Knowing a Web Page by the Company It Keeps*

Xiaoguang Qi and **Brian D. Davison**
Department of Computer Science & Engineering
Lehigh University, Bethlehem, PA 18015 USA
{xiq204,davison}@cse.lehigh.edu

June 2006

Abstract

Web page classification is important to many tasks in information retrieval and web mining. However, applying traditional textual classifiers on web data often produces unsatisfying results. Fortunately, hyperlink information provides important clues to the categorization of a web page. In this paper, an improved method is proposed to enhance web page classification by utilizing the class information from neighboring pages in the link graph. The categories represented by four kinds of neighbors (parents, children, siblings and spouses) are combined to help with the page in question. In experiments to study the effect of these factors on our algorithm, we find that the method proposed is able to boost the classification accuracy of common textual classifiers from around 70% to more than 90% on a large dataset of pages from the Open Directory Project, and outperforms existing algorithms. Unlike prior techniques, our approach utilizes same-host links and can improve classification accuracy even when neighboring pages are unlabeled. Finally, while sibling pages are found to be the most important type of neighbor to use, the other types can also contribute.

1 Introduction

Text classification plays a fundamental role in a number of information management and retrieval tasks. On the Web, classification of page content is essential to focused crawling [6], to the assisted development of web directories such those provided by Yahoo [26] and the Open Directory Project (ODP) [19], topic-specific web link analysis [10, 18, 21], and analysis of the topical structure of the Web [5]. Web page classification can also help improve the quality of query search [7] and web spam demotion [25].

*Technical Report LU-CSE-06-011, Department of Computer Science and Engineering, Lehigh University, Bethlehem, PA, 18015.

The general problem of text classification is well-studied and different classifiers have shown good performance in plain text classification tasks. Unfortunately, simply applying textual classifiers to web documents often produces unsatisfying results because web page classification, or hypertext classification, is a different scenario.

Research shows, however, that incorporating link information along with the content of web pages can enhance classification [4, 3]. Here, we propose to use the topic vector of neighboring pages to help the categorization of a web page. Unlike existing work, our method does not rely on the appearance of labeled pages in the neighborhood of the page under scrutiny, and thus has wider applicability. In addition, not only sibling pages but also three other kinds of neighboring pages are taken into consideration.

The rest of this paper will be organized as follows. In Section 2, we review recent work on using hyperlink information to enhance web page classification. In Section 3, our approach is detailed. Then, the experimental setup and results are shown in Section 4. We conclude with a discussion and a summary of our results in Sections 5 and 6.

2 Related Work

Although web page classification is more challenging than classifying plain text documents, the web pages themselves provide extra features that may help classification, such as HTML tags, URLs, hyperlinks and anchor texts. Much research has been done to utilize these features in web page classification. We categorize this work into two general classes according to the features being used: work that uses only on-page features and that using features from neighboring pages.

On-page features are those from the pages being classified themselves. Kovacevic et al. [16] performed analysis on the web page visual layout to boost the performance of web page classification. Similarly, Shen et al. [22] also used information from components in web page layout. Kan and Thi [14, 15] proposed an approach to classify web pages by using their URLs, in which no HTML content is needed. Other research in this category can be found in [27, 8, 12, 24].

The second category contains research utilizing information in pages nearby in the web link graph. A number of researchers have proposed using some or all of the content in neighboring pages to help inform the classifier about the target page.

Attardi et al. [1] proposed using content information from the pages citing the page being classified to help determine the page's topic. Similarly, Glover et al. [9] proposed an approach using extended anchortexts. Slattery and Mitchell [23] presented the co-citation relation in the form of recursive learning rules and inserted them into a relational learner, FOIL, to improve the accuracy. These three approaches ([1, 9, 23]) only used one particular kind of neighboring page while ignoring the others.

Chakrabarti et al. [4] have shown that directly including neighboring pages'

textual content into the page does not improve the performance of classification because too much noise is introduced in this approach. After studying different kinds of web graph relationships between pages, however, they also showed that labeling a page according to the majority labels of its sibling pages can enhance classification.

Calado et al. [2] evaluated four similarity measures derived from web link structure, and demonstrated that using a Bayesian network model combining the class-similarity measures with the results obtained by traditional content-based classifiers can greatly improve classification performance. Unlike our work, Calado et al. discarded links between pages of the same site, and they do not directly test the influence of parent and child pages.

These two approaches ([4, 2]) rely on the presence of labeled neighbors of the page in question. Otherwise, they would either suffer significantly in terms of coverage (leaving a number of pages undecidable) or reduce to the result of traditional content-based classifiers.

3 Approach

Based on the previous research, it is believed that combining information of neighboring web pages can enhance the web page classification. In the following, the page to be classified is called the "target page", and nearby pages in the link graph are called the "neighboring pages".

3.1 Analyzing the neighborhood

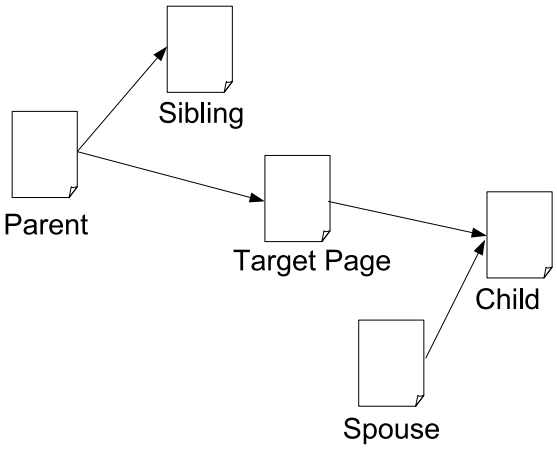
In our work, in order to help classify a target page, we use nearby pages with four kinds of relationships: parent pages, child pages, sibling pages and spousal pages. We name them A, B, C and D, respectively, as is described in Figure 1.

These four sets may overlap with each other. In other words, a page may have multiple roles. For example, a page can be both the sibling and spouse of the target page at the same time. In that case, both roles count.

Furthermore, each of these sets can be divided into two subsets: labeled pages and unlabeled pages. Labeled pages are those pages whose categories are already known, such as the pages in the ODP. Unlabeled pages are those whose labels are not known.

Since the classifier produces an approximation to the human labeling, we will use the human judgment whenever it is available. Otherwise, a text-based classifier will be used to classify them.

In fact, instead of hard labeling, soft classification is used as the internal representation in our work. That is, the probabilities of the page being in each category are given as the classification result. Therefore, after classification, each page p can be represented by a probability distribution vector $v_p = (v_{p,1}, v_{p,2}, ..., v_{p,i}, ..., v_{p,n})$, in which each component $v_{p,i}$ is the probability of the page being in the corresponding category c_i. This vector is referred to as "topic vector" in this paper. Again, for unlabeled pages, this vector is given by



A_p	parent pages	$\{\ q \mid q \rightarrow p \ and \ q \neq p\ \}$
B_p	child pages	$\{\ q \mid p \rightarrow q \ and \ q \neq p\ \}$
C_p	sibling pages	$\{\ q \mid \exists\ r\ s.t.\ r \rightarrow p,\ r \rightarrow\ q \ and \ q \neq p\ \}$
D_p	spousal pages	$\{\ q \mid \exists\ r\ s.t.\ p \rightarrow r,\ q \rightarrow\ r \ and \ q \neq p\ \}$

Figure 1: Four kinds of neighboring pages of p

the classifier. For labeled pages, this vector is set according to Equation 1:

$$v_{p,k} = \begin{cases} 1 & \text{if } C[k] = L[p] \\ 0 & \text{if } C[k] \neq L[p] \end{cases} \quad (1)$$

where C is a sorted list of the names of each category, and $L[p]$ is the human labeling of page p. Such soft classification is mainly used for internal representation. Although the output of our algorithm is also in the form of probability distribution, it is converted into hard labeling for the purpose of evaluation, i.e., labeling the page by the class that it mostly probably belongs to.

The reason why we do soft classification rather than hard labeling is based on the observations of real-world pages. First, web pages have complex structures and each part of the page may talk about related but different topics. Second, even for those pages which concentrate on one topic, it is natural that the topic may belong to multiple categories. For example, the homepage of a health insurance company may belong to both "Business" and "Health". Part of the reason for this lies in the ambiguously-defined taxonomy. Many pages in the ODP Directory are labeled in multiple categories by human editors, which fortifies our confidence in using soft classification.

So far, the neighboring pages have been divided into four sets according to the link structure. Each of them is further categorized into two subsets according to the availability of its pre-determined label. In addition, each page in these subsets is represented by a topic vector. In Section 3.2, we will talk about how to utilize these pieces of information to help classify the target page.

3.2 Utilizing the neighboring information

After analyzing the neighborhood structure, the neighboring pages are categorized into four different sets. Each of them is then divided into two subsets. Each page within those subsets is represented by a topic vector v_p. In the next step, these topic vectors combined with the topic vector of the target page will be combined to improve classification performance.

In general, the topic vectors of all neighboring pages may help in determining the target page's category. For example, we could set the target page's class to the majority class of all neighbors. However, we find that different types of pages are of different importance to this purpose according to their relationship to the target. Therefore, weighting factors will be used to introduce bias among different subsets of neighboring pages.

3.2.1 Bias on labeled pages

As described in Section 3.1, the pages in the neighborhood of the target page can be either labeled or not. The topic vectors of labeled pages are determined by the human labeling, while the topic vectors of unlabeled ones are produced by a classifier. In order to keep under control the noise introduced by the classifier, a factor η (where $0 \leq \eta \leq 1$) is used to down-weight the vectors of unlabeled pages. That is, we modify the topic vector v_p by multiplying it by its weight $w(p)$. The modified topic vector v'_p is computed by Equation 2.

$$v'_{p,k} = v_{p,k} \times w(p) \tag{2}$$

where $w(p) = \begin{cases} 1 & \text{if } p \text{ is labeled} \\ \eta & \text{if } p \text{ is unlabeled} \end{cases}$

When η=0, the influence coming from those unlabeled pages will be totally ignored, which implies that we don't trust the classifier at all. When η=1, the unlabeled pages are treated equally as the labeled ones, which means we assume the classifier is doing as well as human labeling.

3.2.2 Intra-host link bias

Links connecting pages within the same web site often serve the purpose of navigation and do not confer authority. Therefore, they are often ignored or considered less useful in the scenario of link-based ranking. However, the situation can be different in web page classification tasks. For example, on a shopping web site, a product list in digital camera category may contain links to all the digital cameras, which are also on the same topic. By using the parameter θ in our algorithm, we are raising the question of whether internal links are useful in web page classification. Or, more specifically, do internal links imply some connection on topics between the two pages?

In order to find out the answer, θ is used to weight the influence of the neighboring pages that within the same web host of the target page. We modify the topic vector again to include this parameter.

$$v''_{p,k} = \begin{cases} \theta \cdot v'_{p,k} & host(p) == host(s) \\ v'_{p,k} & host(p) \neq host(s) \end{cases} \quad (3)$$

where s is the target page and *host* is a function that returns a page's host name.

When θ=0, intra-host links are dropped. When θ=1, intra-host and inter-host links are counted equally. Note that we do not limit θ to be between 0 and 1. Values larger than 1 are also acceptable, which means we are open to the possibility that intra-host links are more important than inter-host ones.

3.2.3 Counting the multiple paths

Now that we generated and modified the topic vector for each page in the neighborhood, it is time to consider the relationship between the target page and the neighboring pages. Here, an interesting issue may arise: if a sibling page has more than one parent in common with the target page, that is, in a link graph view, there are multiple paths between the target page and its sibling page, should the weight be counted as one or as the number of parents in common? The same question applies to the spousal pages, too. We leave this question to be answered by the experiments.

In the weighted path variation, the influence of a sibling page (or a spousal page) to the target page's topic is weighted by the number of common parents (or children). In the unweighted version, such weighting scheme is ignored. That is, no matter how many parents (or children) that are held in common, it is counted only once.

3.2.4 Bias among neighbors

In analyzing phase, we used 4 types of neighboring pages, parent pages (A), child pages (B), sibling pages (C) and spousal pages (D). We expect that the pages in these four sets may have different influence on the target page's topic. Therefore, a weighting vector $\beta = (\beta_1, \beta_2, \beta_3, \beta_4)$ is used to bias among them, where $\beta_1, \beta_2, \beta_3, \beta_4 \geq 0$ and $\beta_1 + \beta_2 + \beta_3 + \beta_4 = 1$.

The combined neighboring pages' topic vector v_n can be computed by Equation 4.

$$\begin{aligned} v_{n,k} = \beta_1 \times \frac{\sum_{p \in A} v''_{p,k}}{\sum_{p \in A} w(p)} + \beta_2 \times \frac{\sum_{p \in B} v''_{p,k}}{\sum_{p \in B} w(p)} \\ + \beta_3 \times \frac{\sum_{p \in C} v''_{p,k}}{\sum_{p \in C} w(p)} + \beta_4 \times \frac{\sum_{p \in D} v''_{p,k}}{\sum_{p \in D} w(p)} \end{aligned} \quad (4)$$

3.2.5 Combining neighboring pages with target page

Like neighboring pages, the target page s will also go through the classifier and get its topic vector v_s. Then the combined topic vector v for the target page s

will be a weighted combination of v_s and v_n.

$$v_k = \alpha \times v_{s,k} + (1 - \alpha) v_{n,k} \tag{5}$$

or in vector representation:

$$\vec{v} = \alpha \times \vec{v}_s + (1 - \alpha) \vec{v}_n \tag{6}$$

where $(0 \leq \alpha \leq 1)$.

When α=0, the labeling of target page is solely decided by its neighbors without looking at its own content at all. When α=1, the labeling is simply based on the pure textual classifier while the information from the neighboring pages are ignored.

Now that the combined topic vector v is obtained by taking into consideration all the neighboring pages' information as well as that of the target page, a conversion from probabilistic distribution to hard labeling is need before evaluation. The conversion simply picks the category corresponding to the largest component in v as the label of the target page.

4 Experiments

4.1 Experimental setup

4.1.1 Taxonomy

In our work, the Open Directory Project [19] is used as the taxonomy. Constructed and maintained by a large community of volunteer editors, the ODP, also as known as the dmoz Directory, is claimed to be the largest human-edited directory of the Web.

The metadata being used in our work was downloaded from dmoz.org in September 2004, and contains 0.6 million categories and 4.4 million leaf nodes. A crawler was used to fetch the Web pages pointed to by the ODP, out of which 95% were successfully downloaded.

4.1.2 HTML files preprocessing

All of the Web pages which are used in our work have gone through a text preprocessor. This includes the pages to train the classifier, as well as the target pages and the neighboring pages.

The functionalities of the preprocessor are as follows:

- eliminate HTML tags except the "keywords" and "description" metatags (because they may be of help in deciding a page's topic);
- unescape escaped characters;
- eliminate characters other than alphanumeric characters;

Arts	Business	Computers	Games
Health	Home	Recreation	Reference
Science	Shopping	Society	Sports

Table 1: Set of twelve top-level categories used in dmoz Directory

- eliminate terms whose length exceeds a certain limit (4096 characters in this case, because they will cause a crash in *Rainbow*, the classifier being used).

Therefore, after preprocessing, each HTML file is transformed into a stream of terms.

The preprocessing is essential for at least three reasons. First, it filters out noisy terms such as "html", "body", which may compromise the classification accuracy. In our experience, this preprocessing can increase the classification accuracy by 3% (in absolute value), if not more. Second, Rainbow cannot function on the original HTML files without preprocessing due to terms that are too long. Finally, preprocessing eliminates some unnecessary features and makes web pages shorter, which helps reduce the time and space required by the classifier.

4.1.3 Text-based classifier training

Two common textual classifiers are used in our experiments: *Rainbow* [17], a text classifier based on the *Bow* library, and SVM^{light} [13], a implementation of Support Vector Machine in C developed by Joachims. We wish to determine whether the choice of text classifier has an effect on the performance of our algorithm.

First, similar to the work by Chakrabarti et al. [4], 12 out of the 17 top-level categories of the dmoz Directory are selected, which are listed in Table 1. After that, 19,000 pages are randomly selected from each of the 12 categories (i.e., 228,000 in total) and are used to train the textual classifier, using *Rainbow*. A naïve Bayes classifier is trained using *Rainbow* with the built-in Laplace smoothing method. No feature selection was employed (all features were used).

4.1.4 Tuning set and test set

Two datasets are used in our experiments, a sample of 12,000 web pages from ODP and a sample of 2000 web pages from Stanford's WebBase collection [11].

For ODP dataset, 1000 target pages are randomly sampled form each category. After that, the URLs of the neighboring pages are obtained and the pages are crawled from the Web. The outgoing links are directly extracted from the web pages, while the incoming links are obtained by querying Yahoo search with "inlink:" queries through the Yahoo API. Due to the daily access limit of the Yahoo API, if there are more than 50 incoming links for a page, only the first 50 incoming links are obtained.

Parent pages	518,309
Child pages	260,154
Sibling pages	4,917,296
Spousal pages	3,642,242

Table 2: Numbers of the four kinds of neighbors

On average, 778 neighbors are retrieved for each target page. The numbers of the four kinds of neighbors used in our test are listed in Table 2. Although we distinguish the four kinds of neighbors literally, they actually overlap with one another. Therefore, the actual total number of neighboring pages is less than the sum.

In order to tune the parameters, this target pages in ODP dataset are randomly divided into two halves: one half (500 pages per class) for tuning the parameters, the other half to be used for test (which is the real test set).

For the WebBase dataset, 2000 target pages are selected from the 2005 crawl. The link graph provided with the data collection is used to find the neighboring pages. The use of WebBase dataset has two purpose. First, the ODP pages are mostly high quality pages, while WebBase is a generic crawl from the Web. Therefore, the experiments on WebBase dataset are able to demonstrate the performance on generic web pages rather than just high-quality pages. Second, in ODP dataset, the number of neighboring pages being used is limited by the way of collecting incoming links. By using WebBase data, we hope to find out how important is the role played by the number of incoming links in our algorithm.

4.1.5 Removing "dmoz copies"

It is noteworthy to point out that when going through our data set manually, we find out there are plenty of "dmoz copies". A "dmoz copy" is simply a mirror of a portion of the dmoz ODP. Given that dmoz metadata is publicly available, setting up such a mirror site is straightforward, and not necessarily bad. However, our algorithm may unduly benefit from those copies.

Imagine a page pointed to by dmoz Directory is under scrutiny. By querying for the parents of this page, we may get several or even tens of dmoz copies which links to other pages with the same topic. Since the labels of those sibling pages are known (because they are in dmoz Directory), they are utilized by our algorithm in determining the target page's topic. Furthermore, the existence of "dmoz copies" provides multiple paths between the target page and the sibling pages. Therefore, in the weighted path version, the labels of those labeled sibling pages will probably dominate the contribution from the neighbors and thus boost the accuracy.

In order to minimize the benefit from "dmoz copies", we used a simple pruning method to remove the copies from our data set. The method is based on the observation that most URLs of "dmoz copies" contain the names of direc-

tories in dmoz, such as "Computer/Hardware" and "Business/Employment". This program checks the URLs of every neighboring page and removes the ones whose URL contains such directory names.

In ODP dataset, 136,844 pages were found by this pruning step. They are removed for all the experiments. This removal is necessary; a preliminary experiment shows a 3% drop of the accuracy after removal.

The limitation of this approach is obvious. This pruning method may unnecessarily remove pages that are not "dmoz copies". It may also pass by some real "dmoz copies" if they do not use those directory names in their URLs. However, a manual check on random sample of more than 100 parent pages did not discover any additional "dmoz copies".

4.2 Parameter tuning

In Section 3, we left several parameters in our algorithm to be tuned by experiments. In this section, we are going to show how the performance of the algorithm over our tuning set is affected by the value of these parameters.

4.2.1 η: bias on labeled pages

In order to find out what effect η has on the performance of the algorithm, we performed a series of tests by changing the value of η while fixing the values of other parameters. The other parameters are set as follows: θ=2.5, β={0.1, 0, 0.8, 0.1}, and α=0.2. As is shown in Figure 2, the best performance in these tests is achieved when η=0. As we increase the value of η, the accuracy shows a steady decreasing trend, highlighting the importance of human-labeled neighbors. The result suggests that we should fully trust the human labeling while totally ignoring the result of textual classifier. This is understandable given the poor performance of textual classifiers on Web data. As will be shortly, the accuracy of "Rainbow" is only 65% on the target pages.

4.2.2 θ: are internal links useful?

In the previous section, we raised the question: how useful are intra-host links in web page classification tasks? According to our findings, the answer is "yes".

A preliminary study shows that intra-host links are more informative than inter-host links. The experiment is done when η=0, β={0.1, 0, 0.8, 0.1}, α=0.2 and using weighted paths. When using only intra-host links, the classification accuracy is 89.8%. The accuracy drops to 70.6% when only inter-host links are considered.

Additional experiments are repeated to see how the accuracy changes when θ varies from 0 to 10. Figure 3 shows the test result. Although not remarkably, the weight of intra-host links does influence the accuracy observably. When increasing θ starting from 0, the accuracy climbs up steadily until getting to its peak when θ=2.5. After that, the accuracy fluctuates downwards. The result strongly suggests that the neighbors within the same host is more accordant in

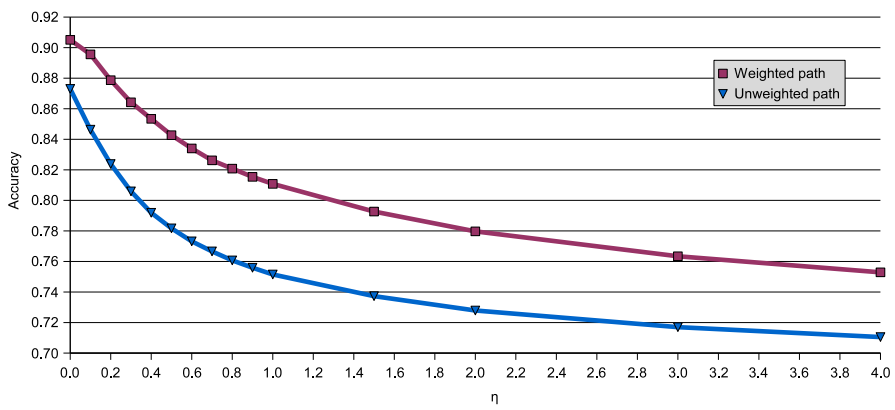


Figure 2: Accuracy vs. weight of unlabeled pages (η)

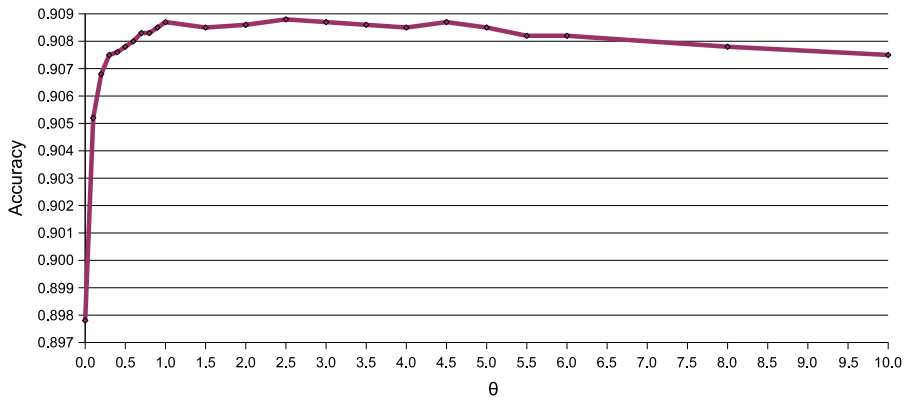


Figure 3: Accuracy vs. weight of intra-host links (θ)

topic with the target page than those from different hosts. Therefore, rather than being weighted less as in link-based ranking algorithms, the intra-host links should be given more weight than inter-host ones.

4.2.3 Weighted paths vs. unweighted paths

The comparison of the weighted and unweighted version is also shown in Figure 2, from which we can see that the weighted version outperforms the unweighted one. The explanation of this result is quite straightforward: having more parents (or children) in common implies a stronger connection between the two pages. Therefore, it is natural for the influence between them to be weighted.

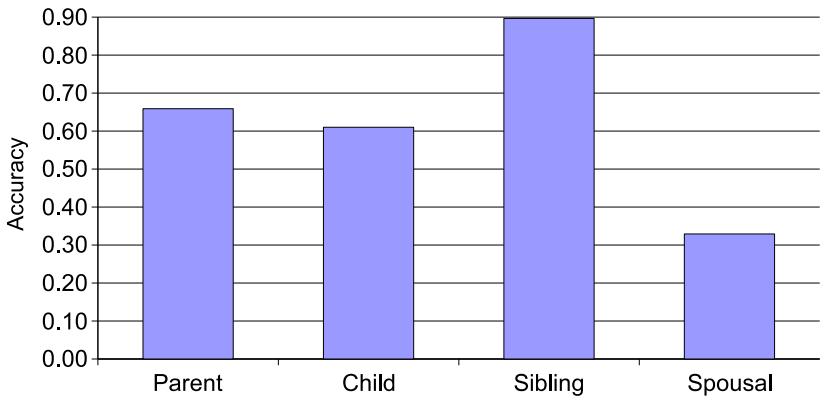


Figure 4: Individual contribution of four types of neighbors

4.2.4 β: weights among neighbors

We expect that different kinds of neighboring pages can have different contributions when deciding target page's topic. By having four separate types of neighboring pages, we can test and tune the impact that each type has on the classification task.

First, we study the individual impact of each kind of neighbors. Figure 4 shows the individual contribution of each of them, among which sibling pages contribute the most. Spousal pages are the least reliable source of information.

Next, in Figure 5, the influence of each kind of neighboring pages is augmented in contrast to the others. For example, in Group A, four tests are performed, each picking one kind of neighbors and setting the corresponding component in β to 0.4 while setting the other three to 0.2. In particular, the "parent" column in Group A shows the accuracy under the setting β= (0.4, 0.2, 0.2, 0.2). Similarly, in Group B, the major component is 0.7 and the remaining three are set to 0.1.

Figure 5 shows that having the sibling pages to make the major contribution is clearly better than the choices in any of the other three. However, does that mean we should give full weight to sibling pages?

In order to answer that question, we gradually change the weight of sibling pages from 0.3 to 1 and let the other three evenly share the remaining weight. The result is plotted in Figure 6. As we can see, although siblings are the best source of information, putting excessive weight on siblings will decrease the accuracy.

Note that β is a four-dimensional vector. The experiment above only explored the parameter space in one dimension, which is far from enough. However, an exhaustive search in a four-dimensional space is quite expensive. Experiments on around 500 different parameter settings have been conducted in order to find the global optimum. Some of the results are listed in Table 3.

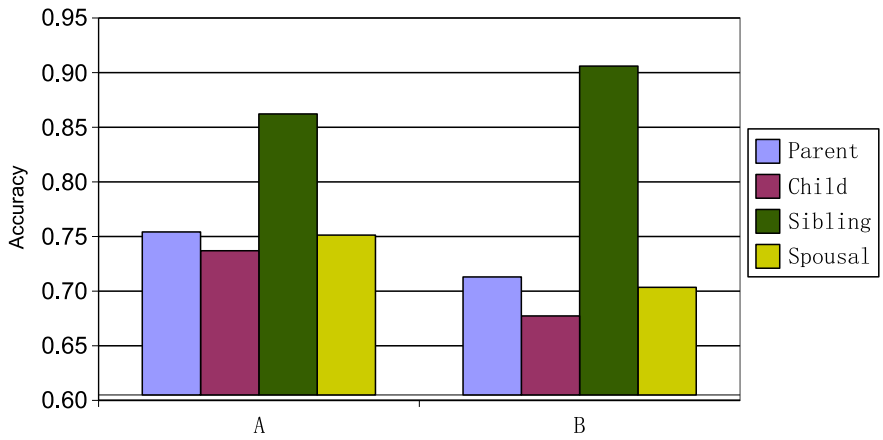


Figure 5: Accuracy as principal neighbor type is changed

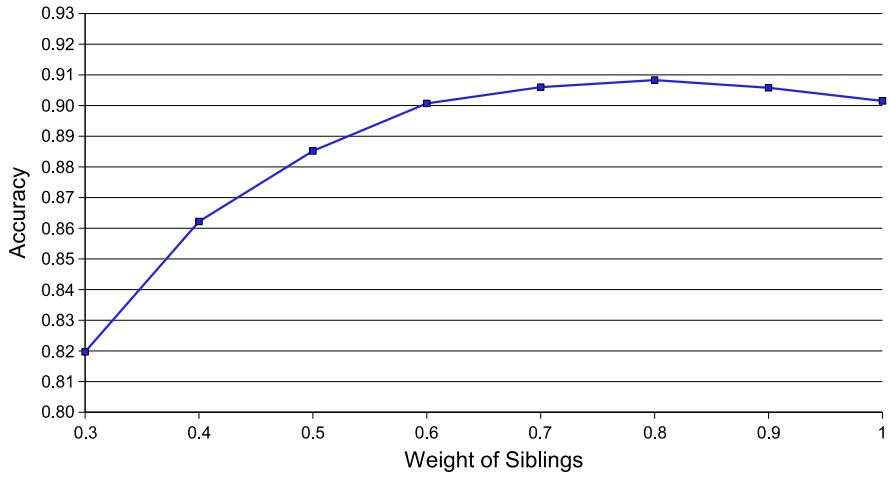


Figure 6: Accuracy vs. weight of siblings

4.2.5 α: combining the neighbors with the target page

We start by applying a textual classifier to the target page and try to correct the classifier's decision when the neighboring pages strongly indicate otherwise. As is shown in Figure 7, the accuracy gets to its peak when α=0.2, which means it is better to put more trust in the information from neighboring pages.

One may argue that it does not make sense that the neighboring pages are a better source of information than the page itself to help judge its topic. A reasonable explanation is as follows. As discussed before, our algorithm is based on an unsatisfying textual classifier. Given that we are utilizing the human labeling of the neighboring pages while hiding the labels of the target page, α is no longer simply a relative weighting between the target pages and the neighboring pages. It is a complex parameter which also indicates our trust in human labeling versus classifiers. As a result, our experiments do not imply that the contents of the target page are not important. It shows that, compared to our indirect way of determining topic based on the neighborhood, the classifier cannot make a reasonable decision on the target page's topic.

β_1	β_2	β_3	β_4	**Accuracy**
0.1	0	0.9	0	0.9055
0	0.1	0.9	0	0.9047
0	0	0.9	0.1	0.9055
0.2	0	0.8	0	0.9087
0	0.2	0.8	0	0.9048
0	0	0.8	0.2	0.9060
0.3	0	0.7	0	0.9050
0	0.3	0.7	0	0.8986
0	0	0.7	0.3	0.9022
0.4	0	0.6	0	0.8938
0	0.4	0.6	0	0.8808
0	0	0.6	0.4	0.8865
0	0.05	0.9	0.05	0.9058
0.05	0	0.9	0.05	0.9060
0.05	0.05	0.9	0	0.9055
0	0.1	0.8	0.1	0.9051
0.1	0	0.8	0.1	**0.9088**
0.1	0.1	0.8	0	0.9081
0	0.15	0.7	0.15	0.9033
0.15	0	0.7	0.15	0.9063
0.15	0.15	0.7	0	0.9048
0	0.2	0.6	0.2	0.8955
0.2	0	0.6	0.2	0.8978
0.2	0.2	0.6	0	0.8958

Table 3: Accuracy under different settings of beta

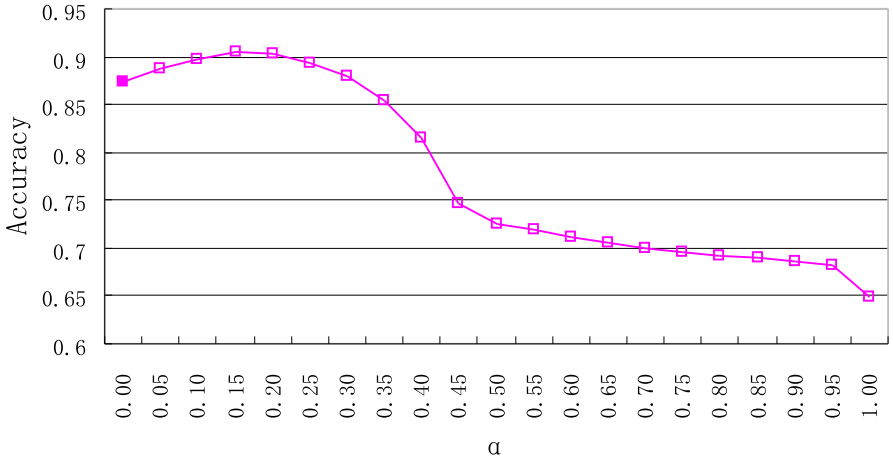


Figure 7: Accuracy vs. weight of target page content (α)

Algorithm	P-value on accuracy
Neighboring (svm)	9.14e-5
Neighboring (naive Bayes)	1.44e-3

Table 4: P-values for soft classification over hard classification

4.2.6 Soft classification vs. hard classification

In the previous section, we intuitively chose soft classification against hard classification. Now it is time to verify that intuition.

Rainbow performs soft classification by default: it predicts the probability for each page being in every category. We convert it to hard classification by taking the soft classification result and labeling each page with the category corresponding to the largest component.

We compared the accuracy of our algorithm when performing it on both soft and hard classification results, with the parameter setting being as follows: α=0.2, β=(0.1, 0, 0.8, 0.1), η=0, θ=2.5 and the weighted path version. The accuracies are 90.5% and 90.3%, respectively, with the soft classification in the lead. Similar experiments are done based on the result of SVM^{light}. The accuracies are 91.4% and 91.2%. Although the difference is not large, soft classification did do a slightly better job than hard classification.

In addition, a t-test is performed on the accuracy of soft classification and hard classification. The result (shown in Table 4) suggests that soft classification performs statistically significantly better than hard classification.

	$K+C$	$K+C$ (link only)	IO_Bridge
Neighboring (naive Bayes, soft)	2.71e-9	5.28e-5	2.33e-9
Neighboring (svm, soft)	5.79e-9	4.35e-5	4.78e-9

Table 5: P-values for Neighboring over other algorithms

4.3 Experimental results

4.3.1 Experiments on ODP dataset

After tuning the parameter settings, we ran our algorithm on the ODP test set with the setting, α=0.2, β=(0.1, 0, 0.8, 0.1), η=0, θ=2.5 and using the weighted path version.

For the purpose of comparison, we also implemented one of the algorithms (*IO-bridge*) suggested by Chakrabarti et al. [4] and the algorithm proposed by Calado et al. [2].

The main idea of *IO-bridge* is to build an engineered document corresponding to each document in the dataset, in which there are simply prefixes of the category names of the sibling pages of the target page. In *IO-bridge*, only the sibling pages within a human labeled dataset are considered. After that, the training and testing is performed on the engineered dataset rather than the original one.

The best performance reported by Calado et al. was achieved when combining the result of the kNN text classifier with co-citation similarity derived from link graph (*K+C*). In the following, we compare our algorithm with both *IO-bridge* and *K+C*.

We trained IO-bridge on the same 228,000 document as is used to train *Rainbow*, and tested it on the same test set as we used to test our algorithm. The comparison is shown in Figure 8. The baseline is the accuracy of *Rainbow*, the textual classifier, which is 65% on the test set. kNN, with k=30, performs almost as well as *Rainbow*. SVM^{light} did a better job than *Rainbow*, with its accuracy being 73.1%. IO-bridge increases the accuracy to 79.6%. (IO-bridge is reported in that paper to have increased the accuracy from 32% to 75% on a 800-document dataset extracted from Yahoo Directory.) $K+C$ has an accuracy of 76.3%. However, if only the link-based measure (co-citation) is considered, its accuracy is much higher (87.1%) than the combination of link-based and content-based measures. Our algorithm (referred to as "Neighboring" in Figure 8), at the best performance, can achieve 91.4% accuracy on this particular data set. Further more, a series of t-test shows $Neighboring$ algorithm performs significantly better than other algorithms.

Although the accuracy of SVM^{light} is nearly 10% better than that of Rainbow, Neighboring algorithm based on SVM^{light} outperforms the one based on Rainbow for only 1%, which weakly implies that our Neighboring algorithm does not rely much on the textual classifier.

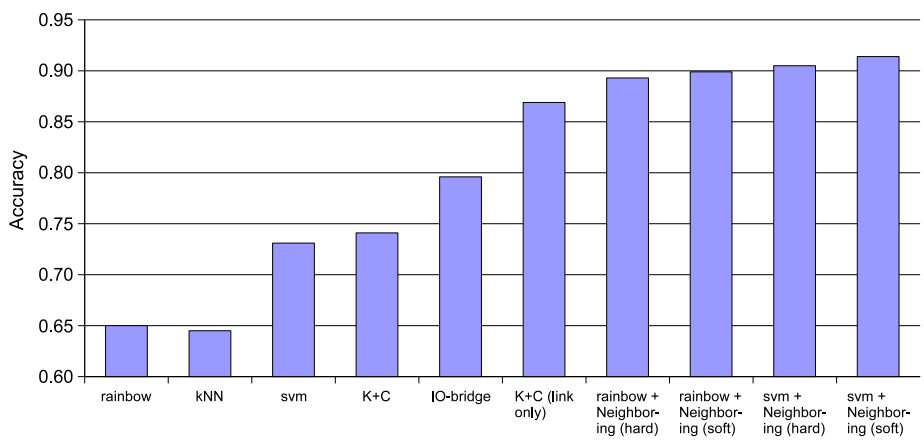


Figure 8: Comparison of accuracy of different algorithms

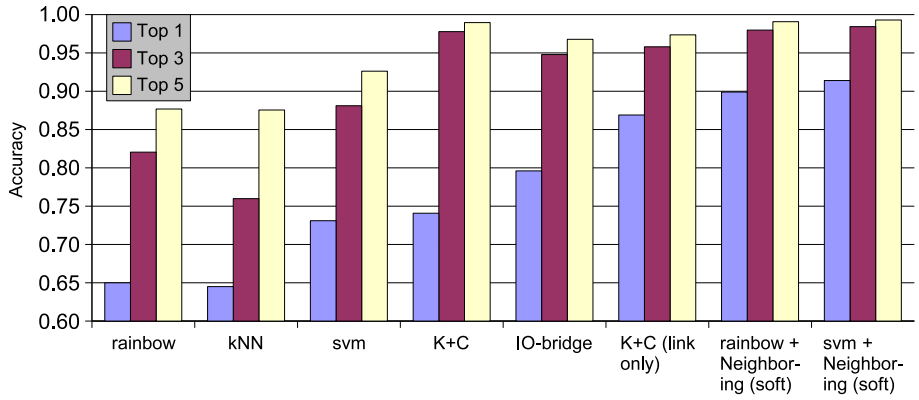


Figure 9: Accuracy on top 3 and top 5 categories

We also calculated accuracy on top 3 and top 5 categories. In these cases, the classes are sorted by the predicted probabilities. If one of the top 3 or top 5 categories matches the label given by human, the prediction is considered correct. As is plotted in Figure 9, *Neighboring* algorithm based on SVM^{light} got a 98.43% high accuracy when looking at top 3 categories, 99.3% at top 5.

Our *Neighboring* algorithm takes advantage from utilizing human labeling. However, this is also one of its major limitations. In our dataset, each target page, on average, has 92 labeled page out of 2,043 pages in its neighborhood. In the real world, we cannot expect such a good number of labeled pages. For this reason, we want to find out how *Neighboring* algorithm performs if there is no human labeling available. We hid the labels of neighboring pages (i.e., as if they do not have labels), and ran *Neighboring* algorithm again based on Rainbow

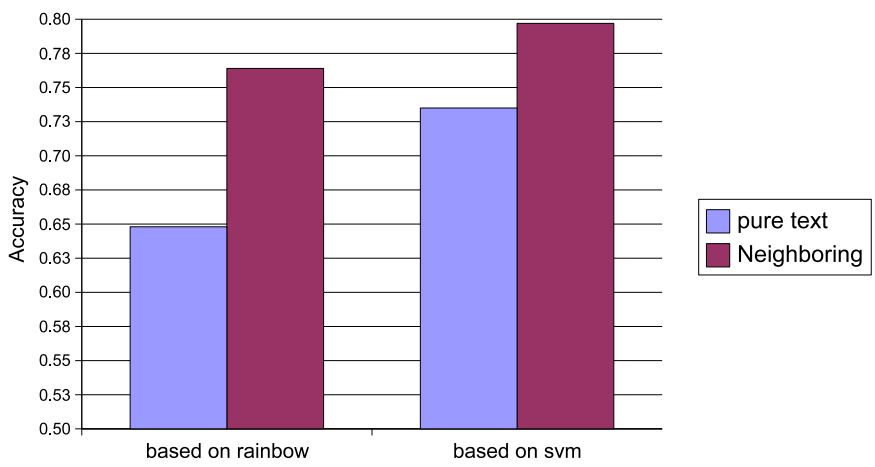


Figure 10: Neighboring algorithm without using human labeling

and SVM^{light}, with the parameters α=0.2, β=(0.1, 0, 0.8, 0.1), η=1, θ=2.5 and using the weighted path version. The result is shown in Figure 10. As we can see, although *Neighboring* algorithm suffered from the lack of labeling, it still has a fairly good performance.

4.3.2 Experiments on WebBase dataset

To eliminate the potential bias of highly-regarded web pages from the ODP, we tested our approach on randomly selected pages from the WebBase dataset. We continued to use the parameter settings of α=0.2, β=(0.1, 0, 0.8, 0.1), η=1, θ=2.5 and using the weighted path version. We manually labeled 118 randomly selected pages for evaluation purposes. The accuracy of our Neighboring algorithm is 58%, reducing the error rate by almost one third compared to a naive Bayes textual classifier (40.7%).

We also used WebBase to explore how sensitive our approach was to the number of incoming links. We ordered the incoming links by their PageRank [20] value and selected the top 50 as the initial incoming links, to best match the expected performance of the Yahoo in-link service. However, when we increased the number of incoming links used to 100, 200, etc., even all incoming links present in the dataset, we found no significant change in performance. Thus we conclude that increasing the number of incoming links is not likely to affect classification accuracy.

5 Discussion

This paper has shown the improvements that our Neighboring algorithm can provide for web page classification. However, this approach also has some limi-

tations, which we discuss here.

- While performance is a function of a number of tunable parameters, we have not fully explored the parameter space, and it remains to be seen whether the parameter choices are independent of the data set utilized.
- The ODP dataset used for most of our experiences generally consists of highly-regarded pages. Our experiment with WebBase data suggests that performance on the ODP dataset may be higher than arbitrary web pages. This effect might be mitigated by using training data that better matches the test data (e.g., training on random web pages).
- We only utilized the neighboring information to help determine the target page's topic. The classification of the target page itself, however, may similarly affect the neighboring pages' topic. A relaxation technique (e.g., as used in another algorithm from [4]) might be a useful addition to our approach.
- For simplicity, the classification showed in this paper is only on the first-level categories of dmoz Directory. Conducting similar classification at a deeper level, or on more fine-grained topics, may expose more interesting facts.

6 Summary

This paper has explored a method to utilize class information from neighboring pages to help judge the topic of a target web page. The experimental results show that, under appropriate parameter settings, our algorithm significantly outperforms the *Rainbow* and SVM^{light} textual classifiers as well as existing web-specific approaches.

Our contributions include the following:

- We tested multiple algorithms on a large, real-world data set.
- We greatly improved the accuracy of web page classification, reducing error rates by about two thirds over common text classification approaches.
- We showed that improvements using our approach were realized even when neighboring pages were unlabeled.
- We explored the effects that a number of factors have on the classification, and proposed explanations of our findings. We found that intra-host links are important in web classification and that sibling pages give a good indication on a page's topic.
- We pointed out the "dmoz copy effect" in web page classification and proposed a way to address it (although this has been raised as a general issue in web link analysis).

In the future, we plan to combine on-page features and neighboring class information with off-page anchor texts (e.g., as in [9]) to improve accuracy further.

Acknowledgments

We thank Baoning Wu, Lan Nie and Vinay Goel for helpful discussions. This material is based upon work supported by the National Science Foundation under Grant No. 0328825.

References

[1] G. Attardi, A. Gulli, and F. Sebastiani. Automatic web page categorization by link and context analysis. In C. Hutchison and G. Lanzarone, editors, *Proceedings of THAI'99, European Symposium on Telematics, Hypermedia and Artificial Intelligence*, pages 105–119, Varese, IT, 1999.

[2] P. Calado, M. Cristo, E. Moura, N. Ziviani, B. Ribeiro-Neto, and M. A. Goncalves. Combining link-based and content-based methods for web document classification. In *Proceedings of the 12th International Conference on Information and Knowledge Management*, 2003.

[3] S. Chakrabarti. *Mining the Web: Discovering Knowledge from Hypertext Data*. Morgan Kaufmann, San Francisco, CA, 2003.

[4] S. Chakrabarti, B. E. Dom, and P. Indyk. Enhanced hypertext categorization using hyperlinks. In *Proceedings of ACM SIGMOD*, Seattle, WA, 1998.

[5] S. Chakrabarti, M. M. Joshi, K. Punera, and D. M. Pennock. The structure of broad topics on the web. In *Proceedings of the Eleventh International World Wide Web Conference*, pages 251–262, Honolulu, May 2002.

[6] S. Chakrabarti, M. van den Berg, and B. E. Dom. Focused crawling: A new approach to topic-specific Web resource discovery. In *Proceedings of the Eighth International World Wide Web Conference*, pages 545–562, Toronto, Canada, May 1999.

[7] C. Chekuri, M. H. Goldwasser, P. Raghavan, and E. Upfal. Web search using automatic classification. In *Proceedings of the 6th International World Wide Web Conference*, 1996.

[8] S. T. Dumais and H. Chen. Hierarchical classification of web content. In *Proceedings of the 23rd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, 2000.

[9] E. J. Glover, K. Tsioutsiouliklis, S. Lawrence, D. M. Pennock, and G. W. Flake. Using web structure for classifying and describing web pages. In

Proceedings of the Eleventh International Conference on World Wide Web, 2002.

[10] T. H. Haveliwala. Topic-sensitive PageRank. In *Proceedings of the Eleventh International World Wide Web Conference*, Honolulu, Hawaii, May 2002.

[11] J. Hirai, S. Raghavan, H. Garcia-Molina, and A. Paepcke. WebBase: a repository of Web pages. *Computer Networks*, 33(1–6):277–293, 2000.

[12] N. Holden and A. A. Freitas. Web page classification with an ant colony algorithm. In *Parallel Problem Solving from Nature - PPSN VIII, LNCS 3242*, pages 1092–1102. Springer-Verlag, Sept. 2004.

[13] T. Joachims. Making large-scale support vector machine learning practical. In B. Schölkopf, C. Burges, and A. Smola, editors, *Advances in Kernel Methods: Support Vector Machines*. MIT Press, Cambridge, MA, 1998.

[14] M. Kan. Web page classification without the web page. In *Proceeding of 13th WWW Conference, poster*, 2004.

[15] M.-Y. Kan and H. O. N. Thi. Fast webpage classification using URL features. In *Proceedings of the 14th ACM International Conference on Information and Knowledge Management (CIKM '05)*, pages 325–326, New York, NY, USA, 2005. ACM Press.

[16] M. Kovacevic, M. Diligenti, M. Gori, and V. Milutinovic. Visual adjacency multigraphs - a novel approach to web page classification. In *Proceedings of SAWM04 workshop, ECML2004*, 2004.

[17] A. K. McCallum. Bow: A toolkit for statistical language modeling, text retrieval, classification and clustering. http://www.cs.umass.edu/~mccallum/bow/, 1996.

[18] L. Nie, B. D. Davison, and X. Qi. Topical link analysis for web search. In *Proceedings of the 29th Annual International ACM SIGIR Conference on Research & Development on Information Retrieval*, Seattle, Aug. 2006.

[19] Open Directory Project (ODP), 2006. `http://www.dmoz.com/`.

[20] L. Page, S. Brin, R. Motwani, and T. Winograd. The PageRank citation ranking: Bringing order to the Web. Unpublished draft, 1998.

[21] M. Richardson and P. Domingos. The Intelligent Surfer: Probabilistic Combination of Link and Content Information in PageRank. In *Advances in Neural Information Processing Systems 14*. MIT Press, 2002.

[22] D. Shen, Z. Chen, H.-J. Zeng, B. Zhang, Q. Yang, W.-Y. Ma, and Y. Lu. Web-page classification through summarization. In *Proceedings of the 27th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, 2004.

[23] S. Slattery and T. Mitchell. Discovering test set regularities in relational domains. In *Seventeenth International Conference on Machine Learning*, 2000.

[24] W. Wibowo and H. E. Williams. Strategies for minimising errors in hierarchical web categorisation. In *Proceedings of the Eleventh International Conference on Information and Knowledge Management (CIKM '02)*, pages 525–531, New York, NY, USA, 2002. ACM Press.

[25] B. Wu, V. Goel, and B. D. Davison. Topical TrustRank: Using topicality to combat web spam. In *Proceedings of the 15th International World Wide Web Conference*, pages 63–72, Edinburgh, Scotland, May 2006.

[26] Yahoo!, Inc. Yahoo! `http://www.yahoo.com/`, 2006.

[27] H. Yu, J. Han, and K. C. Chang. Pebl: Web page classification without negative examples. *IEEE Trans. on Knowledge and Data Engineering*, 16:70–81, 2004.