our initial hypothesis that user click data can reduce the total amount of human labeling cost.

**`how-to`**: CDM on the how-to classification tasks behaves differently from the other two. With 1000 and 2000 human labeled data, CDM performs worse than the baseline. Manual inspection showed that the expanded urls are very noisy and in the next paragraph we report error analysis. With more labeled data (5000 and 10000), CDM performs better than the baseline. With 5000 seed examples, CDM outperforms the baseline with 10,000 data.

We performed error analysis to understand why many false positive and false negative pages were found. Interestingly, some positive urls discovered by our system are indeed how-to pages but they negatively impacted the performance of CDM because they are "excluded" by our class definition. For the example query "how to fix broken necklace", a page with a text description of the steps to fix a necklace has very high page similarity with a page with a video clip showing how to fix it without any textual description. Since our definition of how-to was restricted to analyzing textual descriptions, adding the latter page doesn't improve the classifier performance. Another type of false positive is a page whose content has changed or is deleted. For example, for a query "iphone review", a page with "iphone review" in its title but without a written review is likely to have high similarity to positive review pages since it shares queries with them and is likely to be clicked by users. However, since the page does not have a review and contains other contents such as advertisements and a list of links to buy an iPhone, the performance of the system deteriorates. Since our expansion algorithm does not leverage the textual contents of expanded pages, it is impossible to detect this type of false positive pages. This issue remains a direction for future improvements (see Section 6).

Not all the urls in the human labeled data occurred in the one month click data we used to build the similarity graph. We categorize this data into the following three types:

| Data type | Human labeled | Present in Click Graph |
|---|---|---|
| TypeA | Yes | Yes |
| TypeB | Yes | No |
| TypeC | No | Yes |

TypeA is labeled by human editors and is also present in our page similarity click graph. TypeC is automatically expanded data using TypeA as a seed set. TypeB is labeled by human editors but does not appear in our similarity graph. There are three main reasons why TypeB exists: 1) much of our human-labeled data we use for our research was collected and annotated before click data was available; 2) our experiment uses only one month of click data so urls in TypeB may occur in other months or years; and 3) even if some urls appear in the one month click data, they may get filtered out because we didn't consider a (query, url) pair if the url is clicked less than 10 times with the particular query.

**Table 4. A comparison between a baseline model and a click data model. All the manually annotated data is also present in the click graph.**

| Classifier | model | AUC | Accuracy | Fscore | Precision | Recall | # positive | | # negative | |
|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | | | Manual annotation? | | | |
| | | | | | | | Yes | No | Yes | No |
| Howto | Baseline | 0.8817 | 80.31 | 74.32 | 68.71 | 80.93 | 284 | 0 | 534 | 0 |
| Howto | CDM | 0.8889 | 81.10 | 75.51 | 69.42 | 82.77 | 284 | 63 | 534 | 81 |
| Review | Baseline | 0.7815 | 64.64 | 64.18 | 50.62 | 87.67 | 62 | 0 | 250 | 0 |
| Review | CDM | **0.8216** | **75.53** | **68.60** | 63.96 | 73.97 | 62 | 15 | 250 | 97 |
| Adult | Baseline | 0.9552 | 89.61 | 81.93 | 78.50 | 85.69 | 111 | 0 | 496 | 0 |
| Adult | CDM | 0.9614 | **91.87** | **84.55** | 88.52 | 80.92 | 111 | 84 | 496 | 524 |

One interesting question is how much improvement we can expect if we only had TypeA labeled data and limited editorial resources? Currently, the baseline system in Figures 1, 2 and 3 are trained with both TypeA and TypeB, and CDMs are trained with all three types. TypeB is manually labeled but does not contribute to discovering new pages (i.e., TypeC). In order to investigate this scenario, we performed another experiment with the following setup: 1) collect sample pages that occur in the click data (i.e., TypeA) and 2) expand the urls using the page similarity in the click graph using (i.e., collecting data of TypeC). The baseline system is trained with TypeA and a Click data model is trained with both TypeA and TypeC. The result is shown in Table 4. As we can see from the table, CDM on the all three models had improvements on all three classification metrics. This suggests that for new tasks, when we can afford only a small number of manually labeled examples, it is better to sample the training examples from the click graph so that we can use all the labeled data as seeds and maximize the number of automatically expanded data.

### *5.2.2 QC Performance Analysis*

This section analyses the performance of the QC model as depicted in Figures 2 and 3[5]. The QC model expands seed sets as in CDM and then adds term pattern constraints in the shared queries (see Section 4.2 for the description of the algorithm.) Note that the goal of QC model is to filter out false positives from CDM by looking at patterns in the shared queries between seeds and candidate expansions.

**`how-to`**: For the how-to classifier, we filter out any page that does not have terms that strongly indicate the user intent such as "how to" in the queries of the page even if the page has high similarity score with pages in the seed set. With small training data (i.e., 1000 and 2000), QC outperforms CDM yet CDM performs better with 5000 and 10000 examples.

**`review`**: For review page, similarly, we exclude pages that do not have "review" in the query terms. QC outperforms the baseline in all three measures and outperforms CDM with small amounts of manually labeled examples (i.e., 1000 and 2000). The QC model with 2000 examples outperforms the baseline with 5000 examples. However, QC performs worse than CDM with 5000 examples, which may be due to the fact that QC doesn't have as many expanded data as CDM.

We conclude that QC is useful especially when manually labeled data is small. It can reduce the noise in expanded data. However, error analysis showed that with large collections of manually labeled examples, QC tended to filter out many true positives.

## 5.3 Evaluating the Active Learning Approach

The question we tried to answer to in this section is "Can we adopt CDM model in the active learning frame work?". We investigate whether we can reduce the amount of human annotation effort by introducing the proposed method in data collection phase. This algorithm is described in Section 4.3. We apply the method in howto model (Table 5). We first use a seed set (*Seed1*) labeled by human annotators. Then we expand Seed1 and collect a set of similar urls by our proposed system using page similarity score (*Expan1*). Human annotators label Expan1 and create another set (*Seed2*). This annotation is likely to correct noise in Expan1. We apply the expansion algorithm and collect a set of similar urls using both Seed1 and Seed2 as seed sets (*Expan2*). Then we compare a system trained with only Seed1 (*SYS_Seed1*), a system trained with both Seed1 and Seed2 (*SYS_Seed12*), and a system trained with all Seed1, Seed2 and Expan2 (*SYS_all*). SYS_Seed1 required 818 human judgments whereas SYS_Seed12 needed 962 judgments total. SYS_all also needed only 962 human judgments since Expan2 (124 pages) is automatically generated without human annotation. SYS_all outperforms both SYS_Seed1 and SYS_Seed2. It also outperforms a system trained with 3000 human labeled data (*SYS_3000*). By applying the proposed system of automatically augmenting labeled data in the data collection phase, we were able to build a model with only 962 human judgments yet the system outperformed a system with 3000 human judgment.

## 5.4 Intrinsic Analysis of CDM Expanded Data

Now we evaluate the quality of our CDM expansion algorithm outside of the web page classification tasks. Specifically, for the `how-to` classifier, we took a random sample of 50 positive and 50 negative examples expanded by our model. We manually evaluated the assigned class labels (without looking at the system classification) and report below the precision of both classes. Recall is difficult to approximate since it is impossible to get a complete list of all `how-to` pages on the Web. Positive class has 82.3% precision whereas negative class has 83.6% precision.

[5] Note that we did not apply the QC model on the *adult* query classifier since the constraint patterns were too offensive to analyze. With the adult class, one would require a broad set of constraint patterns since there are a lot of diverse adult terms that indicate an adult intent.

**Table 5. Collecting training data with data augmentation algorithm using click graph can reduce the number of total judgments required to reach the comparable level of performance. The system trained with 962 labeled data (from `howto`) using the proposed method outperforms a system trained with 3000 labeled urls with state-of-the-art classification method.**

| Model name | Model trained with | AUC | Acc. | Fscore | Prec. | Recall | # labeled data |
|---|---|---|---|---|---|---|---|
| SYS_Seed1 | Seed1(818) | 0.8817 | 80.31 | 74.32 | 68.71 | 80.93 | 818 |
| SYS_Seed12 | Seed1+Seed2(144) | 0.9054 | 83.07 | 77.65 | 72.56 | 83.49 | 962 |
| **SYS_all** | Seed1+Seed2+ Expan2(124) | 0.9072 | 83.64 | 78.66 | 72.72 | 85.66 | **962** |
| SYS_3000 | 3000 urls | 0.9061 | 83.84 | 77.45 | 76.10 | 78.85 | 3000 |

# 6. DISCUSSION AND FUTURE WORK

**Is the proposed method always useful for web page classification?**: Intuitively, our hypothesis, that *two pages that tend to be clicked by the same user queries tend to be topically similar*, is better suited for topical page classification tasks such as *review* and *adult* than non-topical classification tasks such as *spam page* (i.e., a page that manipulates search engines in order to get itself ranked higher than it actually deserves) and *missing page* (i.e., a page whose main content has been removed or deleted). For example, even if two pages share same queries in a click graph, one page could simply be a missing page yet the other could be not. Among topical classification tasks, however, it is expected that some classification task gets more benefit than others. In a practical point of view, this would be empirically tested without extra cost once a page similarity graph using click data is constructed. Since we need training and testing data for any supervised learning method, we can first label two sets of data. Then we can simply use part of the training data as seed set, expand the data set with thresholds learned with part of a training data and test with a test set.

**How can we improve the quality of automatically labeled data from unlabeled data?**: As one way of improving the quality of automatically expanded data, we proposed the query constraint model in this paper (Section 4.2). Empirical results demonstrate that the QC model outperforms both the baseline and CDM model, especially when the seed set size is small (Section 5.2.2). Our vision for an approach to further assuring the quality of the expanded data is to consider the contents of those expanded pages. This still doesn't require crawling all pages in the similarity graph. Once our system suggests candidate pages as positive of negative, we can crawl the contents of only those pages and apply a content analysis system (e.g., a classifier or a pattern matcher) to select more convincing candidate pages. It remains as future work how to build an efficient content analysis system for this purpose.

In the future, we would also like to explore if more click data (e.g., one year click data) would improve the performance of our current system, which uses only one month click data. Also, it would be interesting to investigate the impact of propagating class labels of seed sets to nodes that are not directly connected to seed sets. This will introduce more noise to the expanded set but ultimately discover more pages. Finally, we would like to examine more sophisticated methods for query constraint than the current model that is based on simple pattern matching in shared queries.

# 7. CONCLUSION

In this paper, we have presented a method to improve webpage classifiers by leveraging click data to augment training data by propagating class labels to unlabeled similar pages. Our hypothesis is that unlabeled pages similar to our positive and negative labeled documents will tend to be clicked on by the same user queries. Thus, we can augment manually labeled data by modeling the similarity between pages in a click graph. We demonstrated the benefit of our approach with three different classification problems. The proposed method required much less human labeled training data than current state-of-the-art classification methodologies. We also proposed an active learning method using click data. It outperformed current models with less amount of human annotation effort.

# ACKNOWLEDGMENTS

Our thanks go to anonymous reviewers, click data provider, and many annotators who labeled our classification data.

# 8. REFERENCES

[1] Chakrabarti, S., B. Dom, and P. Indyk. Enhanced hypertext categorization using hyperlinks. In SIGMOD '98, Proceedings of the 1998 ACM SIGMOD international conference on Management of data, pages 307–318, New York, NY, USA, 1998. ACM Press

[2] Chapelle, O. and Y. Zhang. A dynamic Bayesian network click model for web search ranking. In Proceedings of the 18th International World Wide Web Conference (WWW), 2009.

[3] Church, K. and Hanks, P. 1989. Word association norms, mutual information, and lexicography. In Proceedings of ACL89. pp. 76–83.

[4] Craswell, N., O. Zoeter, M. Taylor, and B. Ramsey, An experimental comparison of click position-bias models, in WSDM '08: Proceedings of the international conference on Web search and web data mining, ACM, New York, NY, USA, 2008

[5] Dou, Z., Song, R., Yuan, X. and Wen, J-R. 2008. Are Clickthrough Data Adequate for Learning Web Search Rankings? In proceeding of the 17th ACM conference on Information and knowledge management. Napa Valley, California, USA

[6] Fawcett, T., ROC Graphs: Notes and Practical Considerations for Data Mining Researchers. HP Labs Tech Report HPL-2003-4. 2003.

[7] Hastie, T., Tibshirani, R., and Friedman, J. The Elements of Statistical Learning, Data Mining, Inference, and Prediction, Second Edition, pages 299-345, Springer, 2001

[8] Joachims, T. 2001. A Statistical Learning Model of Text Classification with Support Vector Machines. Proceedings of the Conference on Research and Development in Information Retrieval (SIGIR), ACM,.

[9] Joachims, T. 2002. Optimizing Search Engines Using Clickthrough Data, Proceedings of the ACM Conference on Knowledge Discovery and Data Mining (KDD), ACM.

[10] Kamal N., A. McCallum and T. Mitchell. Semi-supervised Text Classification Using EM. In Chapelle, O., Zien, A., and Scholkopf, B. (Eds.) Semi-Supervised Learning. MIT Press: Boston. 2006.

[11] Li, X., Y.-Y. Wang , A. Acero, Learning query intent from regularized click graphs, Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval, July 20-24, 2008, Singapore.

[12] Patrick P. and D. Lin. 2002. Discovering Word Senses from Text. In Proceedings of ACM Conference on Knowledge Discovery and Data Mining (KDD-02). pp. 613-619. Edmonton, Canada.

[13] Radlinski, F., M. Kurup, and T. Joachims. 2008. How Does Clickthrough Data Reflect Retrieval Quality?, Proceedings of the 17th ACM Conference on Information and Knowledge Management (CIKM)

[14] C. J. "Keith" van Rijsbergen. Information Retrieval. Butterworth, 1979.

[15] Yang, Y., S. Slattery and R. Ghani. A Study of Approaches to Hypertext Categorization, Journal of Intelligent Information Systems, pages 219–241,Volume 18, Number 2, March 2002.