, , 1–25 ()

A Study of Approaches to Hypertext Categorization

YIMING YANG*, SEÁN SLATTERY* AND RAYID GHANI*†

yiming.yang@cs.cmu.edu, sean.slattery@cs.cmu.edu, rayid.ghani@cs.cmu.edu

*School of Computer Science, Carnegie Mellon University, Pittsburgh, PA 15213

†Accenture Technology Labs - Research, Northbrook, IL 60062

Editor:

Abstract.

Hypertext poses new research challenges for text classification. Hyperlinks, HTML tags, category labels distributed over linked documents, and meta data extracted from related Web sites all provide rich information for classifying hypertext documents. How to appropriately represent that information and automatically learn statistical patterns for solving hypertext classification problems is an open question. This paper seeks a principled approach to providing the answers. Specifically, we define five *hypertext regularities* which may (or may not) hold in a particular application domain, and whose presence (or absence) may significantly influence the optimal design of a classifier. Using three hypertext datasets and three well-known learning algorithms (Naive Bayes, Nearest Neighbor, and First Order Inductive Learner), we examine these regularities in different domains, and compare alternative ways to exploit them. Our results show that the identification of hypertext regularities in the data and the selection of appropriate representations for hypertext in particular domains are crucial, but seldom obvious, in real-world problems. We find that adding the words in the linked neighborhood to the page having those links (both inlinks and outlinks) were helpful for all our classifiers on one data set, but more harmful than helpful for two out of the three classifiers on the remaining datasets. We also observed that extracting meta data from related Web sites was extremely useful for improving classification accuracy in some of those domains. Finally, the relative performance of the classifiers being tested provided insights into their strengths and limitations for solving classification problems involving diverse and often noisy Web pages.

Keywords: hypertext classification, machine learning, web mining, text mining

1. Introduction

As the size of the Web expands rapidly, the need for good automated hypertext classification techniques is becoming more apparent. The Web contains over two billion pages connected by hyperlinks, making the task of locating specific information on the Web increasingly difficult. A recent user study [2] showed that users often prefer navigating through directories of pre-classified content, and that providing a category-based view of retrieved documents enables them to find more relevant information in a shorter time. The common use of category hierarchies for navigation support in Yahoo! and other major Web portals has also demonstrated the practical utility of hypertext categorization.

Automated hypertext classification poses new research challenges because of the rich information in a hypertext document and the connectivity among documents. Hyperlinks, HTML tags, category distributions over a linked neighborhood, and meta data extracted from related Web sites all provide rich information for hypertext classification, which is not normally available in traditional text classification. Researchers have only recently begun to explore the issues of exploiting rich hypertext information for automated classification.

Chakrabarti et al. [1] studied the use of citations in the classification of IBM patents where the citations between documents (patents) were considered as "hyperlinks", and the categories were defined in a topical hierarchy. Similar experiments on a small set of Web pages (only 900 pages from Yahoo!) with real hyperlinks were also conducted. By using the system-predicted category labels for the linked neighbors of a test document to reinforce the category decision(s) on that document, they obtained a 31% error reduction, compared to the baseline performance when using the local text in the document alone. They also tested a more naive way of using the linked documents, treating the words in the linked documents as if they were local. This approach *increased* the error rate of their system by 6% over the baseline performance.

Oh et al. [18] reported similar observations on a collection of online Korean encyclopedia articles. By using the system-predicted categories of the linked neighbors of a test document to reinforce the classification decision(s) on that document, they obtained a 13% improvement in F_1 (defined in Section 4.2) over the baseline performance (when using local text only). On the other hand, when treating words in the linked neighborhood of a document as if they were local words in that document, the performance of their classifier (Naive Bayes) decreased by 24% in micro-averaged F_1. Instead of using all the links from a document, they decided to use only a subset of the linked documents based on the cosine similarity between the "bags of words" of pairwise linked documents – the links with low similarity scores were ignored. This filtering process yielded a 7% improvement in F_1 over naively using all the links.

Fürnkranz [10] used a set of Web pages from the WebKB University corpus (Section 3) to study the use of anchor text (words on a link) and the words "near" the anchor text in a Web page to predict the class of the target page pointed to by the links. By representing the target page using the anchor words on all the links that point to it, plus the headlines that structurally precede the sections where links occur, the classification accuracy of a rule-learning system (Ripper [5]) improved by 20%, compared to the baseline performance of the same system when using the local words in the target page instead.

Slattery and Mitchell [23] also used the WebKB University corpus, but studied alternative learning paradigms, namely, a First Order Inductive Learner (Foil) [19] which exploits the relational structure among Web pages, and a Hubs & Authorities style algorithm [15] exploiting the hyperlink topology. They found that a combined use of these two algorithms performed better than using each alone.

Joachims et al. [14] also reported a study using the WebKB University corpus, focusing on Support Vector Machines (SVMs) with different kernel functions. Using one kernel to represent a document based on its local words, and another kernel to represent hyperlinks, they give evidence that combining the two kernels leads to better performance in two out of three classification problems. Their experiments suggest that the kernels can make more use of the training-set category labels in the linked neighborhood of a document compared to the local words in that document.

Whereas the work summarized above provides initial insights in exploiting information in hypertext documents for automated classification, many questions still remain unanswered. For example, it is not entirely clear why the use of anchor words improve classification accuracy in Fürnkranz's experiments on the WebKB pages, but the inclusion of all linked words decreased performance in Chakrabarti's experiments on the IBM patents. Recall that anchor words are a subset of the linked words (from the in-links). What would happen if Fürnkranz expanded the subset to the full set of linked words in the WebKB pages, or if Chakrabarti et al. selected a subset (the words from the anchor fields of the in-link pages only, for example) instead of the full set of words in the IBM patents? How much did the difference between the data contribute to the reported performance variance? How much did the particular algorithms used in those experiments influence the observations? Since most of the experimental results are not directly comparable (even true for the results on the WebKB corpus because of different subsets of documents and categories used in those experiments), the answers to these questions are not clear.

In order to draw general conclusions about hypertext classification, we need more systematic experiments and better analysis about the potential reasons behind the observed performance variances. As a step in that direction, we begin with hypotheses about hypertext regularities (Section 2.1), then report systematic examinations of these hypotheses on the cross product of three data collections (Section 3), three classification algorithms (Section 2.3), and various representations for hypertext data (Section 2.2). We also provide direct data analysis in support of our empirical results with our classifiers for the hypertext regularities being tested (Section 4), leading toward generalizable observations and conclusions (Section 5).

2. Methodology

The purpose of the experiments presented in this paper is to explore various hypotheses about the structure of hypertext especially as it relates to hypertext classification. While the scope of the experimental results presented is necessarily confined to the three classification problems described in Section 3, we hope that the analysis that follows will help future research into hypertext classification by providing some ideas about various types of regularities that may be present in other

Table 1. Definitions of five possible regularities we can use when classifying documents of class A.

Regularity	Definition
None	Documents neighboring class A documents exhibit no pattern.
Encyclopedia	Documents neighboring class A documents are all of class A.
Co-referencing	Documents neighboring class A documents all share the same class, but are not of class A.
Preclassified	A single document points only to all documents of class A.
Meta data	Relevant text extracted from sources external to the Web document, or internal but not visible on that document.

hypertext corpora and how one should construct a classifier to take advantage of them.

2.1. Regularities

Before we can exploit patterns in a hypertext corpus, we need to understand what kind of regularities to expect. This section presents a list of what we believe are the simplest kinds of hypertext regularities we might consider searching for.

Of course we still expect the content the document being classified to be a primary source of information and this list is meant to explore where we might look for more information in a hypertext classification problem. Succinct definitions of these regularities are given in Table 1.

2.1.1. No Hypertext Regularity It is important to be aware that for some hypertext corpora and classification tasks on them, the only useful place to look for information about the class label of a document is the document itself. In cases like this, looking outside the document for information about the label is not going to help and may in some cases hurt classification performance. However, we believe that for many real-world hypertext classification tasks, extra information is available to improve upon the performance of a classifier which only uses the content of each document.

2.1.2. Encyclopedia Regularity Perhaps the simplest regularity we might hope to find is one where documents with a given class label only link to documents with the same class label. We might expect to find approximately this regularity in a corpus of encyclopedia articles, such as the ETRI-Kyemong encyclopedia corpus used in [18], since encyclopedia articles generally reference other articles which are topically similar.

2.1.3. Co-Referencing Regularity Instead of having the same class, neighboring documents can have some other topic in common with each other. For example, news articles about a particular current event may link to many articles about the background for that event. As another example, previous work [22] found that when learning to classify course home pages, a group of the neighboring pages about homework assignments were found to be useful, even though those homework assignment pages were not part of the learning task and not labelled in the data. It is important to realize that, in general, the topic of these linked documents may not correspond to any class in the classification problem.

A variant of this regularity relaxes the requirement that all of the neighboring documents share the same topic. Instead, it may be the case that only some neighboring documents of a class share the same topic. For example, all faculty home pages may contain a link to a page describing research interests. If we can find this regularity, it can help us with classification. In previous work [12] we described this type of regularity as a partial co-referencing regularity. This type of regularity is particularly difficult to exploit because it requires searching for subsets of the neighboring documents that have some unseen topic in common.

2.1.4. Preclassified Regularity[1] While the encyclopedia and co-referencing regularities consider the topic of neighboring documents, there can also be regularities in the hyperlink structure itself. One such regularity prevalent on the Web consists of a single document which contains hyperlinks to documents which share the same topic. Finding this "hub" document would help us in classifying all the documents that are linked from it. Categories on the Yahoo! topic hierarchy are a perfect example of this regularity. If a category on the Yahoo! hierarchy happens to corresponds to a class in our classification problem, then we say that the pages linked to that category exhibit a preclassified regularity, since in effect the creator of those hyperlinks has preclassified all the documents of some class for us.

2.1.5. Meta Data Regularity For many classification tasks that are of practical and commercial importance, meta data are often available from external sources on the Web that can be exploited in the form of additional features. Examples of these types of meta data include movie reviews for movie classification, online discussion boards for various other topic classification tasks (such as stock market predictions or competitive analysis). Meta data is also implicitly present in many Web documents in the form of text within `META` tags and within `ALT` and `TITLE` tags (which are not visible when viewing the Web page through most browsers). If we can extract rich and predictive features from such sources, we can build classifiers that can use them alone or combine them with the hyperlinks and textual information.

2.2. Hypertext Classification Approaches

Depending on which of the above regularities holds for the hypertext classification task under consideration, different classifier designs should be considered. Likewise for given applications, using different classifier designs, we can search for various hypertext regularities in the dataset.

In our experiments for this paper, for simplicity we did not distinguish the links to and from a page. Our examination consists of the following components:

2.2.1. No Hyperlink Regularity With no regularity, we expect no benefit from using hyperlinks and would use standard text classifiers on the text of the document itself. Using such classifiers as performance baselines and comparing them to classifiers which take hyperlinks into account will allow us to test for the existence of various hyperlink regularities.

It is quite possible that for a substantial fraction of hypertext classification tasks there is no hyperlink regularity that can be exploited and the best performance we can hope for is with a standard text classifier. Indeed in such cases, learners looking for hyperlink regularities may be led astray and end up performing worse than a simple text classifier.

2.2.2. Encyclopedia Regularity If the encyclopedia regularity holds in a given data set, then augmenting the text of each document with the text of its neighbors should produce better classification results because more topic-related words would be present in the document representation. Chakrabarti et al. applied this approach to a database of patents and found that classification performance suffered, suggesting that the patent database is unlikely to have this structure [1].

2.2.3. Co-Referencing Regularity If the co-referencing regularity holds, then augmenting each document as above except treating the additional words as if they come from a separate vocabulary should help classification. A simple way to do this is to prefix the words in the linked documents with a tag. Chakrabarti et al. also tried this approach on the patent database and again found that performance suffered, suggesting that the patent database is unlikely to have this structure [1].

If instead we have a partial co-referencing regularity, we need to identify the linked pages which are topically similar to each other (the "research interests" pages from the faculty home page example in Section 2.1.3). This can be achieved by computing the text-based similarity among all the documents linked to documents in the same class and clustering them accordingly. In contrast to the previous approaches which can be characterized as using various "bag-of-words" representations and standard text classification algorithms, this approach requires a more elaborate algorithm. One such algorithm is the FOIL algorithm described in the next section. Craven et al. [7] applied FOIL to the WebKB University corpus and found that it did improve classification performance, indicating that this corpus does have this kind of regularity. The cosine-similarity based filtering of linked neighbors by Oh et al. (Section 1) is another example of utilizing this regularity.

2.2.4. Preclassified Regularity If the classification scheme of our corpus is already embedded in the hyperlink structure, we have no need to look at the text of any document. We just need to find those pages within the hypertext "graph" that have this property. We can search for those pages by representing each page with only the names of the pages it links with. If any of these linked pages are correlated with a class label, a reasonable learning algorithm should be able to recognize it as a predictive feature and use it (often together with other features) to make classification decisions. The successful use of the SVM kernel function for hyperlinks by Joachims et al. (Section 1) illustrated one way of exploiting this regularity. In this paper we show alternative approaches and use of this regularity with the kNN, NB and FOIL algorithms.

2.2.5. Meta Data Regularity When external sources of information are available that can be used as meta data, we can collect them, possibly using information extraction techniques. In particular, we look for features that relate two or more entities/documents being classified. Following the approaches outlined above for hyperlinks, these extracted features can then be used in a similar fashion by using the identity of the related documents and by using the text of related documents in various ways. Any information source from the Web about the entity being classified can be used as a meta data resource and the availability and quality of such resources will certainly depend on the classification task. Cohen [4] described some experiments where he automatically located and extracted such features for several (non-hypertext) classification tasks.

We also look for "meta data" contained within Web pages such as `META` and `TITLE` tags. The information contained within these HTML tags in a page is technically not meta data because it is *internal* rather than external to the page. Nevertheless, these tagged fields can be treated differently from other parts of Web pages and can a useful source for classification.

2.3. Learning Algorithms Used

Our experiments used three existing classifiers: Naive Bayes, kNN and FOIL. Naive Bayes and kNN have been thoroughly evaluated for text classification on benchmark collections and offer a strong baseline for comparison. FOIL is a relational learner which has shown promise for hypertext classification.

The following notation is used for the descriptions of Naive Bayes and kNN:

$\mathcal{D}$ – Set of training documents

$\mathcal{D}_j$ – Set of training documents in class c_j

$n(t)$ – Number of training documents containing t

$N(t)$ – Number of occurrences of t

$N(t, d)$ – Number of occurrences of t in document d

2.3.1. Naive Bayes Naive Bayes is a simple but effective text classification algorithm [16, 17]. The parameterization given by Naive Bayes defines an underlying generative model assumed by the classifier. In this model, first a class is selected according to class prior probabilities. Then, the generator creates each word in a document by drawing from a multinomial distribution over words specific to the class. Thus, this model assumes each word in a document is generated independently of the others given the class.

Naive Bayes forms maximum a posteriori estimates for the class-conditional probabilities for each term in the vocabulary, V, from labeled training data $\mathcal{D}$. This is done by calculating the frequency of each term t over all the documents in a class c_j, supplemented with Laplace smoothing to avoid zero probabilities:

$$\Pr(t|c_j) = \frac{1 + \sum_{d \in \mathcal{D}_j} N(t,d)}{|V| + \sum_{t \in V} \sum_{d \in \mathcal{D}_j} N(t,d)}. \tag{1}$$

We calculate the prior probability of each class ($\Pr(c_j)$) from the frequency of each document label in the training set.

At classification time we use these estimated parameters by applying Bayes' rule (using a word independence assumption) to calculate the posterior probability of each class label ($\Pr(c_j|d)$) for a test document d, and taking the most probable class as the prediction (since all the documents in our datasets used for this study belong to one and only one class; see Section3 for details):

$$\arg\max_{c_j} \Pr(c_j) \prod_{t \in d} \Pr(t|c_j)^{N(t,d)}. \tag{2}$$

2.3.2. K-Nearest Neighbor (kNN) kNN, an instance-based classification method, has been an effective approach to a broad range of pattern recognition and text classification problems [8, 25, 26, 28]. In contrast to "eager learning" algorithms (including Naive Bayes) which have an explicit training phase before seeing any test document, kNN uses the training documents "local" to each test document to make its classification decision on that document. Our kNN uses the conventional vector space model, which represents each document as a vector of term weights, and the similarity between two documents is measured using the cosine value of the angle between the corresponding vectors. We compute the weight vectors for each document using one of the conventional TF-IDF schemes [20], in which the weight of term t in document d is defined as:

$$w_d(t) = (1 + \log_2 N(t,d)) \times \log_2(|\mathcal{D}|/n(t)) \tag{3}$$

Given an arbitrary test document d, the kNN classifier assigns a *relevance* score to each candidate category (c_j) using the following formula:

$$s(c_j, d) = \sum_{d' \in R_k(d) \cap \mathcal{D}_j} \cos(d, d') \tag{4}$$

where set $R_k(d)$ are the k nearest neighbors (training documents) of document d. By sorting the scores of all candidate categories, we obtain a ranked list of categories for each test document; by further thresholding on the ranks or the scores, we obtain binary decisions, i.e. the categories above the threshold will be assigned to the document. There are advantages and disadvantages of different thresholding strategies [26, 27]. In this paper, we use the simplest strategy – assigning the top-ranking category only to each document as a baseline; for more flexible trade-off between recall and precision, we further threshold on the scores of the top-ranking candidates, as described in Section 4.4.

2.3.3. FOIL Quinlan's FOIL [19] is a greedy covering algorithm for learning function-free Horn clauses. It induces each clause by beginning with an empty tail and using a hill-climbing search to add literals until the clause covers as few negative instances as possible. The evaluation function used to guide the hill-climbing search is an information-theoretic measure.

FOIL has already been used for text classification to exploit word order [3] and hyperlink information [7]. Here FOIL is used as described in [7], using a unary `has_`*word*`(page)` relation (where *word* is a variable) for each word and a `link_to(page,page)` relation for hyperlinks between pages. The former allows FOIL to distinguish informative *words* from non-informative ones, and the latter gives FOIL the power to recognize predictive *links* among pages among all the links.

It is a common practice to apply class-driven feature selection to documents before training a classifier, for reducing the computational cost and possibly improving the effectiveness. However, for relational hypertext classification, this is less than straightforward (how would we know that words relating to assignments in a linked page would help to classify course home pages without searching for that regularity first?). The experiments presented here use document frequency feature selection.

All the FOIL experiments in this paper were done by running the algorithm on binary subproblems, one for each class in the problem. For each test example, the system computed a score for each class by picking the matching rule with highest confidence score from each of the learned binary classifiers. The confidence scores were based on the training-set accuracies of the rules. This process results in a list of scored categories for each test example, allowing further thresholding for classification decisions, as described in the kNN section above. This is perhaps the simplest approach to combining the outputs of several FOIL classifiers, and more elaborate strategies would almost certainly do better.

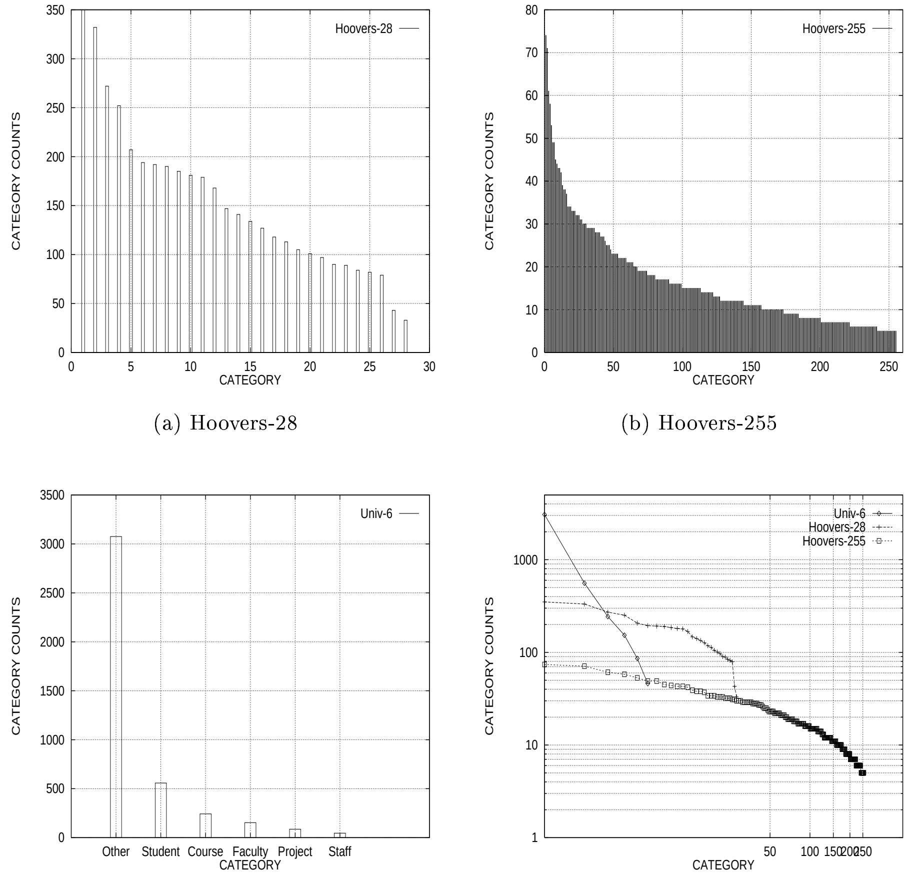
3. Datasets

To test our proposed approaches to hypertext classification, we needed datasets that would reflect the properties of real-world hypertext classification tasks. We wanted a variety of problems so we could get a general sense of the usefulness of each regularity described in the previous section.

We found three hypertext classification problems for this study: two of them are about classification of company Web sites, and the third one is a classification task for university Web pages.

3.1. Hoovers-28 and Hoovers-255

The Hoovers corpora of company Web pages was assembled using the Hoovers Online Web resource (`www.hoovers.com`) which contains detailed information about a large number of companies and is a reliable source of corporate information. Ghani et al. [11] obtained a list of the names and home-page URLs for 4285 companies on the Web and used a custom crawler to extract information from company Web sites. This crawler visited 4285 different company Web sites and searched up to the first 50 Web pages on each site (in breadth first order), examining just over 108,000 Web pages.



(a) Hoovers-28

(b) Hoovers-255

(c) Univ-6

(d) All problems, logarithmic scale

Figure 1. Category distributions for all three problems

Two sets of categories are available from Hoovers Online: a coarse classification scheme of 28 classes ("Hoovers-28", defining industry sectors such as Oil & Gas, Sporting Goods, Computer Software & Services) and a more fine grained classification scheme consisting of 255 classes ("Hoovers-255").

These categories label companies, not particular Web pages. For this reason, we constructed one synthetic page per company by concatenating all the pages (up to 50) crawled for that company and ignoring the inner links between pages of that company. Therefore our task for this dataset is Web site classification rather than Web page classification due to the granularity of the categories in this application. Figures 1(a) and 1(b) show the category distributions for these two problems.

Previous work with this dataset [11] extracted meta data about these Web sites from Hoovers Online, which provided information about the company names, and names of their competitors. The authors constructed several kinds of wrappers (from simple string matchers to statistical information extraction techniques) to extract additional information about the relationships between companies from the Web pages in this dataset, such as whether one company name is mentioned by another in its Web page, whether two companies are located in the same state (in U.S.) or the same country (outside of U.S.), and so forth. In the results section, we only report our experiments using the competitor information because of space limitations.

The resulting corpora (namely, Hoovers-28 and Hoovers-255) consist of 4,285 synthetic pages with a vocabulary of 256,715 unique words (after removing stop words and stemming), 7,762 links between companies (1.8 links per company) and 6.0 competitors per company. Each Web site is classified into one category only for each classification scheme.

3.2. Univ-6 Dataset

The second corpus comes from the WebKB project at CMU [6]. This dataset was assembled for training an intelligent Web crawler which could populate a knowledge base with facts extracted directly from the Web sites of university computer science departments.

The dataset consists of 4,165 pages with a vocabulary of 45,979 unique words (after removing stop words and stemming). There are 10,353 links between pages in the corpus (2.5 links per page). Figure 1(c) shows the category distribution and Figure 1(d) shows how the category distributions for all three problems compare in logarithmic scales.

The pages were manually labelled into one of 7 classes: student, course, faculty, project, staff, department and other. The department class was ignored in our experiments as it had only 4 instances. The most populous class ("other") is a catch-all class which is assigned to documents (74% of the total) that do not belong in any of the defined classes of interest.

4. Empirical Validation

We conducted experiments aimed at testing the performance of each of the algorithms from Section 2.3 using Web page representations based on the discussion from Section 2.2. Note that each problem may or may not contain any of the regularities defined previously. Therefore, if a method does not perform well with a particular representation, it may be interpreted as either that the regularity does not exist in the task, or as evidence that the method is not well suited for making an effective use of the representation.

4.1. Experiments

To examine the six possible regularities discussed in Section 2.1, we tested NB, kNN and FOIL with the following representations of Web pages for all the three datasets (the hypertext regularity being considered is given in parentheses):

Page Only (No Regularity) Use only the words on the pages themselves (used with NB, kNN, FOIL)

Linked Words (Encyclopedia Regularity) Add words from linked pages (used with NB, kNN; not applicable to FOIL)

Tagged Words (Co-Referencing Regularity) Add words from linked pages but distinguish them with a prefix (used with NB, kNN; not needed for FOIL)

Tagged Words (Partial Co-Referencing Regularity) Represent Web pages individually and use a binary relation to indicate links (used with FOIL; not applicable to NB and kNN)

Linked Names (Preclassified Regularity) Represent each Web page by the names (or identifiers) of the Web pages it links to and ignore the words on the Web page entirely (used with NB, kNN and FOIL)

HTML Title (Meta Data Regularity) Use the HTML title of a Web page (used with NB, kNN and FOIL)

HTML Meta (Meta Data Regularity) Use the text found in `META` tags on a Web page (used with NB, kNN and FOIL).

In addition to the above representations, we also explored the use of the following representation for the Hoovers experiments:

Competitors (Meta Data Regularity) Use the competitor identifiers (aka "competitors") of a company to represent that company instead of the original Web page (used with NB, kNN, FOIL)

All of the results of the experiments are averages of five runs: each dataset was split into five subsets, and each subset was used once as test data in a particular run while the remaining subsets were used as training data for that run. The split into training and test sets for each run was the same for all the classifiers.

Table 2. Micro-averaged F_1 results for each classifier on each representation. Best results for each dataset with each representation are shown in bold.

	Hoovers28			Hoovers255			Univ6		
	NB	kNN	Foil	NB	kNN	Foil	NB	kNN	Foil
Page Only	55.1	**58.1**	31.5	**32.5**	32.0	11.6	69.6	**83.0**	82.7
Linked Words	**40.1**	38.6	N/A	18.9	**20.4**	N/A	74.1	**86.2**	N/A
Tagged Words	**49.2**	49.0	31.8	24.0	**26.9**	12.1	76.3	**88.0**	86.0
HTML Title	40.8	**43.3**	28.7	17.9	**22.6**	11.5	78.6	81.5	**86.3**
HTML Meta	48.6	**49.8**	29.3	23.1	**28.3**	13.1	73.3	78.6	**81.5**
Linked Names	**14.8**	13.3	12.3	5.0	**5.9**	4.6	81.7	**87.2**	86.6
Competitor Names	**75.4**	74.5	33.8	52.0	**53.0**	12.0	N/A	N/A	N/A

Table 3. Macro-averaged F_1 results for each classifier on each representation. Best results for each dataset with each representation are shown in bold.

	Hoovers28			Hoovers255			Univ6		
	NB	kNN	Foil	NB	kNN	Foil	NB	kNN	Foil
Page Only	54.3	**55.3**	31.6	**24.6**	19.8	8.0	31.4	46.4	**51.3**
Linked Words	**40.3**	35.1	N/A	**14.8**	12.0	N/A	38.3	**53.0**	N/A
Tagged Words	**49.0**	46.5	31.9	**17.9**	15.9	8.3	46.1	**59.1**	52.9
HTML Title	37.1	**39.9**	27.5	10.2	**14.5**	9.6	43.2	41.8	**50.1**
HTML Meta	45.1	**47.4**	29.8	13.8	**18.7**	10.6	14.1	**40.7**	39.3
Linked Names	11.8	**12.8**	9.5	2.4	**5.2**	3.6	47.0	44.3	**62.9**
Competitor Names	**75.2**	74.2	33.7	40.8	**44.5**	8.3	N/A	N/A	N/A

4.2. Overall Results

Tables 2, 3 and Figure 2 summarize the main results, where the performance of each classifier is measured using the conventional *micro-averaged* and *macro-averaged* recall, precision and F_1 values [24, 26]. Recall (r) is the ratio of the number of categories correctly assigned by the system to the test documents to the actual number of relevant document/category pairs in the test set; precision (p) is the ratio of the number of correctly assigned categories to the total number of assigned categories. The F_1 measure is defined to be $F_1 = 2rp/(r+p)$ which balances recall and precision in a way that gives them equal weight.

The recall, precision and F_1 scores can first be computed for individual categories, and then averaged over categories as a global measure of the average performance over all categories; this way of averaging is called *macro-averaging*. An alternative way, *micro-averaging*, is to count the decisions for all the categories in a joint pool and compute the global recall, precision and F_1 values for that global pool. Micro-averaged scores tend to be dominated by the performance of the system on common categories, while macro-averaged scores tend to be dominated by the performance on rare categories if the majority of categories in the task are rare. For skewed category distributions (Figure 1(c) in Section 3) in our tasks, providing both types of evaluation scores gives a clearer picture than considering either type alone.

Since our datasets used in this study are single-label-per-document tasks, micro-averaged accuracy, precision, recall and F_1 are all equal. We therefore report the F_1 score in our micro-averaged results, although all of the above measures can be used interchangeably. However, in general, the macro-averaged recall, precision and F_1 values are not the same. Further discussions on this issue can be found in the text categorization literature [26].

All the results reported are for optimal vocabulary sizes for each algorithm. The effect of feature selection on the categorization performance of our classifiers is analyzed in Section 4.3.

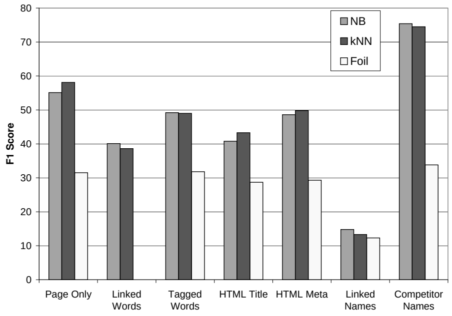
Some general observations can be made from the results of these experiments. The performance of a classifier depends on the characteristics of the problem, the information encoded in the document representation and the capability of the classifier in identifying regularities in documents. For the Univ-6 problem, all the three classifiers performed better when using hyperlink information (including *Linked Names*, *Tagged Words* and *Linked Words*) compared to using *Page Only*. For the two Hoovers problems, on the other hand, both kNN and NB suffered a significant performance decrease when using hyperlink information, except *Competitor Names*, while FOIL's performance was not significantly affected when given information about hyperlinked documents.

As for our specific hypotheses on hypertext regularities, we have the following observations:

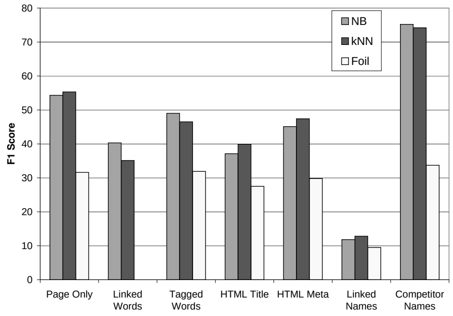
4.2.1. Page Only The *Page Only* results tell us something about the overall difficulty of each task. Unsurprisingly, problems with more classes proved to be more difficult. On the Hoovers problems, NB and kNN have roughly equal performance, while FOIL's performance is more competitive on Univ-6 than on the Hoovers datasets.

The big contrast between FOIL performance on Univ-6 and its performance on the Hoovers datasets is surprising, suggesting that FOIL may not be as robust or stable as NB and kNN for conventional text categorization. In particular FOIL is known to have a tendency to overfit the training data, since it was designed for learning logic programs. However, it is interesting that FOIL is the only classifier (among these three) with no performance degradation on all the three datasets when using *Tagged Words* instead of the *Page Only* setting, while NB and kNN suffered on Hoovers datasets due to the highly noisy hyperlinks (Section 4.2.2).

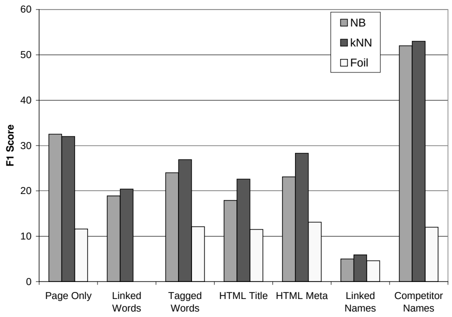
4.2.2. Linked Words The kNN and Naive Bayes results under the *Linked Words* condition in the Hoovers sets show that performance suffers badly when compared to the baseline. It is quite clear that these datasets do not exhibit this regularity. A close look at these datasets revealed that 56.8% of the Hoovers pages have links (1.8 links per page), but, according to the Hoovers-255 labelling, only 6.5% of the linked pairs of pages belong to the same category. This means that at most 3.7% (i.e., 56.8% $\times$ 6.5%) of the total pages would be possibly helped when using the system-assigned category labels to linked pages to reinforce the classification of those pages. In other words, a "perfect" hyperlink classifier (making perfect use of the category labels of linked pages) would show improved performance in classifying at most 3.7% of the total pages. Since our classifiers are not perfect,



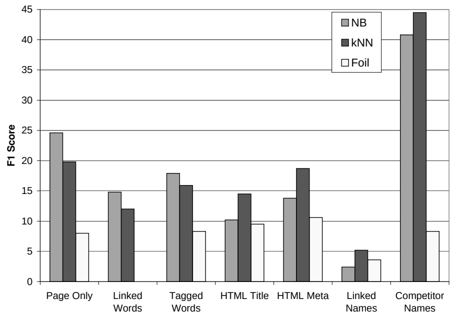
(a) Hoovers-28, micro-avg F_1



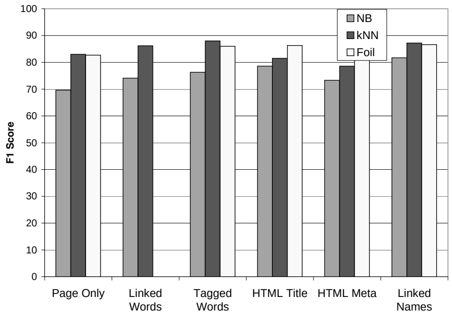
(b) Hoovers-28, macro-avg F_1



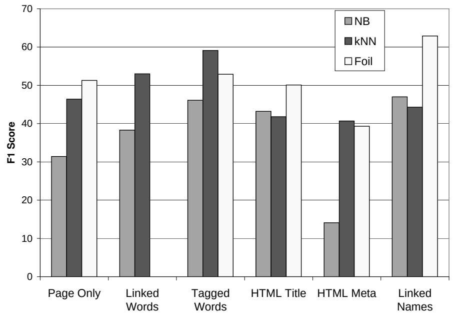
(c) Hoovers-255, micro-avg F_1



(d) Hoovers-255, macro-avg F_1



(e) Univ-6, micro-avg F_1



(f) Univ-6, macro-avg F_1

Figure 2. Performance of classifiers on Hoovers-28, Hoovers-255 and Univ-6

dumping the words from linked documents into the document having these links adds a tremendous amount of noise to the representation of that document. It is unsurprising, therefore, that the performance of kNN and NB with *Linked Words* suffered significantly on the Hoovers datasets. Even FOIL, designed for leveraging relational information in data, gains no improvement by using *Tagged Words* instead of *Page Only*.

In the Univ-6 dataset, on the other hand, 98.9% of pages have links (2.5 links per page on average), and 22.5% of the linked pairs of pages belong to the same category. This means that a perfect hyperlink classifier would improve performance on 22.3% of the total pages on Univ-6, which is much higher than the 3.7% of the Hoovers datasets. As a result, all the classifiers have improved results on Univ-6 when using *Linked Words* or *Tagged Words* instead of *Page Only*, indicating that hyperlinks in this dataset are informative for those classifiers.

A potential problem in the algorithm proposed by Oh et al. [18] for exploiting the *Encyclopedia* regularity is that it calculates the likelihood for each document belonging to a certain class by multiplying the system-estimated class probability (using the words in the Web page) by the fraction of neighbors that are in the same class. Applying their method to our Hoovers datasets would result in very poor performance since 96.3% of the Web pages do not have any linked neighbors in the same class and multiplying such a low probability to the likelihood scores based on page words will result in a zero or near zero probability for a category candidate for many documents.

4.2.3. Tagged Words The *Tagged Words* experiments examine the impact of the Co-Referencing regularity. Unlike previous results reported by Chakrabarti et al. [1] where tagging the words from the neighbors (treating them as if they're from a separate vocabulary) did not affect results, we find that this document representation results in significant performance degradation from the baseline for two (Hoovers-28 and Hoovers-255) of our problems but significant performance improvement from the baseline on one problem (Univ-6). An interesting observation is that using *Tagged Words* instead of *Linked Words* yielded less severe performance degradation for NB and kNN on the Hoovers datasets, and better performance for all the classifiers on Univ-6. One possible explanation for this is that tagging linked words allows a feature selection procedure to remove irrelevant linked words leaving the learner (classifier) with less noise in the data (evidence supporting this hypothesis is shown in Figure 3(e)), and allows the classifier to weight linked words differently from within-page words when making classification decisions.

Note that if our classifiers could handle noise "perfectly", then the performance would be at least as good as the baseline which ignores link information. This suggests that the feature selection methods and term weighting schemes we used with NB and kNN may have problems handling noise which could be overcome with more training data.

The FOIL results under *Tagged Words* look for partial co-referencing regularities. Here we see a slight improvement over the result of the baseline FOIL for Hoovers-255, and a small degradation for Hoovers-28 (in contrast to the severe performance degradations in NB and kNN), indicating that some small partial "co-referencing" regularity does exist and that FOIL was capable of exploiting it with its relational representation.

4.2.4. Linked Names The *Linked Names* experiments examine the impact of the Pre-classified Regularity. Our results show that this representation works surprisingly well for Univ-6 (which is consistent with the observations by Joachims et al. [14] for SVMs on another version of the same corpus), but was the worst choice (for all three classifiers) for the Hoovers datasets. Figure 2 shows the clear contrast that this regularity holds strongly in Univ-6 but not nearly so strongly in Hoovers datasets. The names or identifiers of linked Web pages in Univ-6 are at least as informative as words (local, linked or tagged) in those pages for the classification task; however, they do not provide as much information for the Hoovers classification tasks.

4.2.5. Meta Data Regularity The meta data we report results from is the Competitors data. We treat the competitor information in the same way as we do the hyperlinks and use the meta data in two ways: as category labels (Preclassified Regularity) and as links between pages of the same class (Encyclopedia Regularity). In the case of the Preclassified Regularity, we use only the names of the competitors and find a sharp boost in performance for both Hoovers-28 and Hoovers-255. For the Encyclopedia Regularity, we use the words from the competitors in the same way as we use the words from hyperlinked neighbors. Since the two representations yielded an almost equal performance boost for our classifiers, which was reported in a previous paper [12], we only include the results for *Competitor Names* in Tables 2, 3 and Figure 2. Evidently, the competitor information is more useful than any other representations we examined for classifying the Hoovers Web sites, including the local text in a page or the linkage among pages. A detailed analysis reveals that 70% of the pairs of competitors share the same class label, which is much higher than the 6.5% of the hyperlinked pairs sharing the same class. Another point to note is that using the *names* of competitors as the hypertext representation had much smaller vocabulary size than using the *words* in competitor pages, and thus making it much more efficient to train and test the classifier based on competitor names.

As for the other types of meta data, including the text in the HTML meta and title fields in Web pages, our results show that they are quite informative for the classification tasks although not as predictive as *Competitor Names* and *Page Only* for all the classifiers on the Hoovers datasets, and not as good as *Linked Names* and *Tagged Words* on the Univ-6 dataset. Nevertheless, using these kinds of meta information *in addition* to Web pages (and links) can improve the classification performance than using Web pages alone, which has been shown in our experiments as reported in a previous paper [12].

4.3. Performance Analysis with respect to Feature Selection

The results discussed so far are for approximately optimal vocabulary sizes we found in feature selection. For NB we used *Information Gain* as the feature selection criterion because our NB system supported this functionality. For kNN we used χ^2 statistics (after removing the words which occur only once or twice in the training set) because we found this worked slightly better than *Information Gain* for our kNN in a previous study[29]. For FOIL we used *Document Frequency* to rank and select words, for the reason discussed in Section 2.3.3. Since the original vocabulary sizes of the Hoovers datasets are very large for some representations (over 300,000 terms for *Tagged Words*, for example), it would take too long to test each classifier with the full vocabulary sizes for these representations. We then only tested each classifier with subsets of features with increasing sizes until the performance curve of that classifier approached a stable plateau.

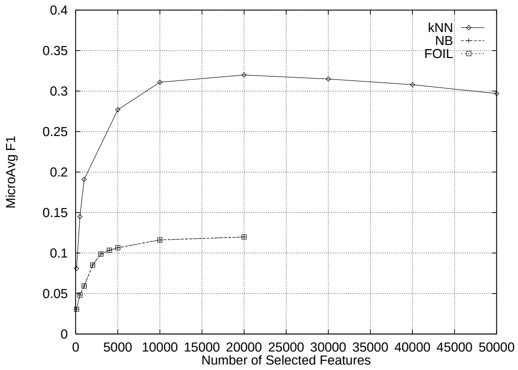
Figure 3 shows the curves in micro-averaged F_1 for our classifiers with three representations: *Page Only*, *Linked Words* and *Tagged Words*. We omit the macro-averaged curves and the results for other representations because of space limitations; we also omit the curves on Hoovers-28 which are similar in shape to those on Hoovers-255. Notice that we do not have the curves for FOIL with the *Linked Words* representation because this method always treats words in linked pages as *Tagged Words* by definition.

We find that the observed performance variations over the cross product of the datasets, representations and classifiers are larger than those reported in feature selection for conventional text categorization[29, 13]. Not only do the performance curves of our classifiers peak with very different vocabulary sizes, the shapes of those curves also show a larger degree of diversity than those previously reported. Perhaps the highly noisy nature of Web pages makes feature selection more important for performance optimization in hypertext classifiers. The inconsistent shapes of the curves for NB suggests a potential difficulty in obtaining stable or optimized performance for this method on highly noisy and heterogeneous hypertext collections. FOIL exhibited smaller performance variances with respect to vocabulary-size changes, but larger performance differences across datasets and when switching from micro-averaged measures to macro-averaged measures.

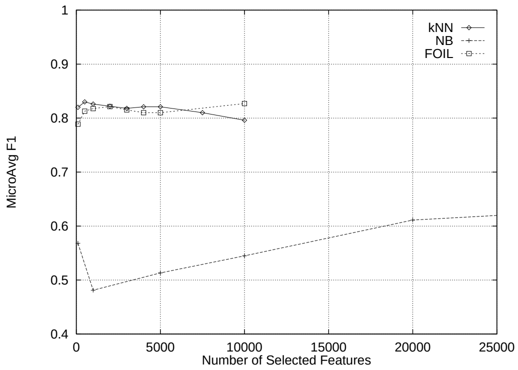
4.4. Recall-Precision Trade-off

In addition to evaluating our classifiers at a single point using the F_1 metric, we also examined their potential for making flexible trade-offs between recall and precision. In practice, it is important to know whether or not a classifier can produce either high-precision decisions, or conversely high-recall output, depending on the type of real-world application being addressed. Cross-classifier comparison also allows us to get a deeper understanding of the strengths and weaknesses of our methods in hypertext classification.

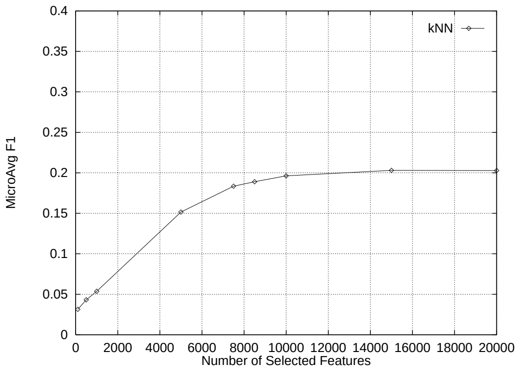
Figure 4 shows the recall-precision trade-off curves for NB, kNN and FOIL on Hoovers-28, Hoovers-255 and Univ-6. The micro-averaged and macro-averaged



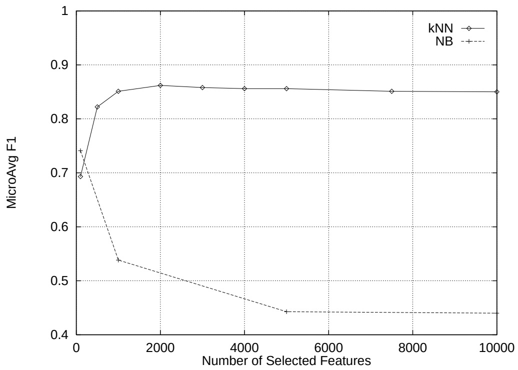
(a) Classifiers on Hoovers-255 *Page Only*



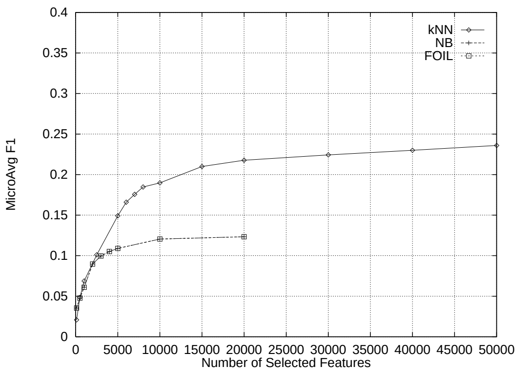
(b) Classifiers on Univ-6 *Page Only*



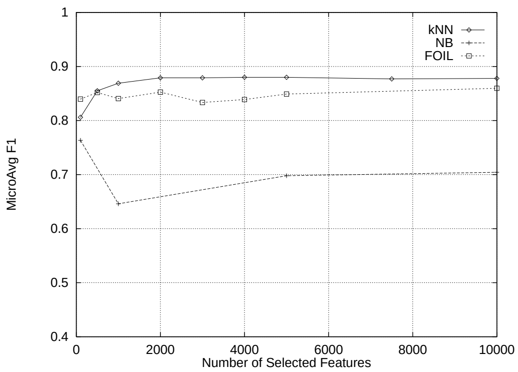
(c) Classifiers on Hoovers-255 *Linked Words*



(d) Classifiers on Univ-6 *Linked Words*



(e) Classifiers on Hoovers-255 *Tagged Words*



(f) Classifiers on Univ-6 *Tagged Words*

Figure 3. Performance of classifiers with respect to feature selection.

curves are compared side-by-side. For each classifier and dataset, we present its curve for the representation with which the averaged F_1 score of this classifier is optimized. For example, on the Univ-6 dataset, *Tagged Words* is the choice for FOIL and *Linked Names* is the choice for kNN and NB. For the Hoovers datasets, on the other hand, *Competitor Names* is the choice for all the three classifiers.

These curves were generated by thresholding over the system-generated ranks and scores of the candidate categories for test documents, by the following procedure:

1. All the candidate categories are first ordered by their rank ($R = 1, 2, 3 \ldots$ for each test document) – the higher the rank, the more relevant they are considered.

2. For the candidate categories with the same rank, their scores are used for further ordering.

3. Move (automatically) the threshold over the candidate categories, compute the recall and precision values for each threshold, and interpolate the resulting plots.

A *break-even* line is shown in each graph for reference, on which the recall and precision are equally valued, and around which the F_1 values are typically optimized. The interesting observations from these graphs are:

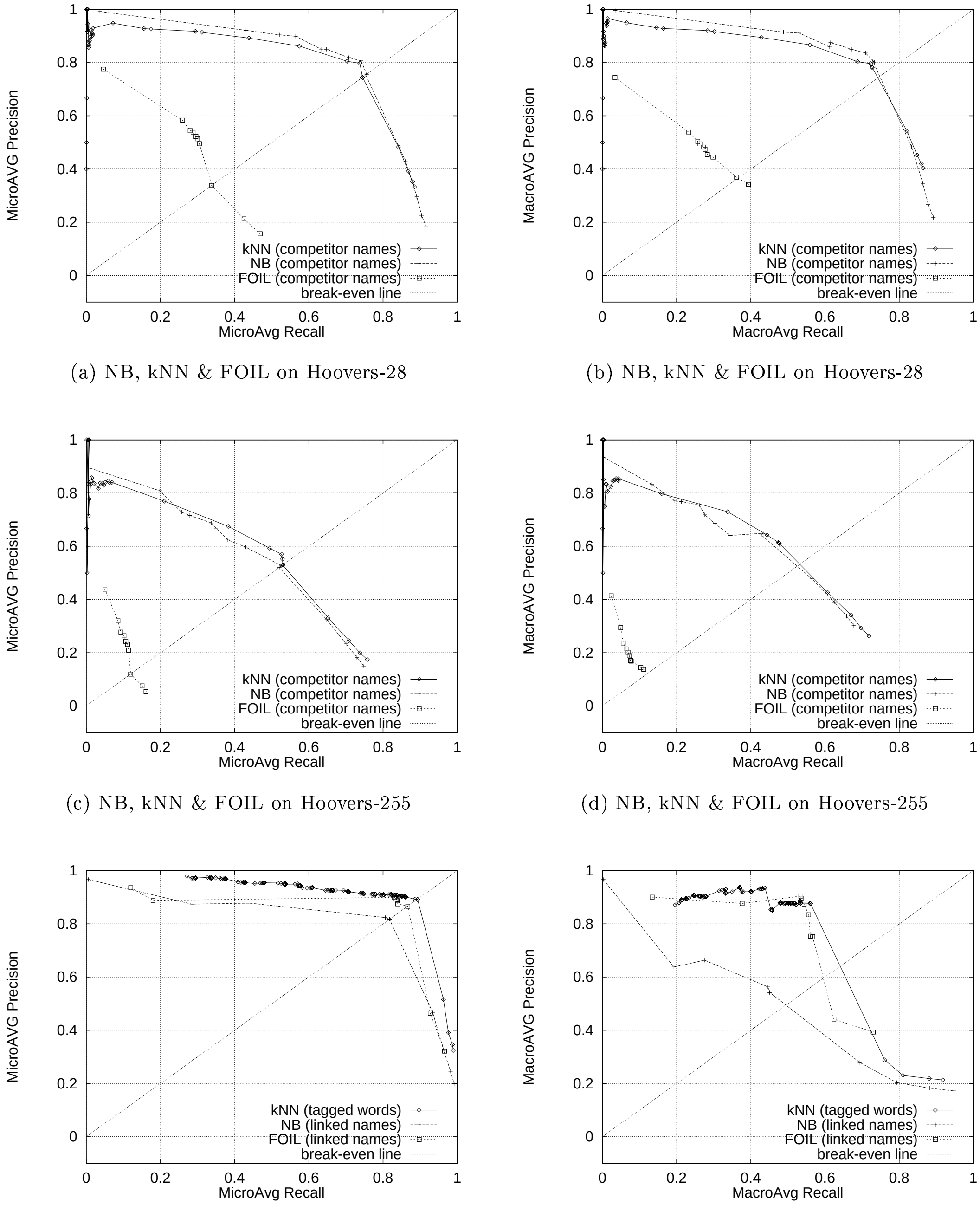
- Both kNN and NB are flexible in trading-off recall and precision from the high-precision extreme to the high-recall extreme, and in both micro-averaged and macro-averaged evaluations.

- FOIL's curves exhibit a large performance variance across datasets. Its curves on Univ-6 are competitive with kNN's in both micro-averaged and macro-averaged measures. However, it has difficulty in getting high-recall results for the Hoovers task. Improving the pruning strategy for rules in FOIL and inventing richer scoring schemes may be potential solutions for this kind of problem, which requires future investigation. It may also worth mentioning that Univ-6 has the most skewed distribution of categories, and performance on the largest class "Other" (miscellaneous) tends to dominate the overall performance of a classifier on Univ-6 if the system is not sufficiently sensitive to rare categories.

- The different performance curves of these classifiers suggest an intriguing potential of combining these classifiers for a classification system that is more robust than using each method alone. Similar questions have been raised in both conventional text categorization and hypertext mining [28, 9, 21]; how to solve it for better hypertext categorization also requires future research.

4.5. Run-Time Observation

Table 4 shows the running times (including training and testing) in CPU minutes for our classifiers in some of the experiments. When using the representations of *Linked Names* or *Competitor Names*, all the classifiers were very fast and we omit



(a) NB, kNN & FOIL on Hoovers-28

(b) NB, kNN & FOIL on Hoovers-28

(c) NB, kNN & FOIL on Hoovers-255

(d) NB, kNN & FOIL on Hoovers-255

(e) NB, kNN & FOIL on Univ-6

(f) NB, kNN & FOIL on Univ-6

Figure 4. Performance in Recall-Precision Trade-off.

Table 4. Average running times in CPU minutes for each algorithm with different representations.

	Page Only			Linked Words		Tagged Words		
	NB	kNN	FOIL	NB	kNN	NB	kNN	FOIL
Hoovers-28	1.5	5.0	288.4	2	23	2.3	12	304.3
Hoovers-255	3	2.5	315.6	3.5	3.1	3.5	3.9	274.5
Univ-6	0.07	0.97	2.8	0.1	2.8	0.27	1.3	10.25

the time for those runs; on the other hand, for the representations of *Page Only*, *Linked Words* and *Tagged Words*, the computations were much more intensive due to the large vocabulary sizes. Table 4 shows the times for the latter. Since we used different machines for running those classifiers, the computation times are not directly comparable, but rather indicative for a rough estimate. On average, Naive Bayes experiments run faster than kNN while FOIL takes considerably longer than both kNN and Naive Bayes.

5. Concluding Remarks

The rich information typically available in hypertext corpora makes the classification task significantly different from traditional text classification. In this paper, we presented the most comprehensive examinations to date, addressing some open questions of how to effectively use hypertext information by examining the cross product of explicitly defined hypertext regularities (five), alternative representations (eight), multiple established hypertext datasets (three) from different domains, and several well-known supervised classification algorithms (three). This systematic approach enabled us to explicitly analyze potential reasons behind the observed performance variance in hypertext classification, leading toward generalizable conclusions. Our major findings include:

- The identification of hypertext regularities (Section 2.1) in the data and the selection of appropriate representations for hypertext in particular domains are crucial for the optimal design of a classification system. The most surprising observations are that *Linked Names* are at least as informative as words (local words, linked words or target words all together) on Univ6, and that *Competitor Names* are more informative than any other alternative representations we explored on the Hoovers datasets. These observations suggest that *Preclassified Regularity* strongly influences the learnability of those problems, although for one problem it appears in hyperlinks, and for the other problem it exhibits in meta data beyond the Web pages themselves.

- The "right" choice of hypertext representation for a real-world problem is crucial but seldom obvious. Adding linked words (tagged or untagged) to a local page, for example, improved classification accuracy on the Univ-6 dataset for all three classifiers, but had the opposite effect on the performance of NB and kNN on the Hoovers datasets (which is consistent with previously reported results by

Chakrabarti et al. and Oh et al. on different data). Moreover, *Linked Names* and *Tagged Words* were almost equally informative for all the three classifiers on Univ-6; but this phenomenon was not observed from the experiments on the Hoovers datasets. Our extensive experiments over several domains show that drawing general conclusions for hypertext classification without examinations over multiple datasets can be seriously misleading.

- Meta data about Web pages or Web sites can be extremely useful for improving the classification performance, as shown by the Hoovers Web site classification tasks. This suggests the importance of examining the availability of meta data in the real world, and exploiting information extraction techniques for automated acquisition of meta data. Recognizing useful HTML fields in hypertext pages and using those fields jointly in making classification decisions can also improve classification performance, which is evident in our experiments on Univ-6.

- kNN and NB, with extended document representations combining within-page words, linked words and meta data in a naive fashion, show simple and effective ways of exploiting hypertext regularities. Their simplicity as algorithms allows them to scale well for very large feature spaces, and their relatively strong performance (with the "right choice" for hypertext representation) across datasets makes them suitable choices for generating baselines in comparative studies.

- Algorithms focusing on automated discovery of the relevant parts of the hypertext neighborhood should have an edge over more naive approaches treating all the links and linked pages without distinction. FOIL, with the power for discovering relational regularities, gave mixed results in this study, indicating the discovery problem to be non-trivial especially given the noisy nature of links and inviting future investigation for improving this algorithm and on other algorithms of this kind.

- The use of micro-average and macro-average together is necessary to actually understand the results and relative performance of classifiers. This is of special significance for datasets where the category distribution is extremely skewed. Also, investigating the precision-recall tradeoff is important in order to observe the performance of classifiers in specific regions of interest. This issue becomes essential for hypertext applications where high precision results are extremely important.

While the scope of the experimental results presented is necessarily confined to our datasets, we hope that our analysis will help future research into hypertext classification by defining explicit hypotheses for various types of regularities that may be present in a hypertext corpus, by presenting a systematic approach to the examination of those regularities, and by providing some ideas about how one should construct a classifier to take advantage of each.

Acknowledgments

The authors would like to thank Bryan Kisiel for his significant help in improving the evaluation tools and applying them to the output of different classifiers, and Thomas Ault for editorial assistance. This study was funded in part by the National Science Foundation under the grant number KDI-9873009.

Notes

1. Thorsten Joachims provided the name and the intuition behind this regularity (personal communication).

References

1. Soumen Chakrabarti, Byron E. Dom, and Piotr Indyk. Enhanced hypertext categorization using hyperlinks. In Laura M. Haas and Ashutosh Tiwary, editors, *Proceedings of SIGMOD-98, ACM International Conference on Management of Data*, pages 307–318, Seattle, US, 1998. ACM Press, New York, US.
2. Hao Chen and Susan T. Dumais. Bringing order to the Web: automatically categorizing search results. In *Proceedings of CHI-00, ACM International Conference on Human Factors in Computing Systems*, pages 145–152, Den Haag, NL, 2000. ACM Press, New York, US.
3. William Cohen. Learning to Classify English Text with ILP Methods. In L. De Raedt, editor, *Advances in Inductive Logic Programming.* IOS Press, 1995.
4. William Cohen. Automatically extracting features for concept learning from the web. In *Seventeenth International Conference on Machine Learning*, June 2000.
5. William W. Cohen. Learning to classify English text with ILP methods. In Luc De Raedt, editor, *Advances in inductive logic programming*, pages 124–143. IOS Press, Amsterdam, NL, 1995.
6. Mark Craven, Dan DiPasquo, Dayne Freitag, Andrew K. McCallum, Tom M. Mitchell, Kamal Nigam, and Seán Slattery. Learning to construct knowledge bases from the World Wide Web. *Artificial Intelligence*, 118(1-2):69–113, 2000.
7. Mark Craven, Seán Slattery, and Kamal Nigam. First-order learning for web mining. In *Tenth European Conference on Machine Learning*, April 1998.
8. Belur V. Dasarathy. *Nearest Neighbor (NN) Norms: NN Pattern Classification Techniques.* McGraw-Hill Computer Science Series. IEEE Computer Society Press, Las Alamitos, California, 1991.
9. Dayne Freitag. Multistrategy learning for information extraction. In *Proceedings of the Fifteenth International Conference on Machine Learning*, pages 161–169, San Francisco, 1998. Morgan Kaufmann.
10. Johannes Fürnkranz. Exploiting structural information for text classification on the WWW. In David J. Hand, Joost N. Kok, and Michael R. Berthold, editors, *Proceedings of IDA-99, 3rd Symposium on Intelligent Data Analysis*, pages 487–497, Amsterdam, NL, 1999. Springer Verlag, Heidelberg, DE. Published in the "Lecture Notes in Computer Science" series, number 1642.
11. Rayid Ghani, Rosie Jones, Dunja Mladenic, Kamal Nigam, and Sean Slattery. Data mining on symbolic knowledge extracted from the web. In *Workshop on Text Mining at the Sixth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2000.
12. Rayid Ghani, Sean Slattery, and Yiming Yang. Hypertext categorization using hyperlink patterns and meta data. In *Proceedings of ICML-01, 18th International Conference on Machine Learning*, Williams College, US, 2001. Morgan Kaufmann Publishers, San Francisco, US.

13. Thorsten Joachims. Text categorization with support vector machines: learning with many relevant features. In Claire Nédellec and Céline Rouveirol, editors, *Proceedings of ECML-98, 10th European Conference on Machine Learning*, pages 137–142, Chemnitz, DE, 1998. Springer Verlag, Heidelberg, DE. Published in the "Lecture Notes in Computer Science" series, number 1398.
14. Thorsten Joachims, Nello Cristianini, and John Shawe-Taylor. Composite kernels for hypertext categorisation. In *Proceedings of ICML-01, 18th International Conference on Machine Learning*, Williams College, US, 2001. Morgan Kaufmann Publishers, San Francisco, US. Forthcoming.
15. Jon Kleinberg. Authoritative Sources in a Hyperlinked Environment. In *Proceedings of the Nineth Annual ACM-SIAM Symposioum on Discrete Algorithms*, 1998.
16. David D. Lewis. Naive (Bayes) at forty: The independence assumption in information retrieval. In Claire Nédellec and Céline Rouveirol, editors, *Proceedings of ECML-98, 10th European Conference on Machine Learning*, pages 4–15, Chemnitz, DE, 1998. Springer Verlag, Heidelberg, DE. Published in the "Lecture Notes in Computer Science" series, number 1398.
17. Andrew McCallum and Kamal Nigam. A comparison of event models for naive Bayes text classification. In *AAAI-98 Workshop on Learning for Text Categorization*, 1998. Tech. rep. WS-98-05, AAAI Press.
18. Hyo-Jung Oh, Sung Hyon Myaeng, and Mann-Ho Lee. A practical hypertext categorization method using links and incrementally available class information. In Nicholas J. Belkin, Peter Ingwersen, and Mun-Kew Leong, editors, *Proceedings of SIGIR-00, 23rd ACM International Conference on Research and Development in Information Retrieval*, pages 264–271, Athens, GR, 2000. ACM Press, New York, US.
19. J. R. Quinlan. Learning logical definitions from relations. *Machine Learning*, 5:239–266, 1990.
20. G. Salton and C. Buckley. Term weighting approaches in automatic text retrieval. *Information Processing and Management*, 24:513–523, 1988.
21. S. Slattery and M. Craven. Combining statistical and relational methods for learning in hypertext domains. In *Proceedings of the 8th international Conference on Inductive Logic Programming*, Madison, WI, 1998.
22. Seán Slattery. *Hypertext Classification*. PhD thesis, Carnegie Mellon University, 2001.
23. Seán Slattery and Mark Craven. Discovering test set regularities in relational domains. In Pat Langley, editor, *Proceedings of ICML-00, 17th International Conference on Machine Learning*, Stanford, US, 2000. Morgan Kaufmann Publishers, San Francisco, US.
24. C.J. van Rijsbergen. *Information Retrieval*. Butterworths, London, 1979.
25. Yiming Yang. Expert network: effective and efficient learning from human decisions in text categorisation and retrieval. In W. Bruce Croft and Cornelis J. van Rijsbergen, editors, *Proceedings of SIGIR-94, 17th ACM International Conference on Research and Development in Information Retrieval*, pages 13–22, Dublin, IE, 1994. Springer Verlag, Heidelberg, DE.
26. Yiming Yang. An evaluation of statistical approaches to text categorization. *Information Retrieval*, 1(1-2):69–90, 1999.
27. Yiming Yang. A study on thresholding strategies for text categorization. In W. Bruce Croft, David J. Harper, Donald H. Kraft, and Justin Zobel, editors, *Proceedings of SIGIR-01, 24th ACM International Conference on Research and Development in Information Retrieval*, pages 137–145, New Orleans, US, 2001. ACM Press, New York, US.
28. Yiming Yang, Thomas Ault, and Thomas Pierce. Combining multiple learning strategies for effective cross-validation. In Pat Langley, editor, *Proceedings of ICML-00, 17th International Conference on Machine Learning*, pages 1167–1182, Stanford, US, 2000. Morgan Kaufmann Publishers, San Francisco, US.
29. Yiming Yang and Jan O. Pedersen. A comparative study on feature selection in text categorization. In Douglas H. Fisher, editor, *Proceedings of ICML-97, 14th International Conference on Machine Learning*, pages 412–420, Nashville, US, 1997. Morgan Kaufmann Publishers, San Francisco, US.