Classifiers Without Borders: Incorporating Fielded Text From Neighboring Web Pages

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

ABSTRACT

Accurate web page classification often depends crucially on information gained from neighboring pages in the local web graph. Prior work has exploited the class labels of nearby pages to improve performance. In contrast, in this work we utilize a weighted combination of the contents of neighbors to generate a better virtual document for classification. In addition, we break pages into fields, finding that a weighted combination of text from the target and fields of neighboring pages is able to reduce classification error by more than a third. We demonstrate performance on a large dataset of pages from the Open Directory Project and validate the approach using pages from a crawl from the Stanford WebBase. Interestingly, we find no value in anchor text and unexpected value in page titles (and especially titles of parent pages) in the virtual document.

Categories and Subject Descriptors

H.3.3 [**Information Storage and Retrieval**]: Information Search and Retrieval—*Information filtering*; I.5.2 [**Pattern Recognition**]: Design Methodology—*Classifier design and evaluation*; H.3.1 [**Information Storage and Retrieval**]: Content Analysis and Indexing

General Terms

Algorithms, Performance

Keywords

Web page classification, SVM, naive Bayes, Neighboring

1. INTRODUCTION

Web page classification is the process of assigning a web page to one or more predefined category labels. Classification is often posed as a supervised learning problem in which a set of labeled data is used to train a classifier which can be applied to label future examples.

Web page classification helps retrieval and management of web information in various respects. Many tasks may benefit from accurate categorization of web pages, such as the development of web directories like those provided by Yahoo![1] and the dmoz Open Directory Project (ODP)[2], topic-sensitive web link analysis [17, 20, 24], contextual advertising, analysis of the topical structure of the Web [7], and focused crawling [8].

One baseline approach to web page classification is to treat web pages as text documents without considering the additional features that the web can provide (e.g., hyperlinks, markups). Such an approach usually yields suboptimal performance. In a preliminary experiment in which we applied support vector machines to a 3,600-page ODP dataset, merely 77% of the pages were correctly classified into one of the twelve broad categories when only on-page textual features are used.

Research has been widely conducted exploring the usefulness of non-textual features (sometimes in addition to textual features) in web page classification. A number of studies have shown that web page classification can be performed more accurately using information gathered from neighboring pages in the web graph [6, 15, 4, 23]. However, when using content of neighbors, existing work typically uses information from neighboring pages as a whole. Inspired by the success of the fielded extension of BM25 in information retrieval [25], we conjecture that permitting text from different fields of neighboring pages to contribute differently may improve classification performance. Our intuition is that information from different parts of neighboring pages should have different importance. For example, anchor text, considered a concise description of the target page, should be more important than generic text.

In this paper, we propose the F-Neighbor algorithm, as a fielded extension to our Neighboring algorithm [23] which used class information and full content of neighboring pages. In particular, we break up neighboring web pages, as well as the page to be classified into several text fields (e.g., title, anchor text), and combine them according to the individual importance they have. Our experiments show that this method is able to reduce the error rate of textual classifiers by more than half. We also found that page titles, especially parent titles, are more useful than generic text.

The rest of this paper is organized as follows. In Section 2, we review related work exploiting hyperlink information to enhance web page classification. In Section 3, we introduce and describe F-Neighbor, the proposed algorithm. We tune and test this algorithm in Section 4, and conclude with a discussion and a summary of our results.

SIGIR'08, July 20–24, 2008, Singapore.
Copyright 2008 ACM 978-1-60558-164-4/08/07 ...$5.00.

[1] http://www.yahoo.com/
[2] http://www.dmoz.org/

2. BACKGROUND

2.1 Related work on fielded model

In 1994, Robertson and Walker [26] proposed a function to rank documents based on the appearance of query terms in those documents, which was later referred to as Okapi BM25. In time, BM25 became a common component of information retrieval systems. Later, Robertson et al. [25] extended this model to combine multiple text fields including anchor text, and showed improvement over original BM25. Currently the fielded BM25 model (BM25F) is widely used, taking the place of its non-fielded predecessor. Besides retrieval, the fielded model has also shown to be useful in expert finding from email corpora [3, 22]. In this paper, we borrow the idea that extended BM25 to BM25F, hoping it can boost classification performance as it did for retrieval.

2.2 Related work on web classification

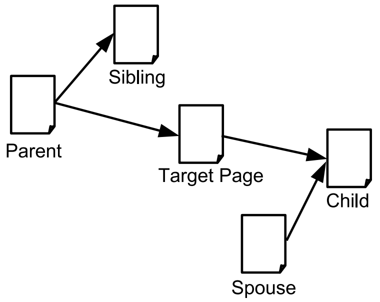
Categorization of web pages has been an active research area. Recently, many studies have focused on using information from neighboring web pages. Exploiting the topical locality of the web [11, 7], this type of approach gathers information from neighboring pages on a local link graph to help classify the page in question.

Many kinds of information from neighbors can be used in classification. Some approaches benefit from taking advantage of neighboring pages that have been labeled by human experts [6, 28, 4, 23]. However, if an approach exclusively relies on such labels, its applicability is restricted due the limited number of labeled pages on the web.

The content of neighboring pages is also useful. In general, directly incorporating text from neighboring pages introduces more noise than signal [6, 14, 33]. Such noise can be greatly reduced by separating local text and text from neighbors [23], by also considering the human-assigned labels of neighbors [21, 23], or by using only important parts of content from neighbors, such as titles, anchor text, and the surrounding text of anchor text [2, 12, 13, 33, 30, 10, 15, 31, 34]. In particular, Attardi et al. [2] showed promising results using title, anchor text, and a portion of text surrounding the anchor text on parent pages to help determine the target page's topic. Similarly, Fürnkranz [12] proposed an approach to classification using features on the parent pages, such like anchor text, surrounding text of anchor text and the headings that precede the hyperlink. A graph-based approach to classification was proposed by Angelova and Weikum [1] based on relaxation labeling, in which labels of pages in a weighted local link graph are iteratively adjusted. The use of information from neighbors is not restricted to topical classification; it can also be applied in spam classification as proposed by Castillo et al. [5], as well as document representation refinement for the purpose of retrieval [29].

The following points separate our current work from the previous:

- Existing work either considers text of neighbors as a whole without differentiation, or only considers part of the text while ignoring the rest. We argue that although some text may bear higher importance, other text sources may also contribute.
- Most existing work that uses partial text of neighbors only considers features from parent pages. However, other studies show that sibling pages contain strong signals for the topic of the target page [6, 28, 23]. Therefore, utilizing text fields from other neighbor types may additionally improve classification.



Parent pages	$\{\, q \mid q \rightarrow p \; and \; q \neq p \,\}$
Child pages	$\{\, q \mid p \rightarrow q \; and \; q \neq p \,\}$
Sibling pages	$\{\, q \mid \exists\, r \; s.t. \; r \rightarrow p, \; r \rightarrow \; q \; and \; q \neq p \,\}$
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.

2.3 Neighboring algorithm

Before introducing our proposed algorithm, we briefly summarize our earlier Neighboring algorithm [23] as it is the basis of the work described in this paper.

The basic idea of Neighboring algorithm is the use of pages nearby in the link graph (called "neighboring pages") to help classify the page in question (called "target page"). As illustrated in Figure 1 from [23], four types of neighboring pages are used: parent, child, sibling and spouse.

The Neighboring algorithm, as shown in Figure 2, consists of four steps: page level weighting, grouping, group level weighting, and the weighting between the target page and neighbors. First, each neighboring page, represented by its topic vector, is weighted according to its individual properties. Such properties include whether its human-generated label is available, whether it is from the same host as the target page, etc. Then, these pages are assembled into four groups according to their link relationship with the target page, that is, parent, child, sibling, or spouse. At this step, the topic vectors of the pages within the same group are aggregated. The topic vector of a group is computed as the centroid of the topic vectors of all its members. After grouping, there are four topic vectors, one for each group. Then, group level weighting combines the four vectors and generates the topic vector of all the neighbors. Finally, the topic vector of neighbors and the topic vector of the target page are combined to generate the final topic vector, based on which a prediction of the page's topic is made. The parameters used in these weighting schemes are tuned through experiments.

The intuition of the Neighboring algorithm is to utilize the information from neighbors to adjust the classifier's view of the target page. For example, consider the scenario illustrated in Figure 2. The target page was originally classified as "yellow" by a classifier that only considers the page itself. However, the neighbors strongly suggest otherwise. Therefore, by considering the information of neighbors, the Neighboring algorithm is able to adjust its decision and classify the target page as "red".

Through studying the effect of multiple factors in web classification, our earlier work found that human-generated labels of neighbors should be used whenever they are available; neighbors within the same site as the target page are useful; a sibling (or a spouse) should be weighted by the number of parents (or children) it has in common with the target page; and finally, while all the four neighbor types can contribute, sibling pages are the most important. Those experiments showed an approximately two thirds reduction

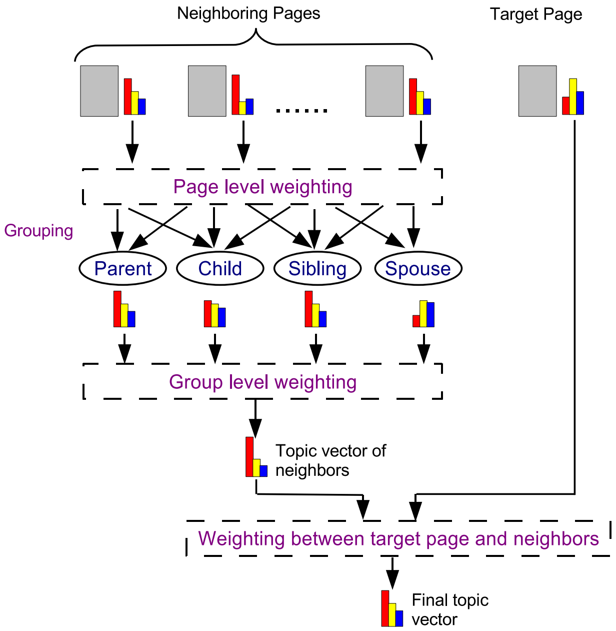


Figure 2: Neighboring algorithm overview

in error rate compared with support vector machines operating on target text only. In Section 4 we reproduce these experiments on revised datasets.

3. UTILIZING FIELD INFORMATION OF NEIGHBORS

The default Neighboring algorithm considers text on each neighboring page as a whole, disregarding where the text appears. Here, we argue that text appearing in different fields may carry different values. For example, anchor text (the text to be clicked on to activate and follow a hyperlink to another web page, placed between HTML <A> and </A> tags) is usually considered a good description of the page to which it points; therefore, anchor text could be more useful in classification than other text on the parent page. As an extension to the Neighboring algorithm, we examine the importance of text in different fields on neighboring pages.

3.1 Utilizing text fields

The idea of differentiating text in different fields of hypertext is not new. It has been successfully applied to web information retrieval [25], to web classification [16], as well as other research. However, little research has examined the importance of text fields on neighboring pages in classification problems. In this paper, we propose to utilize the textual information appearing in the following fields to help classify a web page:

- title of the target page;
- full text of the target page;
- titles of parent, child, sibling, and spouse pages;
- full text of parent, child, sibling, and spouse pages;
- anchor text (referring to target) on parent pages;
- surrounding text of anchor text (including anchor text itself) on parent pages (referred to as "extended anchor text", which in our experiments consists of the 50 characters before and after the anchor text).

For each page that needs be classified, these fields are extracted. Unlike fields residing on the target page, each type of the text fields from neighboring pages usually has multiple instances. For example, a target page with ten sibling pages has ten instances of "sibling:title" field. These instances are aggregated by computing the centroid of each type of field, as described below.

3.2 Text representation

We formalize the computations described above using the following equations. For the fields on the target page, a tfidf representation is computed for each virtual document using the equations used by the Cornell SMART system [27]. The term frequency and inverse document frequency are defined by Equation 1 and 2, where $n(d,t)$ is the number of times term t appears in document d, $|D|$ is the the total number of documents in the collection D, and $|D_t|$ is the number of documents that contain term t.

$$TF(d,t) = \begin{cases} 0 & \text{if } n(d,t) = 0 \\ 1 + log(1 + log(n(d,t))) & \text{otherwise} \end{cases} \qquad (1)$$

$$IDF(t) = log\frac{1+|D|}{|D_t|} \qquad (2)$$

Each document d is represented by a vector $\vec{d}$ in which each component d_t is its projection on axis t, given by

$$d_t = TF(d,t) \times IDF(t) \qquad (3)$$

Finally, vector $\vec{d}$ is normalized using Equation 4 so that the length of the vector is 1.

$$d'_t = \frac{d_t}{\sqrt{\sum_{s \in T} d_s^2}} \qquad (4)$$

The vector $\vec{d'}$ computed by Equation 4 is used to represent a field from the target page.

Equations 1 through 4 are also applied to fields of neighboring pages. However, in contrast to the fields on the target page, each type of neighboring field usually has multiple instances coming from different neighbors (as in the previous example, a target page with ten siblings has ten instances of sibling title and ten instances of sibling text). In order to generate a single representation for each type of neighboring field, an additional step is needed. This is performed by simply computing the centroid of the instances of the type. Empty fields (such as an empty title) are not considered in this computation. Note that the tfidf vectors are normalized before combination to prevent a long textual field of one page from dominating the fields from other pages.

$$\vec{d_{f_i}} = \begin{cases} \vec{d'} & \text{if } f_i \text{ is a field on target} \\ \frac{1}{N_{f_i}} \sum_{j=1}^{N_{f_i}} \vec{d'}_j & \text{if } f_i \text{ is a field of neighbors} \end{cases} \qquad (5)$$

Now for each target document we have computed twelve vectors ($\vec{d_{f_1}}$ through $\vec{d_{f_{12}}}$) representing the various text fields. We will combine them by weighted sum as in Equation 6 to form a single vector on which the classifier is performed.

$$\vec{d}_{comb} = \sum_{i=1}^{12} w_{f_i} * \vec{d_{f_i}} \quad \text{where } 1 = \sum_{i=1}^{12} w_{f_i} \qquad (6)$$

The vector $\vec{d}_{comb}$ is used as the document representation in F-Neighbor classification. The weights w_{f_i} in Equation 6 will be determined experimentally.

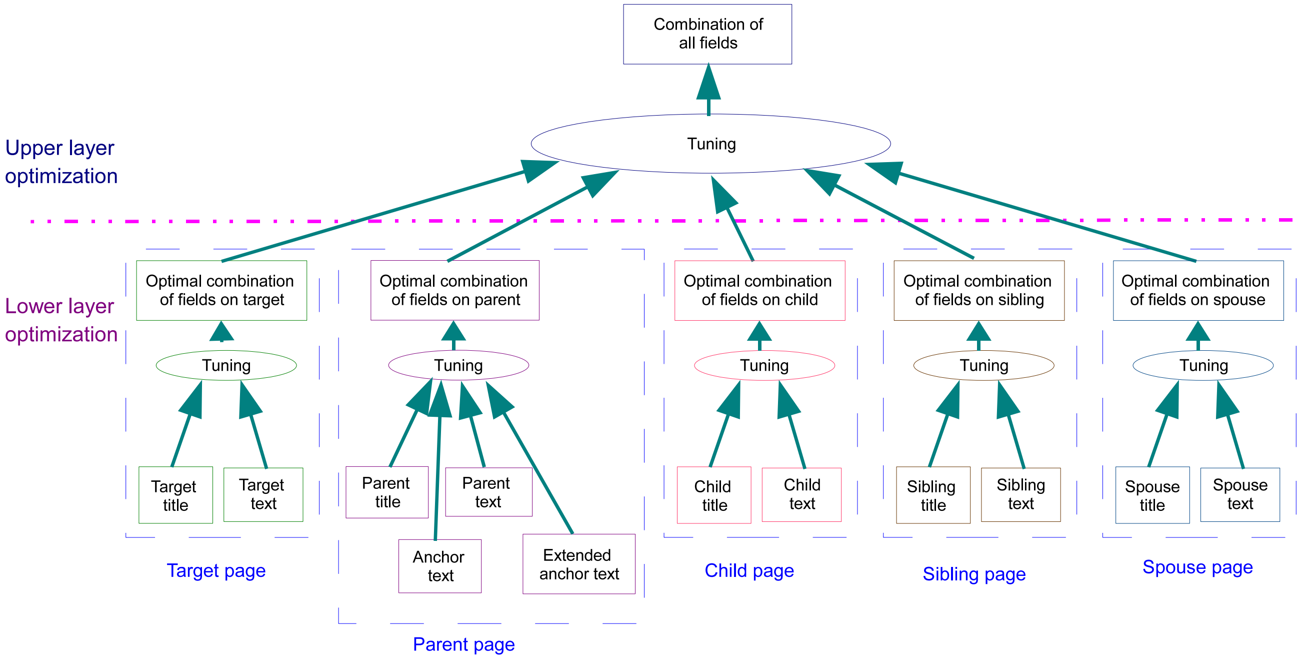


Figure 3: The process of two layer optimization.

3.3 Parameter tuning

The determination of the weights w_{f_i} in Equation 6 can be seen as an optimization problem in which the classification accuracy based on the aggregated representation is the target to be optimized. Therefore, any generic optimization technique can be applied. In this work, we used a two-layer optimization approach, in which the lower layer optimization optimizes the combination among fields on the same neighbor type (e.g., child title and child text), while the upper layer optimizes among all the neighbors based on the result of the lower layer optimization. Figure 3 illustrates this optimization process.

4. EXPERIMENTS

So far, we have described a parameterized model for web classification. In this section, we tune and test it on real-world datasets.

4.1 Datasets

Two datasets are used in our experiments to measure performance: a sample of 12,000 web pages from ODP and a sample of 2,000 web pages from the Stanford WebBase collection [9]. The ODP metadata being used 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 retrieved.

For the ODP dataset, as in the work by Chakrabarti et al. [6], 12 out of the 17 top-level categories of the dmoz Directory were selected, and we list them in Table 1. The 12,000 pages are sampled only from these 12 categories, 1,000 per category. From each category, 700 target pages are randomly sampled to tune the parameters and another 300 for testing the algorithms. The URLs of the neighboring pages are obtained and then the union of those pages is 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[3]. Due to API usage limits, at most the first 50 incoming links for each page were obtained.

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

Table 1: Set of twelve top-level categories used from the dmoz Directory.

For the WebBase dataset, 2000 target pages are selected from a 2005 crawl. The link graph provided with the data collection is used to find the neighboring pages. The purpose of using WebBase dataset is to offset the bias of the ODP dataset. The ODP pages are mostly high quality pages, while WebBase is a generic crawl from the Web. Therefore, experiments on the WebBase dataset are potentially able to demonstrate performance on more typical web pages rather than just high-quality pages.

4.2 Preprocessing and feature selection

All of the web pages in our experiments have gone through the same text preprocessor as in [23]. The functionality of the preprocessor is as follows:

- eliminate HTML tags except the content from the “keywords” and “description” metatags (because they may be of help in deciding a page’s topic);
- decode HTML character entity references (e.g., “&nbsp;”, “&amp;”);
- replace non-alphanumeric characters with spaces;
- eliminate terms whose length exceeds a certain limit (4096 characters in this case); and,
- eliminate useless titles and anchor text (e.g., “home page”, “untitled document”, “here”, “click here”) so that such fields

[3] http://developer.yahoo.com/

become empty and are excluded from the document representation.

After preprocessing, each HTML file is transformed into a stream of terms. The preprocessing increases classification accuracy by a marginal improvement and reduces time and space required by the classifier.

Our prior work pointed out that "dmoz copies" in the experiment dataset may produce over-optimistic estimation of classification performance [23]. A "dmoz copy" is defined as a mirror of a portion of the ODP. The existence of "dmoz copies" creates extra connections between the target page and its sibling pages. Therefore, it may benefit algorithms which utilize such connections. In that paper, we also proposed a simple heuristic approach to prune "dmoz copies", which matches page URLs against ODP category names. A careful check through the dataset finds that there are still "dmoz copies" after applying such an approach. In this work, we extended this pruning approach one step further: to match page *titles* as well as URLs against ODP category names. This extended approach found 160,868 "dmoz copy" pages from the ODP dataset, 24,024 (18%) more than the previous approach.

After the preprocessing described above, feature selection is performed based on mutual information (MI). Note that the same term may have different importance (different discriminating power) in different text fields. Therefore, rather than computing a global mutual information for each term, we calculated MI on a per field basis. For each field, 5000 terms with highest MI are selected. The final set of features is the union of the top 5000 features selected for each field. On our ODP dataset, 11,395 features are selected out of a vocabulary of 14 million terms.

4.3 Classifier

A linear kernel support vector machine with default settings based on SVM^{light} [18] is used as the base classifier in our experiments. Since SVM is a binary classifier, we generalize it using one-against-others approach, i.e., to train twelve classifiers each using one class as the positive class and the rest as the negative class. For each parameter combination we examined in the tuning process, a two-fold cross validation is performed. The reported classification accuracy is the average across the two folds.

4.4 Lower layer optimization

At lower layer optimization, fields of each neighbor type are combined independently to achieve the best accuracy within that neighbor type. The questions are: what is the best combination that can be gained within each neighbor type; and for parents, how much is the benefit of emphasizing anchor text over other text.

We start by showing the benefit of emphasizing titles by investigating the combinations of title and text within each neighbor type (and target page itself), where the weight of title and the weight of text add up to one. Figure 4 shows the result. The x-axis is the weight of the title. A weight of 0 means classification over the whole content, without emphasizing the title. A weight of 1 means classification using only the title. The results show that although there is marginal benefit in emphasizing titles of the target and siblings, other neighbor types benefit noticeably (with a top relative improvement of more than 14%).

We continue by examining the usefulness of anchor text and extended anchor text. However, unlike previous work [12, 15], the result is not encouraging. We combined target text with one of anchor text, extended anchor text, and parent title. The result is shown in Figure 5. Although compared with only using target text alone (74.8% accuracy), there is some benefit when target text is combined with anchor text (75.9%) or with extended anchor text (76.3%). However, combining parent title with target text provides much better improvement (raising accuracy to 84.3%).

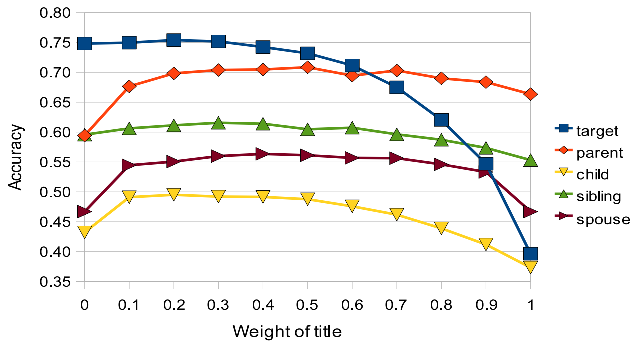


Figure 4: The tuning of title and text.

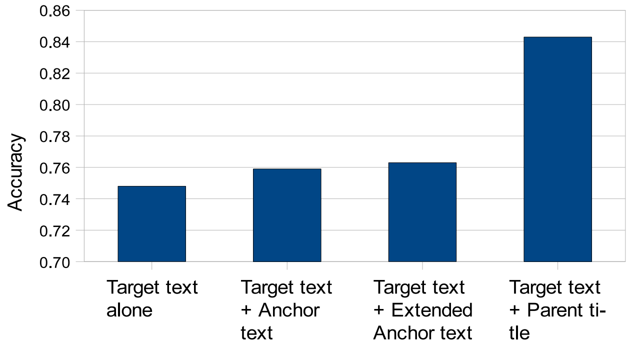


Figure 5: Combining target text with anchor, extended anchor, or parent title.

We also combined the four fields on parent page together, i.e., parent title, parent text, anchor text, extended anchor text, with their weights summing to one. As shown in Table 2, the top combinations with highest accuracy assign little value to anchor text and extended anchor text.

In addition, we examined the combination of the four fields on parent with the text on the target page (without emphasizing title of target). The result is similar: anchor text and extended anchor text is somewhat useful when combined with target text, but not as useful as the others.

Fürnkranz [12] and Glover et al. [15] showed that off-page anchor text is quite useful. Although we also find anchor text useful, it is not as significant. The reasons could be:

- different datasets: although pages in the Yahoo! directory used by Glover et al. are similar in many ways to those in ODP, Glover et al. used finer categories (second-level or deeper) while we only consider top-level categories;

Anchor	Extended anchor	Title	Text	Accuracy
0.0	0.0	0.5	0.5	0.7085
0.0	0.0	0.4	0.6	0.7048
0.0	0.2	0.2	0.6	0.7028
0.2	0.0	0.4	0.4	0.7003
0.0	0.2	0.4	0.4	0.6997

Table 2: Combinations of parent fields with highest accuracy.

Field	Target	Parent	Child	Sibling	Spouse
Title	0.2	0.5	0.2	0.3	0.4
Text	0.8	0.5	0.8	0.7	0.6
Anchor	N/A	0.0	N/A	N/A	N/A
Extended anchor	N/A	0.0	N/A	N/A	N/A
Accuracy	0.7541	0.7085	0.4951	0.6156	0.5635
Baseline	0.7482	0.5942	0.4321	0.5957	0.4669
Relative improvement	0.79%	19.24%	14.58%	3.34%	20.69%

Table 3: Summary of lower layer optimization for each neighbor type.

- different numbers of incoming links: Glover et al. used at most 20 incoming links for each target page, while we used at most 50, which may consequently affect the describing power of the collected anchor text;
- different ways of using anchor text: Glover et al. directly pulled all anchor text into a virtual document without normalization, and only consulted the classifier based on the target text when the classifier based on anchor text is uncertain, while we normalize each anchor text before combining, and combine it with local text without considering uncertainty.

Throughout our experiments, we found that the title of a parent page is more important than anchor text. Possible explanations include the following.

- We focused on broad categories. Anchor text are usually specific (e.g., "Nokia"), therefore may not help much in classifying broad categories. On the other hand, titles are usually more generic than any anchor text on the same page (e.g., "cell phones"), which makes them more useful.
- The pages that we used in our ODP experiment are considered to be good quality pages. The content of these pages tends to be well-written and self-describing. Therefore, resorting to supplemental information (such as anchor text) is not as likely to achieve significant improvement.

In summary, the best combination achieved in lower layer optimization is listed in Table 3. To show the benefit from emphasizing titles, we compared them with a baseline which classifies based on the full content of that neighbor type.

4.5 Upper layer optimization

Based on the optimal combinations achieved in the lower layer optimization of each neighbor type, upper layer optimization tunes the weighting between neighbor types with the weighting within each neighbor type fixed. For example, if the best weight of sibling title and sibling text found in lower layer optimization are $weight_{sibling:title}$, and $weight_{sibling:text}$ (as listed in Table 3, with values 0.3 and 0.7), respectively, then their final weight in the full combination is $weight_{sibling} \times weight_{sibling:title}$, and $weight_{sibling} \times weight_{sibling:text}$, respectively, where $weight_{sibling}$ (as well as weighting of other neighbor types) is to be determined by experiments.

The top combinations with highest accuracy are listed in Table 4, which shows little value for child and spouse pages, low but consistent value of sibling pages, and high value of parent pages. Compared with the usefulness showed by the default Neighboring algorithm in Section 2.3, the usefulness of child and spouse is similar. Parent pages gain more importance because of emphasizing title; while siblings become less useful.

target	Parent	Child	Sibling	Spouse	Accuracy
0.2	0.4	0.0	0.2	0.2	0.8729
0.4	0.4	0.0	0.2	0.0	0.8709
0.2	0.6	0.0	0.2	0.0	0.8688
0.2	0.4	0.2	0.2	0.0	0.8676
0.4	0.4	0.0	0.0	0.2	0.8645

Table 4: Combination of higher layer optimization with highest accuracy.

target	Parent	Child	Sibling	Spouse	Accuracy
0.4	0.2	0.0	0.2	0.2	0.8467
0.2	0.4	0.0	0.2	0.2	0.8443
0.2	0.4	0.0	0.2	0.2	0.8440

Table 5: Combination of non-fielded (full) content of neighbors with highest accuracy.

In order to compare fielded classification with a non-fielded version, we also performed another tuning in which only *full text* of the target and its neighbors are used. The top combinations are listed in Table 5. The best combination of full text is not as accurate as the fielded combination.

After lower and upper layer optimization, the best parameter combination of all fields is listed in Table 6.

4.6 Experimental result on ODP dataset

After tuning the weighting of each field, we apply the best parameter setting (as listed in Table 6) to the set-aside test documents. We compare the result with SVM classifiers (based on *SVM*^{light}) using only target text, with the default Neighboring algorithm using the best parameter setting reported in [23], and with our implementation of *IO-bridge*, one of the algorithms suggested by Chakrabarti et al. [6].

The main idea of *IO-bridge* is to build an engineered document corresponding to each document in the dataset, in which the constructed document consists only of 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. In the following, we compare our algorithm with both *IO-bridge* and the baseline, textual classifiers. We trained *IO-bridge* on the same 8,400 documents as was used to tune our F-Neighbor algorithm, and tested it on the same test set.

Note that the feature selection based on mutual information was not present in the original implementation of the default Neighboring algorithm as described in [23]. However, the results reported here is based on the selected feature subset for a fair comparison.

Field	Weight
Target title	0.04
Target text	0.16
Parent title	0.2
Parent text	0.2
Anchor text	0
Extended anchor text	0
Child title	0
Child text	0
Sibling title	0.06
Sibling text	0.14
Spouse title	0.08
Spouse text	0.12

Table 6: The best combination of parameters.

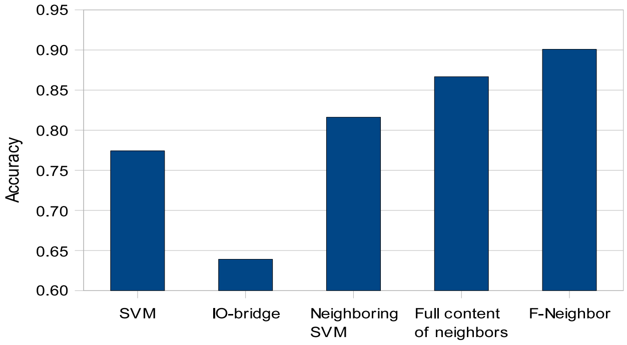


Figure 6: Comparison of accuracy of different algorithms on ODP data.

The result is shown in Figure 6. Textual SVM classifiers classified 77% of the pages correctly. The default Neighboring algorithm (based on SVM) outperformed textual SVM by 5%. Previously in the optimization process, we tested combinations using only the full content of the target page and its neighbors. We also applied the best combination obtained from that experiment to the test set (labeled as “full content of neighbors”), and got an accuracy of 86.6%. F-Neighbor algorithm further raised that to 90%, reducing the error rate of the default Neighboring algorithm by almost half. Two-tailed student t-tests show that the improvements of F-Neighbor algorithm over other algorithms are statistically significant ($p << .01$).

4.7 Experimental result on WebBase dataset

In order to demonstrate our algorithm’s performance on more generic web pages, we also apply the trained classifier to the WebBase dataset as mentioned in Section 4.1. Since there are no human-assigned labels for WebBase pages, we manually labeled 122 pages randomly sampled from the WebBase dataset using one of the twelve categories listed in Table 1. On our labeled subset of pages, the default Neighboring algorithm increased the accuracy of textual SVM classifiers from 31.7% to 37.7%. The F-Neighbor algorithm further improved that to 44.7%.

5. DISCUSSION

This paper has demonstrated the value of utilizing text fields from neighboring pages in web page classification. Here, we discuss some limitations of our current approach.

- The current approach does not utilize human labels of neighboring pages, which has been shown to be a strong signal for the topic of a page [6, 4, 23] (although we did compare to two methods that did use labels). Therefore, further study may show that F-Neighbor algorithm can also benefit from using human labels of neighbors.
- By assuming that the weighting of the fields of a neighbor type is independent of other neighbor types, our two-layer optimization method greatly reduced the number of combinations that needed to be tested at the risk of possibly finding a sub-optimal solution. We did not expect to find the best solution through this method. However, a second round of optimization was only able to increase accuracy by 0.2%.
- While we break up a web page into several fields according to its HTML markup, other fields may also be worth considering, e.g., headers or text in large fonts. To be more general, one may break up pages based on metrics other than HTML tags, such as spatial information or even complex models as the one proposed by Xiao et al. [32].
- 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 neighbor 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 [6, 19, 1]) might be a useful addition to our approach.
- For simplicity, the classification showed in this paper is only on first-level categories of dmoz Directory. Conducting similar classification at a deeper level, or on more fine-grained topics, may expose more interesting facts.
- Our implementation of *IO-bridge* may not reflect the full power of *IO-bridge* in that we utilized a naive Bayes classifier with it (rather than their TAPER classifier) and we report the average classification accuracy over all test pages, including ones that do not have a labeled sibling where *IO-bridge* will always fail.

6. SUMMARY

This paper has exploited field information from neighboring pages to help judge the topic of a target web page. In contrast to previous approaches, our approach utilizes text from different fields from neighbors with consideration of their different importance. As a result, our approach is able to generate more accurate representations of documents, facilitating more accurate classification. In summary, our contributions include the following:

- We tested multiple classification algorithms on a large, real-world web page dataset.
- We showed greatly improved accuracy on web page classification, reducing error rates by more than half over a commonly used text classification approach.
- We demonstrated the usefulness of field information in HTML document classification, and found that emphasizing page titles, especially parent titles, over other text can bring improvement.

In the future, we plan to incorporate human-generated labels into this classification approach for further improvement.

Acknowledgments

We thank Na Dai for helpful discussions. This material is based upon work supported by the National Science Foundation under Grant Numbers IIS-0328825 and IIS-0545875.

7. REFERENCES

[1] R. Angelova and G. Weikum. Graph-based text classification: Learn from your neighbors. In *Proc. 29th Annual Int’l ACM SIGIR Conf. on Research and Development in Info. Retrieval*, pages 485–492, 2006.

[2] G. Attardi, A. Gulli, and F. Sebastiani. Automatic web page categorization by link and context analysis. In *Proc. of the European Symposium on Telematics, Hypermedia and Artificial Intelligence (THAI)*, pages 105–119, 1999.

[3] K. Balog and M. de Rijke. Finding experts and their details in e-mail corpora. In *Proc. of the 15th Int'l Conf. on the World Wide Web*, pages 1035–1036. ACM, 2006.

[4] 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 *Proc. of the 12th Int'l Conf. on Info. and Knowledge Mgmt.* ACM, 2003.

[5] C. Castillo, D. Donato, A. Gionis, V. Murdock, and F. Silvestri. Know your neighbors: web spam detection using the web topology. In *Proc. of the 30th Annual Int'l ACM SIGIR Conf. on Research and Development in Information Retrieval*, pages 423–430, 2007.

[6] S. Chakrabarti, B. E. Dom, and P. Indyk. Enhanced hypertext categorization using hyperlinks. In *Proceedings of ACM SIGMOD Int'l Conf. on Management of Data*, pages 307–318, 1998.

[7] S. Chakrabarti, M. M. Joshi, K. Punera, and D. M. Pennock. The structure of broad topics on the web. In *Proceedings of the 11th International World Wide Web Conference*, pages 251–262. ACM Press, May 2002.

[8] S. Chakrabarti, M. van den Berg, and B. E. Dom. Focused crawling: A new approach to topic-specific Web resource discovery. In *Proc. of the 8th Int'l World Wide Web Conference*, pages 545–562, May 1999.

[9] J. Cho, H. Garcia-Molina, T. Haveliwala, W. Lam, A. Paepcke, S. Raghavan, and G. Wesley. Stanford WebBase components and applications. *ACM Trans. on Internet Technology*, 6(2):153–186, 2006.

[10] W. W. Cohen. Improving a page classifier with anchor extraction and link analysis. In S. Becker, S. Thrun, and K. Obermayer, editors, *Advances in Neural Information Processing Systems*, volume 15, pages 1481–1488. MIT Press, Cambridge, MA, 2002.

[11] B. D. Davison. Topical locality in the Web. In *Proc. of the 23rd Annual ACM SIGIR Int'l Conf. on Research and Dev. in Info. Retrieval*, pages 272–279, July 2000.

[12] J. Fürnkranz. Exploiting structural information for text classification on the WWW. In *Proc. of the 3rd Symp. on Intelligent Data Analysis (IDA)*, volume 1642 of *LNCS*, pages 487–497. Springer-Verlag, 1999.

[13] J. Fürnkranz. Hyperlink ensembles: A case study in hypertext classification. *Journal of Information Fusion*, 1:299–312, 2001.

[14] R. Ghani, S. Slattery, and Y. Yang. Hypertext categorization using hyperlink patterns and meta data. In *Proc. of the 18th Int'l Conf. on Machine Learning (ICML)*, pages 178–185. Morgan Kaufmann, 2001.

[15] E. J. Glover, K. Tsioutsiouliklis, S. Lawrence, D. M. Pennock, and G. W. Flake. Using Web structure for classifying and describing Web pages. In *Proc. of the 11th Int'l Conf. on the World Wide Web*, 2002.

[16] K. Golub and A. Ardo. Importance of HTML structural elements and metadata in automated subject classification. In *Proc. of the 9th European Conf. on Research and Advanced Technology for Digital Lib. (ECDL)*, pages 368–378, 2005.

[17] T. H. Haveliwala. Topic-sensitive PageRank. In *Proc. of the 11th Int'l World Wide Web Conf.*, pages 517–526. ACM Press, May 2002.

[18] T. Joachims. Making large-scale support vector machine learning practical. In *Advances in Kernel Methods: Support Vector Machines*. MIT Press, Cambridge, MA, 1998.

[19] Q. Lu and L. Getoor. Link-based classification. In *Proc. of the 20th Int'l Conf. on Machine Learning (ICML)*, Menlo Park, CA, Aug. 2003. AAAI Press.

[20] L. Nie, B. D. Davison, and X. Qi. Topical link analysis for web search. In *Proc. of the 29th Annual Int'l ACM SIGIR Conf. on Research & Development in Information Retrieval*, pages 91–98, Aug. 2006.

[21] H.-J. Oh, S. H. Myaeng, and M.-H. Lee. A practical hypertext catergorization method using links and incrementally available class information. In *Proc. of the 23rd Annual Int'l ACM SIGIR Conf. on Research and Development in Information Retrieval*, pages 264–271, 2000.

[22] D. Petkova and W. B. Croft. Hierarchical language models for expert finding in enterprise corpora. In *Proc. of the 18th IEEE Int'l Conf. on Tools with Artificial Intelligence (ICTAI)*, pages 599–608. IEEE Computer Society, 2006.

[23] X. Qi and B. D. Davison. Knowing a web page by the company it keeps. In *Proc. of the 15th ACM Int'l Conf. on Information and Knowledge Management (CIKM)*, pages 228–237, Nov. 2006.

[24] 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.

[25] S. Robertson, H. Zaragoza, and M. Taylor. Simple BM25 extension to multiple weighted fields. In *Proc. of the 13th ACM Int'l Conf. on Information and Knowledge Management (CIKM)*, pages 42–49, 2004.

[26] S. E. Robertson and S. Walker. Some simple effective approximations to the 2-poisson model for probabilistic weighted retrieval. In *Proc. of the 17th Annual Int'l ACM SIGIR Conf. on Research and Development in Info. Retrieval*, pages 232–241, 1994.

[27] G. Salton, editor. *The SMART Retrieval System: Experiments in Automatic Document Retrieval*. Prentice Hall, Englewood Cliffs, NJ, 1971.

[28] S. Slattery and T. Mitchell. Discovering test set regularities in relational domains. In *Proc. of the 17th Int'l Conf. on Machine Learning (ICML)*, 2000.

[29] K. Sugiyama, K. Hatano, M. Yoshikawa, and S. Uemura. Refinement of TF-IDF schemes for web pages using their hyperlinked neighboring pages. In *Proc. of the 14th ACM Conf. on Hypertext and Hypermedia*, pages 198–207, 2003.

[30] A. Sun, E.-P. Lim, and W.-K. Ng. Web classification using support vector machine. In *Proc. of the 4th Int'l Workshop on Web Information and Data Management (WIDM)*, pages 96–99. ACM Press, 2002.

[31] H. Utard and J. Fürnkranz. Link-local features for hypertext classification. In *Semantics, Web and Mining: Joint International Workshops, EWMF/KDO*, volume 4289 of *LNCS*, pages 51–64, Berlin, Oct. 2005. Springer.

[32] X. Xiao, Q. Luo, X. Xie, and W.-Y. Ma. A comparative study on classifying the functions of web page blocks. In *Proc. of the 15th ACM Int'l Conf. on Info. and Knowledge Mgmt*, pages 776–777, 2006.

[33] Y. Yang, S. Slattery, and R. Ghani. A study of approaches to hypertext categorization. *J. of Intelligent Info. Systems*, 18(2-3):219–241, 2002.

[34] S. Zhu, K. Yu, Y. Chi, and Y. Gong. Combining content and link for classification using matrix factorization. In *Proc. of the 30th Annual Int'l ACM SIGIR Conf. on Research and Development in Information Retrieval*, pages 487–494, 2007.