Web Content Categorization Using Link Information

Zoltán Gyöngyi* and Hector Garcia-Molina†
Department of Computer Science, Stanford University

Jan Pedersen‡
Yahoo! Inc.

(Dated: August 5, 2007)

Document categorization is one of the foundational problems in (web) information retrieval. Even though web documents are hyperlinked, most proposed classification techniques take little advantage of the link structure and rely primarily on text features, as it is not immediately clear how to make link information intelligible to supervised machine learning algorithms. This paper introduces a link-based approach to classification, which can be used in isolation or in conjunction with text-based classification. Various large-scale experimental results indicate that link-based classification is on par with text-based classification, and the combination of the two offers the best of both worlds.

1. INTRODUCTION

Knowledge of the general topics of individual documents in a collection can greatly enhance information retrieval. More generally, partitioning a collection along different categorical axes (topic is one possibility) could prove useful in various scenarios. On the World Wide Web, for instance, shopping comparison services, such as Froogle,[1] need to distinguish e-commerce web sites from the rest. As another example, emerging technologies, such as blog and RSS feed aggregation, could benefit from topic-based categorization. Finally, the research community has demonstrated the utility of implicit categorization in personalizing search results [12, 19].

Given the size and dynamism of the Web, manual categorization is often infeasible. The natural alternative is to use various supervised or unsupervised learning algorithms to assign one of some predefined category labels to each document (*classification*) or to produce groups of related documents (*clustering*). In the classification scenario, known properties (*features*) of a web document

*Electronic address: zoltan@cs.stanford.edu
†Electronic address: hector@cs.stanford.edu
‡Electronic address: jpederse@yahoo-inc.com
[1] `http://froogle.google.com`

are used to predict its category.

Prior research produced time-tested, effective text-based classification techniques. To complement text-based classification, some existing approaches incorporate access or link pattern data (e.g., number of in- and outlinks), or neighbor document contents as well. An alternative approach is pioneered in [5], which classifies a document and its neighbors within a given (small) radius using iterative relaxation labeling. Accordingly, in each iteration, category predictions are generated based on a document's textual content and the previously-assigned categories of the neighbors. While elegant in formulation, this approach relies on a classification model that is hard to generate for the Web. (For an overview of web categorization techniques, readers are referred to Chapter 5 of [4]. We further discuss [5] and other related work in Section 8.)

Instead of analyzing text or other page features, in this paper we propose a global approach to harvesting web link structure in classification, which performs well on large data sets. We start with a set $\mathcal{S}$ of web documents of known category. (Such *a priori* knowledge can be accumulated, for instance, by having a human editor manually label some of the documents.) Then, we automatically predict the category of some previously unlabeled web document x by considering paths from the documents in $\mathcal{S}$ to x. Our prediction model is based on measuring how strongly x is connected to groups of documents of known category i (the influence of i on x), and on comparing the relative influences to each other. We measure influence by propagating influence scores over the web graph, essentially employing biased forms of PageRank. While similar biased PageRank scores have been used to increase the topical sensitivity of search results [12] and quality of ranking [17], to our best knowledge our work is first in using it for classification. The contribution of the paper consists in the novel way in which we combine well-known, scalable link analysis and classification algorithms, as opposed to introducing some new machine learning technique.

The first part of the paper presents the concepts behind the proposed classification method. In Section 2 we use connectivity information and a variety of graph propagation algorithms to produce a numeric vector of link-based features for each web document. Section 3 then introduces the process of creating, evaluating, and using classification models that rely on link-based feature vectors. In the second part of the paper we use data from the Yahoo! web index and the Open Directory Project[2] (described in Section 4) to evaluate link-based classification in isolation (Section 5) as well as side-by-side and combined with text-based classification (Section 6). Finally, in Section 7 we evaluate several interesting variations to our link-based classification method.

[2] `http://dmoz.org`

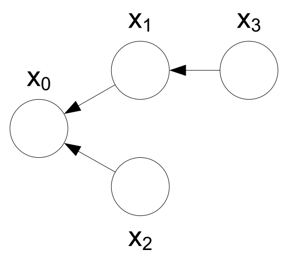

FIG. 1: A simple web graph.

2. FEATURE GENERATION

One can investigate the Web by zooming in or out across pages, hosts, or sites. For the moment, let us abstract from the particular level of granularity and use the term *node* to refer to either pages, hosts, or sites. Our starting point is the assumption that links between nodes on the Web indicate some kind of relationship. Consider, for example, the simple graph in Figure 1. Intuitively, if we happen to know the category of nodes x_1 and x_2, that could help us infer the category of node x_0, as the latter has incoming links (*inlinks*) from the former two. For instance, if both x_1 and x_2 are blogs of ski enthusiasts, it is fairly probable that x_0 is about skiing as well, or is at least of interest to skiers. Conversely, knowing that x_3 has an *outlink* to ski blog x_1 might help us infer the category of x_3. We will build our classification technique around these intuitions.

Obviously, in a real-world setting we may expect to know in advance the category of only a few web nodes. Therefore, we would like to be able to reason about the category of x_0 even if we only have information about more distant *in-neighbors*, such as x_3. It turns out that a reinterpretation of the popular PageRank algorithm comes to our help—the rest of this section discusses exactly how.

2.1. Node Influence

We wish to establish a measure of connectedness between pairs of nodes in the web graph. Accordingly, we propose the following diffusion (score propagation) process: each node x is given initially a certain amount of credit, which then gets distributed among the nodes that can be reached from x following the links between nodes. The shorter the path from x to a node y and the fewer the nodes reachable from x, the larger the fraction of x's credit that gets propagated to y. The exact influence of a node x on some node y can then be quantified as follows.

The connection between the nodes x and y is captured by the concept of a walk. A *walk* W from x to y in a directed graph is defined as a finite sequence of nodes $x = x_0, x_1, \ldots, x_k = y$, where there is a directed edge (x_i, x_{i+1}) between every pair of adjacent nodes x_i and x_{i+1}, $i = 0, \ldots, k-1$.

Let $\mathbf{v}$ represent the vector of initial credits given to the n nodes of the web graph, usually $\mathbf{v} = (1/n)_n$. We define the *influence* of x on y *over the walk* W as

$$q_y^W = c^k \pi(W)(1-c)v_x,$$

where c is a damping factor (between 0 and 1) and $\pi(W)$ is the *weight* of the walk[3]:

$$\pi(W) = \prod_{i=0}^{k-1} \frac{1}{\mathrm{out}(x_i)} \,.$$

This weight can be interpreted as the probability that a Markov chain of length k starting in x reaches y through the sequence of nodes $x_1, \ldots, x_{k-1}$.

In a similar manner, we define the total influence of x on y, $x \neq y$, over the (possibly infinite) set $\mathcal{W}_{xy}$ of *all walks* from x to y (or simply: the influence of x on y) as

$$q_y^x = \sum_{W \in \mathcal{W}_{xy}} q_y^W \,.$$

The influence of a set of nodes $\mathcal{A}$ on y is

$$q_y^{\mathcal{A}} = \sum_{x \in \mathcal{A}} q_y^x \,.$$

It turns out that the presented concept of influence is strongly related to PageRank. Consider the PageRank score of a node y, defined as

$$p_y = c \sum_{z:\exists(z,y)} \frac{p_z}{\mathrm{out}(z)} + (1-c)v_y \,.$$

Let $\mathcal{V}$ denote the set of all web nodes. Then, p_y can be written in terms of influences on y as

$$p_y = \sum_{x \in \mathcal{V}} q_y^x \,,$$

that is, the PageRank of a node y is the sum of all influences of other nodes on y.[4]

For a fixed web graph and c, the vector $\mathbf{p}$ of PageRank scores is a function of $\mathbf{v}$ only, that is, $\mathbf{p} = \mathrm{PR}(\mathbf{v})$. It is easy to verify that the influence of a set of nodes $\mathcal{A} \subseteq \mathcal{V}$ on all nodes can be

[3] $\mathrm{out}(x)$ is the outdegree of node x.
[4] For a proof, see either [13] or [11].

computed as $\mathbf{q}^{\mathcal{A}} = \mathrm{PR}(\mathbf{v}^{\mathcal{A}})$ where

$$v_x^{\mathcal{A}} = \begin{cases} v_x, & \text{if } x \in \mathcal{A}, \\ 0, & \text{otherwise.} \end{cases}$$

2.2. Variations

There are several interesting variations to the influence computation scheme we have just presented. In this section we describe some of these variations, which will then be studied in Section 7.

In our initial definition of node influence we considered all possible paths between x and y. One could argue that node influence becomes irrelevant after a certain number of propagation steps. Accordingly, we might want to limit ourselves to paths of up to a certain length k. We can introduce a corresponding concept of *limited-distance influence* as follows. Consider the paths from x to y that are of length at most $k \geq 1$, $\mathcal{W}_{xy}^{(k)}$. Then the distance-k influence of x on y is:

$$\tilde{q}_y^x = \sum_{W \in \mathcal{W}_{xy}^{(k)}} q_y^W ,$$

and the distance-k influence of a set of nodes $\mathcal{A}$ on y is

$$\tilde{q}_y^{\mathcal{A}} = \sum_{x \in \mathcal{A}} \tilde{q}_y^x .$$

Distance-k influence can be computed by performing an iterative PageRank computation and stopping it after k iterations. Correspondingly, $\tilde{\mathbf{q}}^{\mathcal{A}} = \mathrm{PR}^{(k)}(\mathbf{v}^{\mathcal{A}})$, where $\mathrm{PR}^{(k)}$ stands for the k-iteration PageRank.

Note how our current influence propagation is unidirectional, always following the direction of the links. That is, a node x influences other nodes that it links to and a node y is influenced by nodes that link to it (its in-neighbors). We may want to propagate influence in the reverse direction, so that y would be influenced by its out-neighbors. This corresponds to our previous example from Figure 1 when we inferred the category of x_3 knowing that it links to ski blog x_1. We can easily define and compute *reverse influence* by considering the walks $\bar{\mathcal{W}}_{xy}$ from x to y on the reverse web graph, in which the direction of every link is reversed.

A final variation we examine in Section 7 is to add weights to the edges in the graph. For example, the weight on an edge between two nodes of a host-level web graph could represent the number of actual links between the two hosts. The formulas of Section 2.1 can be extended in a straightforward way to take the weights into account.

2.3. Category Influence

Influences let us quantify the connection between a group of nodes $\mathcal{A}$ and some node x, as measured by the total credit propagated from the nodes in $\mathcal{A}$ to x. For instance, if we are given a collection of web nodes about skiing, we can easily compute how much these nodes contribute to the PageRank of an arbitrary other web node x. A significant contribution might then indicate that x has much to do with skiing.

Moving one step further, given two (disjoint) sets of nodes $\mathcal{A}$ and $\mathcal{B}$, we can compare the relative influences of the two sets upon x, and determine which is more pronounced. If $\mathcal{A}$ is made up of arts nodes while $\mathcal{B}$ is made up of science nodes, comparing $q_x^{\mathcal{A}}$ to $q_x^{\mathcal{B}}$ would let us infer whether the topic of x is more biased toward the arts or the sciences.

In general, we can expect to have *a priori* knowledge of some representative *seed* nodes for each category (constituting the seed set $\mathcal{S}_i$ for category i), while not having category information about the majority of the nodes. We propose to categorize the nodes x of unknown category by estimating the *category influence* scores $q_x^{\mathcal{S}_i}$ for each node x and category i, and using these estimates in supervised learning. (For simplicity, we will write q_x^i instead of $q_x^{\mathcal{S}_i}$ from now on.)

For each node $x \in \mathcal{V}$, we construct a *profile* $\mathbf{r}_x$, a numeric vector of length m, consisting of the influences of different categories on x:

$$\mathbf{r}_x = \left(q_x^1,\ q_x^2,\ \ldots,\ q_x^m\right).$$

Profiles then represent *link-based features* of web nodes, which are used as inputs in machine learning (to build a category prediction model) and then in model-based categorization. (The concept of profiles has been introduced independently by Nie *et al.* in [17], but their focus is improved ranking as opposed to categorization.) Note that profiles may be the only features used, or may be accompanied by other, for instance, text-based features. Sections 5 and 6 cover both scenarios in turn.

3. CLASSIFICATION PROCESS

Given the set of web nodes $\mathcal{V}$, a list of categories (for instance, the high-level topics of nodes, such as *Arts*, *Business*, or *Science*), and the corresponding seed sets, our goal is to build a classification model (classifier) that would predict the category of each web node based on its profile. Category predictions could then be used in various web information retrieval settings, for instance, to alleviate the problem of keyword ambiguity. This section presents the steps we propose to reach our goal.

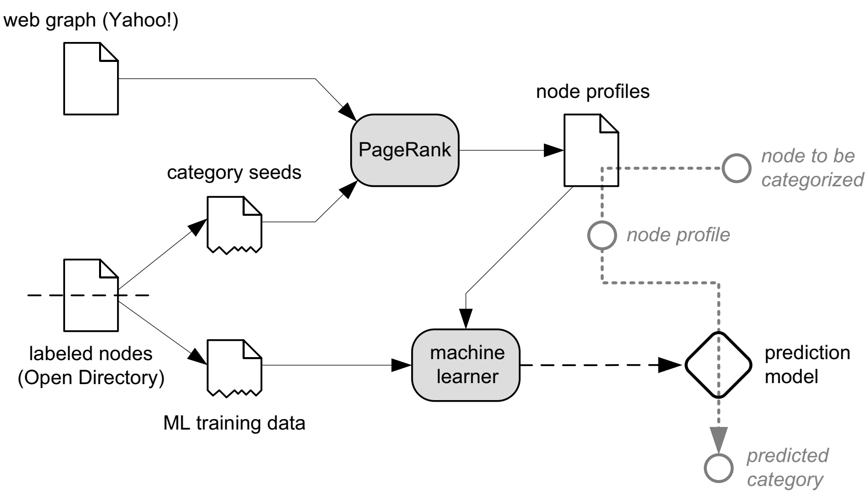

FIG. 2: Link-based classification steps.

3.1. Input Data

Consider Figure 2, a schematic representation of the data and algorithms involved in our categorization problem. The two type of input data are:

- A directed web graph, capturing the links between nodes, which is used in the computation of category influence scores. As discussed in Section 4, we used the host-level Yahoo! web graph in our experiments.

- A set $\mathcal{U}$ of nodes with categories known *a priori*. This data is most often the product of the manual labeling of a subset of web nodes by human editors. Large sets of labeled nodes are readily available for various categorization tasks, such as topic identification or web spam detection: web directories, e.g., the Open Directory or the Yahoo! Directory[5], organize links to millions of web pages in a tree of topics and subtopics; search engines have assembled over the years black-lists and white-lists of spam and reputable pages, respectively. Our experiments focus on topic identification and rely on Open Directory data.

The set $\mathcal{U}$ serves a dual purpose: a part of it is used to construct the seed sets and another part of it helps in building and evaluating the classifier. Accordingly, the set $\mathcal{U}$ is randomly partitioned into the set of all seeds $\mathcal{S} = \mathcal{S}_1 \cup \cdots \cup \mathcal{S}_m$ and the set $\mathcal{X}$, a part of which is then used for training ($\mathcal{X}_{\text{train}}$) and another for evaluation ($\mathcal{X}_{\text{eval}}$).

[5] `http://dir.yahoo.com`

3.2. Node Profile Construction and Supervised Learning

The two main steps shown in Figure 2 as rounded boxes are (1) the computation of category influences (using a biased form of PageRank) that yields the node profiles and (2) the training of the classifier. Let us consider them in turn.

Profile construction. First, given the m seed sets $\mathcal{S}_1, \ldots, \mathcal{S}_m$ corresponding to each category, m PageRank computations produce the influence vectors $\mathbf{q}^i = \mathrm{PR}(\mathbf{v}^{\mathcal{S}_i})$. Next, the individual category influences for a node x are grouped to form the node profile $\mathbf{r}_x$, as discussed in Section 2.3.

Machine learning. At the end of the day, we wish to predict the category of a node based on its profile. The input of the machine learning algorithm is a set of training nodes with known profiles and categories. Accordingly, for each node $x \in \mathcal{X}_{\text{train}}$ of known category i, we take x's profile $\mathbf{r}_x$ and feed the pair $(\mathbf{r}_x, i)$ to the machine learning algorithm. Based on the examples, the algorithm constructs a classification model, essentially a function $\mathrm{M} : \mathcal{V} \to \{1, \ldots, m\}$ that takes as input a node, consults the node profile (and/or other relevant node features) and returns the predicted category.

Depending on the cardinality of the range of M, the machine learning literature distinguishes two main groups of classifiers [20]. On one hand, *binary classifiers* make false/true predictions and are immediately applicable when there are only two categories to distinguish between (for instance, spam versus non-spam nodes). On the other hand, *m-way* (or multiway) classifiers predict one of $m > 2$ categories.

3.3. Model Evaluation

There are many different machine learning algorithms one could use and many node features to pick from among, if one considers other sources of information in addition to node profiles. This section discusses standard ways of evaluating the performance of a given classification model and of comparing models with each other.

Suppose we have a set of $\mathcal{X}_{\text{eval}}$ of evaluation nodes. Each node $x \in \mathcal{X}_{\text{eval}}$ is labeled in advance, thus we know the *true category* i of x. Consider a classifier that we use to predict the category of x: the predictions is either correct ($\mathrm{M}(x) = i$) or incorrect ($\mathrm{M}(x) \neq i$).

Given the counts of correct and incorrect predictions of a classifier, the corresponding *confusion matrix* $\mathbf{C}$ can be used to evaluate the prediction performance. Each element C_{ij} of the matrix indicates the number of nodes with true category i that were predicted to belong to category

j. Thus, the diagonal of the matrix represents correct predictions, while off-diagonal elements represent incorrect predictions. For example, a simple confusion matrix for three categories could have the form:

$$\mathbf{C} = \begin{array}{cc} & \text{Predicted} \\ \text{True} & \begin{pmatrix} 7 & 2 & 1 \\ 3 & 8 & 0 \\ 4 & 4 & 3 \end{pmatrix} \end{array}.$$

The typical performance metrics used for evaluating a model are:

- **Error rate**. The overall error rate of a model is the ratio between the incorrectly classified nodes and all the nodes. For a confusion matrix $\mathbf{C}$,

$$\text{error rate } = 1 - \frac{\text{sum of diagonal elements of } \mathbf{C}}{\text{sum of all elements of } \mathbf{C}}.$$

  For instance, the error rate for our example is $1-(7+8+3)/(7+2+1+3+8+0+4+4+3) = 0.4375$.

- **Precision and recall**. While the error rate quantifies the overall performance of a model, precision and recall reveal the prediction performance for a specific category i. They are defined as:

$$\begin{aligned} \text{precision} &= \frac{\text{correctly classified nodes of category } i}{\text{all nodes predicted as } i} \\ &= \frac{C_{ii}}{\text{sum of all elements in column } i \text{ of } \mathbf{C}} \end{aligned}$$

  and

$$\begin{aligned} \text{recall} &= \frac{\text{correctly classified nodes of category } i}{\text{all nodes with true category } i} \\ &= \frac{C_{ii}}{\text{sum of all elements in row } i \text{ of } \mathbf{C}}. \end{aligned}$$

  For our example, the precision and recall for the first category in our example are $7/(7+3+4) = 0.5$ and $7/(7+2+1) = 0.7$, respectively.

It is common to indicate the average precision/recall of a classifier. In this paper we perform *macro-averaging*, that is, present the simple arithmetic mean over all categories of the precision/recall.

In subsequent sections we will use all the above metrics to evaluate the performance of the proposed classification models.

4. EXPERIMENTAL DATA

4.1. Input Data

The two sources of data for our experiments were the Yahoo! web graph and the Open Directory. (A second, different web graph was used to evaluate variations to our scheme in Section 7. The second graph is described in that section.)

In order to compute the category influences and build the node profiles, we used a host-level web graph generated from the Yahoo! index in 2004. The graph consisted of $n = 73{,}270{,}399$ hosts and 978,885,427 links. The main advantage of the courser granularity, i.e., using hosts as nodes as opposed to using pages, is that the data set is easier to handle. The disadvantage is that it is often harder to assign a single category to an entire host than a page, as pages on a host may belong to different categories.

The Open Directory is a hierarchy of topics organized as a tree, where tree nodes contain links to web pages relevant for the corresponding topic. We relied on the Open Directory to

- Establish the set of target categories. Accordingly, our goal was to label web hosts as belonging to one of the following $m = 14$ (possibly overlapping) categories, matching the top level of the Open Directory: *Arts*, *Business*, *Computers*, *Games*, *Health*, *Home*, *Kids*, *News*, *Recreation*, *Reference*, *Science*, *Shopping*, *Society*, and *Sports*. While the Open Directory contains two additional top-level categories, *Adult* and *Regional*, we decided to disregard them: the former because it is heavily spammed and the latter because we felt it was more of a super-category as compared to the other 14 than an equal peer.

- Obtain a list of hosts of known category. In both the construction of the seed sets for influence computation and the subsequent machine learning process we relied on the topic information available for the pages in the Open Directory. We determined the host corresponding to each listed URL, and then labeled the host with the top-level category under which the URL occurs. (Note that each host could thus have one or more labels.) We also discarded the hosts that were present in the Open Directory but not in the web graph.

Table I presents the summary statistics for the Open Directory data. In particular, it lists the top-level categories and the abbreviations we use in some of our later figures. The table shows the total number of page URLs that appear in the Open Directory in the subtree of each top-level category, the number of unique hosts the URLs map to, and the number of *matching hosts* we

Category	Abbr.	URLs	Hosts	Matching Hosts
Arts	*Ar*	296,312	136,469	112,644
Business	*Bu*	253,929	238,383	203,291
Computers	*Co*	145,388	104,935	85,657
Games	*Ga*	61,617	26,267	20,678
Health	*He*	66,718	44,264	36,649
Home	*Ho*	34,059	16,755	13,475
Kids	*Ki*	35,089	19,978	15,760
News	*Ne*	235,664	7,264	6,098
Recreation	*Rc*	120,813	84,881	69,591
Reference	*Re*	66,923	37,706	30,473
Science	*Sc*	107,132	56,377	46,139
Shopping	*Sh*	119,588	112,377	92,432
Society	*So*	271,121	153,099	126,635
Sports	*Sp*	108,606	67,467	54,703

TABLE I: Open Directory data statistics.

were able to locate in our web graph. The matching hosts are the ones used in the seed sets and in machine learning, as we have both their true category and the necessary link information.

From among the matching hosts, 46,477 individual hosts were listed under more than one category, representing 118,183 (12.9%) out of the total number of 914,225 host-category pairs. These will be referred to as the *hosts with ambiguous labeling*.

4.2. Node Profiles

We used half of the matching hosts in each category i, selected at random, to construct the seed $\mathcal{S}_i$. After computing the influence vectors, we performed two post-processing steps to make the scores easily representable in our plots: First, we scaled each category influence vector $\mathbf{q}^i$ so that the smallest score would become 1. Then, we applied a base-10 logarithm to all the scores. The influence scores presented hereafter are thus converted, typically in the range 0 to 5. Note that we only applied monotonic transformations that our machine learning algorithms were insensitive to, therefore the conversion did not affect the classification performance.

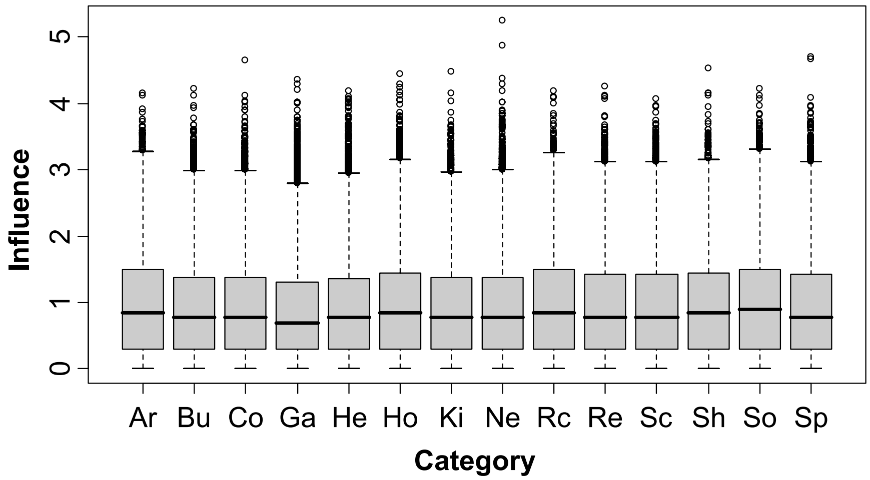

FIG. 3: Category influence distributions.

4.3. Training and Evaluation Data

The second half of the Open Directory data constituted the training and evaluation sets for machine learning. As most machine learning algorithms are sensitive to noise, we cleaned the data: in a preprocessing step, we removed

- the ambiguous hosts,[6]
- 8,124 hosts that were isolated (with all 14 influence scores equal to 0),
- and 72,863 hosts with extremely low influence scores. In general, low influences indicate the lack of evidence based on which the respective hosts could be categorized. While one can set different rules for what qualifies as low influence, we empirically determined that removing the hosts *with all 14 influence scores below 2* yields good results.

As a result of the preprocessing step, our pruned data set $\mathcal{X}$ consisted of 316,787 hosts. The box-and-whisker plot in Figure 3 summarizes all 14 category influence distributions for all hosts in $\mathcal{X}$. Bold lines indicate the medians and boxes correspond to the range between the first and third quartiles of the distributions. The dashed lines mark 150% of the interquartile range and dots represent outliers. In terms of means and variances of the influence distributions are similar for all the categories.

[6] Note that ambiguous hosts do not represent a problem in category influence computation: they cover different topics and thus it is reasonable to include them in different seeds. At the same time, the hosts used for machine learning should belong to a single category, in order to avoid contradictions in the training data.

We randomly partitioned $\mathcal{X}$ into a training set $\mathcal{X}_{\text{train}}$ containing 20% of the host-category pairs and an evaluation set $\mathcal{X}_{\text{eval}}$ containing the remaining 80%.

5. LINK-BASED CLASSIFICATION EXPERIMENTS

5.1. Baseline Classification

Let us start by analyzing the distribution of influence scores more carefully. Observe Figure 3 from Section 4.3. While, for instance, the *Arts* and *Recreation* influence scores have similar distribution over all hosts, will that be the case if we only look at the *Arts* hosts? In general, how do distributions differ from each other if we only consider hosts of a specific category?

Figure 4 sheds a light on the question, by presenting two extreme cases: the box-and-whisker plot of influence distributions for *Business* and for *Sports* hosts in $\mathcal{X}$. In both cases (and in fact for all 14 categories), the influence scores for the category the hosts belong to are larger on average than the rest. Accordingly, the darker boxes in the figure are above the lighter ones. The difference is only in the magnitude of the skew: the average *Business* influence on *Business* hosts is only slightly larger than the other category influence averages. At the same time, the *Sports* influence distribution is considerably skewed toward larger values in the case of the *Sports* hosts. The skew for the other 12 categories is in between those presented in the figure.

One cannot expect that for an arbitrary host of category i the i^{th} influence is always *maximal*, that is, larger than all the others. Nevertheless, the distribution skew indicates that there is a correlation between the maximal category influence score and the true category of a host, and that this correlation could be exploited in automatic categorization. This observation leads us to our first, naive link-based categorization model, which we present next.

Consider the set of hosts $\mathcal{X}$ and the corresponding profiles $\mathbf{r}_x$ for all $x \in \mathcal{X}$. Remember that the i^{th} element of vector $\mathbf{r}_x$ is the category influence score of x for category i, q_x^i. We define our baseline classifier $\mathrm{M}_B : \mathcal{V} \rightarrow \{0, \ldots, m\}^*$ as

$$\mathrm{M}_B(x) = \{i \mid \forall j : q_x^i \geq q_x^j\} \,,$$

that is, M_B returns the category (or categories) for which the category influence on x is maximal. (Note that we may produce an *ambiguous predictions*, that is, return several possible categories for the same host. We treat ambiguous predictions conservatively as incorrect ones.)

The prediction performance over $\mathcal{X}_{\text{eval}}$ for individual categories is shown in Figure 5 with dark gray bars representing precision and light gray bars representing recall. For instance, the precision

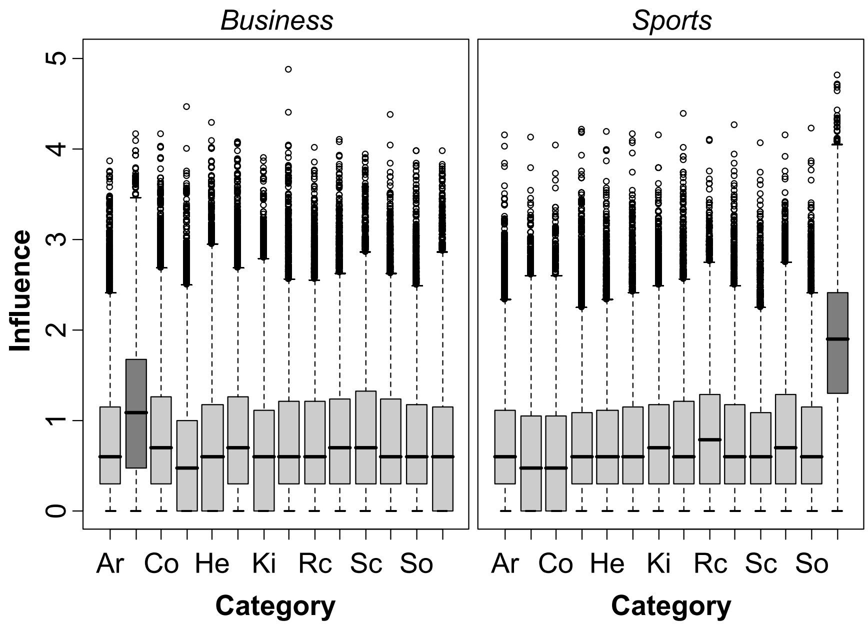

FIG. 4: Category influence distributions for *Business* (left) and *Sports* (right) hosts.

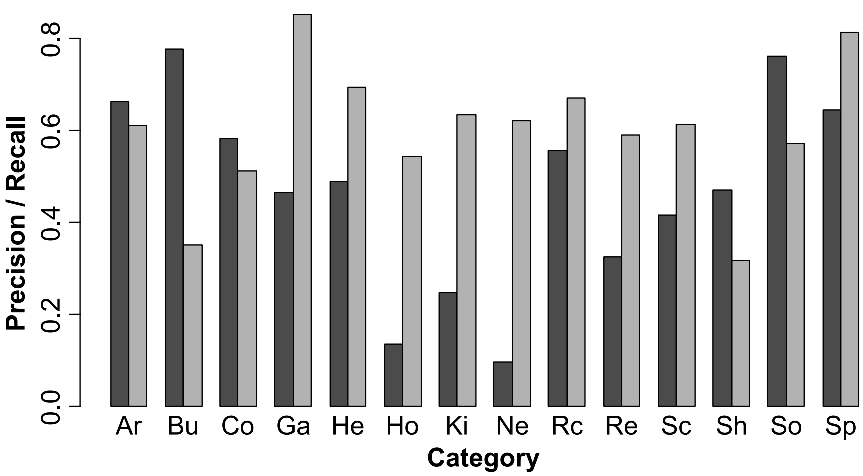

FIG. 5: Precision and recall of M_B over $\mathcal{X}_{\mathrm{eval}}$.

and recall for *Computers* hosts are 0.58 and 0.512, respectively. Note that maximal influence works best as a predictor for *Sports* hosts, while the results are quite poor for categories like *Home*, *News*, and *Shopping*. The baseline performance for *Business* hosts is reasonable in comparison to other categories, despite the small distribution skew witnessed in Figure 4.

The variations in precision and recall indicate that there is a difference in the way hosts of a specific category link to each other. The *Sports* community seems to be *cooperative*, with frequent

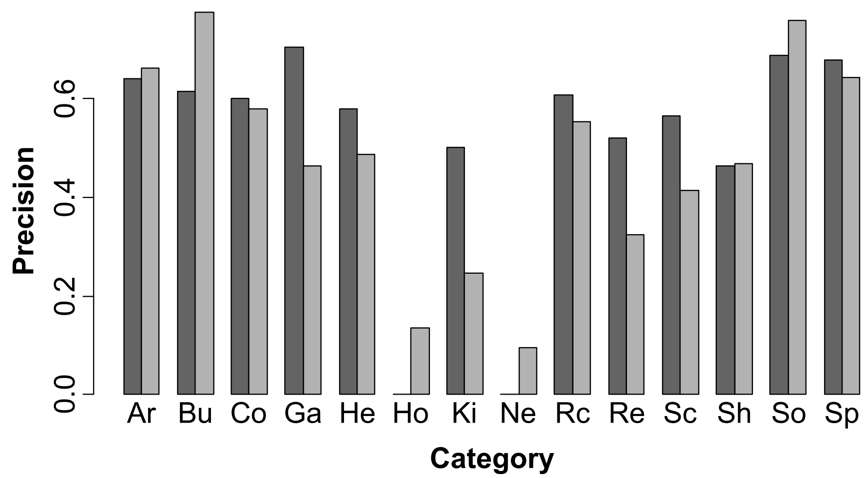

FIG. 6: Precision of M_1 (dark) and M_B (light) over $\mathcal{X}_{\text{eval}}$.

links between *Sports* hosts, whereas for instance the *Shopping* community is *competitive*, with *Shopping* hosts seldom pointing to each other.

The average precision and recall of the baseline classifier over all categories are 0.472 and 0.598, respectively. The error rate over the entire evaluation set is 0.465, where containing predictions account for an error of about 0.03.

5.2. Classification Based on Profiles

The baseline classifier lets us establish the correlation between the maximal influence score and the category of a host. We have learned, for instance, that *Sports* hosts tend to have a high *Sports* category influence. Could it be that for some category i a large influence for another category $j \neq i$ is a strong predictor? For instance, could it be that a large *Shopping* influence indicates a *Business* host? Or could a combination of high *Reference* and low *Kids* influences indicate a *Science* host? Supervised machine learning algorithms produce classification models that can account for such non-obvious relationships. We decided to use boosted decision trees and support vector machines (SVMs), and after an initial evaluation, we restricted our focus to linear SVMs. (Readers unfamiliar with popular machine learning techniques are referred to introductory texts such as [20] and [16].)

In a first step we built a 14-way classifier $M_1 : \mathcal{V} \rightarrow \{0, 1, \ldots, m\}$ based on binary linear SVMs. That is, using the host profiles $\mathbf{r}_x$ from the training data set $\mathcal{X}_{\text{train}}$, we constructed $m(m-1)/2 = 14 \cdot 13/2 = 91$ binary SVMs that distinguish between pairs of classes. A voting scheme is employed

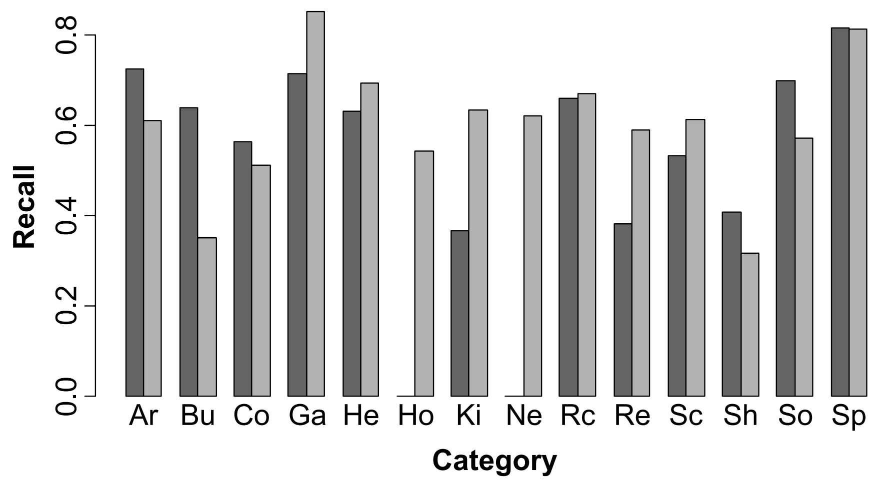

FIG. 7: Recall of M_1 (dark) and M_B (light) over $\mathcal{X}_{\text{eval}}$.

to combine the binary predictions, where the category that receives the highest number of votes is assigned to the host. (This approach is known as the *one-against-one* in machine learning literature.)

The error rate of M_1 on the evaluation data $\mathcal{X}_{\text{eval}}$ is 0.387. In other words, the classifier predicts the true class in 61.3% of the cases, outperforming the baseline classifier by correctly categorizing 7.8% more of the evaluation hosts. Figures 6 and 7 show the category-wise precision and recall for the 14-way SVM classifier (dark gray bars). For comparison, we also include the precision and recall for the baseline classifier (light gray bars). In general, the baseline classifier seems to have better precision (and worse recall) for categories with many hosts, such as *Arts*, *Business*, or *Sports*. The trend is reversed for most other categories, for which the SVM has better performance but worse recall.

Curiously, the SVM-based classifier makes incorrect predictions for *all* hosts in the *Home* and *News* categories. It is unclear whether this deficiency can be attributed to the lack of solid training data for the two categories (*Home* and *News* are two of the three categories with the fewest hosts). Observe that in the case of the third small category, *Kids*, the SVM-based classifier does a fair job.

5.3. Classification Based on Augmented Profiles

Figures 6 and 7 create the impression that somehow the SVM and the baseline classifiers are complementary. A natural step would be to leverage their combined power. Hence, we construct a new classifier as follows.

Remember, we defined the profile $\mathbf{r}_x$ of a host x as

$$\mathbf{r}_x = (q_x^1, q_x^2, \ldots, q_x^m)\,,$$

where q_x^i represents the category influence of x for category i. In a similar fashion, we introduce the *augmented profile* $\mathbf{r}'_x$ of a host x,

$$\mathbf{r}'_x = (q_x^1, q_x^2, \ldots, q_x^m, \boxed{\mathrm{argmax}_i\, q_x^i})\,,$$

that is, we add an extra (boxed) field to the profile that indicates *which* category influence on x is the largest. For instance, if the profile of host x is

$$\mathbf{r}_x = (2, 4, 2, 6, 3, 4, 5, 1, 1, 3, 2, 2, 1, 4)$$

then its augmented profile would be

$$\mathbf{r}'_x = (2, 4, 2, 6, 3, 4, 5, 1, 1, 3, 2, 2, 1, 4, \boxed{4})\,,$$

where the last (boxed) field, having the value 4, indicates that the largest category influence, namely 6, corresponds to the 4^{th} category.

We built a 14-way classifier M_2 based on new binary linear SVMs that use the augmented profiles as input data. Accordingly, the 91 binary SVMs have and extra piece of evidence based on which they can make their predictions.

The performance of M_2 turns out to be below expectation. It's error rate on the training set is 0.397, so it performs worse than the previous classifier M_1 by 0.01. Figures 8 and 9 show the precision and recall achieved by M_2 for each class (dark gray bars), respectively. For comparison, the precision and recall figures for M_1 (medium gray bars) and the baseline classifier M_B (light gray bars) are indicated as well.

The figures indicate that the precision and recall numbers for M_2 and M_1 are almost identical for most of the categories. The only categories in case of which M_2 indeed has improved precision are the "problematic" *Home* and *News*. At the same time, the extremely low recall for these two categories do not warrant much enthusiasm. Overall, the use of augmented profiles does not seem to render significant (if any) improvement over the original SVM-based categorization technique presented in Section 3.2.

6. BLENDING TEXT AND LINK FEATURES

While in the previous section we evaluated profile-based classification in isolation, we now proceed to comparing it to, and using it in conjunction with, text-based classification.

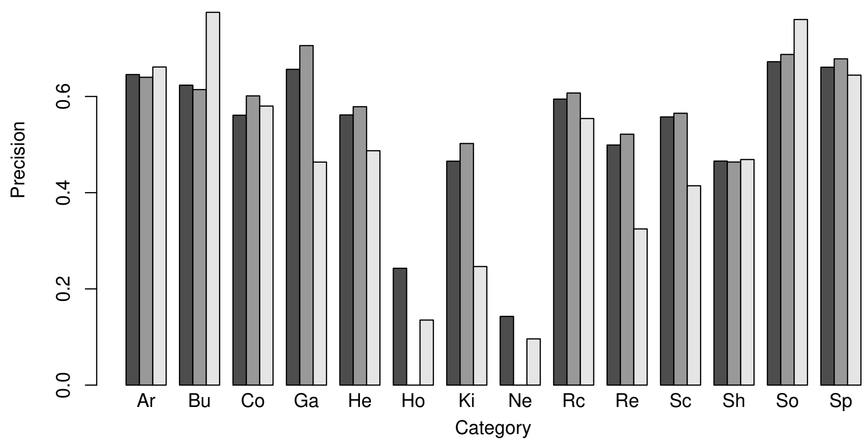

FIG. 8: Precision of M_2 (dark), M_1 (medium), and M_B (light) over $\mathcal{X}_{\text{eval}}$.

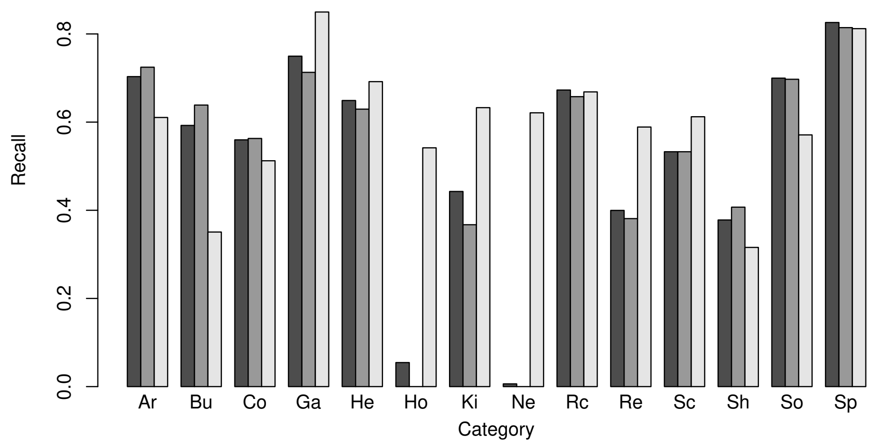

FIG. 9: Recall of M_2 (dark), M_1 (medium), and M_B (light) over $\mathcal{X}_{\text{eval}}$.

6.1. Text Extraction

In building a text-based classifier, we considered the following three possible sources of text:

- **Page descriptions**. Each URL listed in the Open Directory has one or more descriptions attached, where a description contains the *title* of the page and a short, one or two-sentence *summary* of the page topic, as provided by the person who added the URL to the directory. Page descriptions have two main advantages: (1) they are easy to extract and (2) being summaries, they tend to capture the essence of the page in a very compact form. In this latter respect they resemble anchor text, which has been very popular in web page classification in recent years (see, for instance, [9]).

- **Page contents**. To collect more textual data, one can download the web page at a given URL listed in the Open Directory. It is then possible to use all the text on the page or parts of it (such as the document title, the section titles, or the first and last sentences of paragraphs) to generate classification features. Search engines can readily access the textual content of pages kept in their indexes.

- **Host contents**. Using a crawler and possibly the web graph, one can download all accessible web pages of a host listed in the Open Directory, and then combine all the text on the pages. Again, search engines already have the pages in their indexes.

In our experiments, page descriptions yielded good enough classification performance.

We extracted 4,694,303 individual page descriptions from the Open Directory, which added up to over 81 million terms (1,783,218 unique terms). The number of descriptions per host varied significantly, between 1 and the maximum of 162,124 descriptions for pages on `www.cnn.com`. Without distinguishing between the terms in the titles and those in the summaries, we simply merged all the descriptions for a host x to form a corresponding bag of terms $\mathcal{B}_x$.

6.2. Classification Using Text Features

To evaluate text-based and link-based classification side by side, we built binary classifiers that detect hosts of a specific category. Let us start the discussion with the detection of *News* hosts.

From the set $\mathcal{X}$ we selected at random 2,500 *News* hosts and another 2,500 hosts of other categories. These 5,000 hosts constituted the set $\mathcal{T}_{\mathrm{Ne}}$ used for the training and evaluation of our classifiers.

The 5,000 hosts had 7,169 corresponding descriptions with 18,099 unique terms. In order to construct the text feature vectors of hosts, we relied on approaches adopted in earlier research, in particular in [6] and [9]. For a host x, our feature vector $\mathbf{s}_x$ was binary, each element indicating whether a particular term t is present in $\mathcal{B}_x$ or not. However, instead of having a vector element for each of the 18,099 terms, we restricted our attention to a small subset of 300 most discriminating terms. This practice is known as *feature selection* or *dimensionality reduction* in the machine learning and text categorization literatures. It is widely used both to remove non-informative terms (such as stop words) and to reduce training and prediction times.

In order to identify the most discriminating terms, we computed the average *mutual informa-*

tion [22] MI for each term t, defined as

$$\mathrm{MI}(t) = \sum_{i=0}^{1}\sum_{j=0}^{1} P(I_t = i \wedge I_k = j) \log_2 \frac{P(I_t = i \wedge I_k = j)}{P(I_t = i)P(I_k = j)} ,$$

where

- I_k is an indicator variable for the category, in our case *News* ($I_k = 1$) or non-*News* ($I_k = 0$);
- I_t is an indicator variable for the presence ($I_t = 1$) or absence ($I_t = 0$) of term t;
- $P(I_t = i)$ represents the probability of the presence/absence of term t in the term bags of hosts, in our case the fraction of hosts with descriptions containing/not-containing t;
- $P(I_k = j)$ represents the probability of category k, in our case $P(I_k = 0) = P(I_k = 1) = 0.5$; and
- the term $P(I_t = i \wedge I_k = j)$ represents a joint probability, for instance, $P(I_t = 1 \wedge I_k = 1)$ corresponds to the fraction of *News* hosts with descriptions containing t.

To illustrate, consider the term "a", which occurred in 834 descriptions of 606 individual hosts. Of the 606 hosts, 279 were *News* hosts and 327 belonged to some other category. Hence, in this case $P(I_t = 1) = 606/5000 = 0.12$, $P(I_t = 1 \wedge I_k = 0) = 327/5000 = 0.0654$, and $P(I_t = 1 \wedge I_k = 1) = 279/5000 = 0.0558$. The average mutual information for "a" is 0.0011, a value close to zero indicating that the term has little use in discriminating between the two categories.

The sorted list of the 10 most discriminating terms for the *News* binary classifier are shown in the first column of Table II.

Once we identified the 300 most discriminating terms, we constructed the text feature vectors $\mathbf{s}_x$ for all hosts $x \in \mathcal{T}_{\mathrm{Ne}}$ and trained several types of linear SVM classifiers:

- M_1, which uses the host profiles $\mathbf{r}_x$ (introduced in Section 5.2);
- M_2, which uses the augmented host profiles $\mathbf{r}'_x$ (introduced in Section 5.3);
- M_3, a purely text-based classifier, which uses the feature vectors $\mathbf{s}_x$;
- M_4, which uses both profiles and text-based features; and finally
- M_5, which uses augmented profiles and text-based features.

News	Business	Health	Sports
news	inc	health	club
newspaper	manufacturer	treatment	results
local	company	care	news
sports	ltd	medical	team
weekly	products	hospital	golf
daily	equipment	center	football
classifieds	links	information	horses
county	industrial	disease	league
community	industries	medicine	teams
obituaries	services	support	events

TABLE II: Most discriminating terms used in binary classification.

In all five cases we performed 5-fold *cross-validation*: We randomly partitioned the set $\mathcal{T}_{\mathrm{Ne}}$ into five equal subsets in advance, and constructed five instances of each of the five types of classifiers M_1 through M_5. For each instance, we used 4 subsets for training and the 5^{th} subset for evaluation.

Figure 10a presents the error rates of the various *News* classifiers in the form of a box-and-whisker plot. The five classifiers are listed along the horizontal axis; error rates are shown on the vertical. Note that the average error of M_1 and M_2 is around 16%, with M_2 performing slightly better (though displaying more variance). The text-based classifier M_3 outperforms the profile-based ones significantly, with an error rate around 10%. The combined classifiers M_4 and M_5 represent the best of both worlds, with error rates below 9%.

We performed similar experiments with binary classifiers for other categories as well. Next, we present the results for the classification of *Business*, *Health*, and *Sports* hosts. In all three cases we followed the same procedure established for the *News* classifiers: First, we created the samples $\mathcal{T}_{\mathrm{Bu}}$, $\mathcal{T}_{\mathrm{He}}$, and $\mathcal{T}_{\mathrm{Sp}}$, each containing 5,000 hosts. Next, we identified the 300 most discriminating terms for each set (the top 10 terms for each set are shown in columns 2, 3, and 4 of Table II) and built the classifiers M_1 through M_5.

Figure 10b presents the error rates for the classification of *Business* hosts. The gap between link-based and text-based classification is not as evident as for the *News* hosts, and error rates are higher in general.

Figure 10c, corresponding to the classification of *Health* hosts, illustrates the situation when link-based classification outperforms the text-based one. It seems that the two approaches are com-

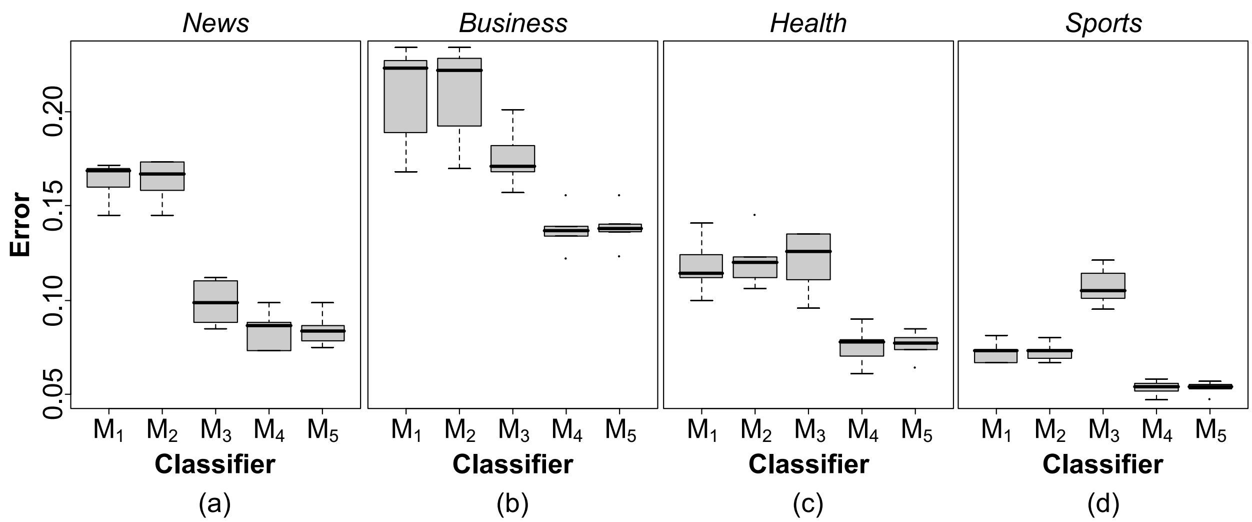

FIG. 10: *News*, *Business*, *Health*, and *Sports* classification error rates.

plementary, one being more appropriate in some settings and the other in others. Still, classifiers M_4 and M_5 that combine link and text features have the lowest error rates.

Finally, Figure 10d shows the error rates for the classification of *Sports* hosts. The performance difference between the link-based and the text-based classifiers is even even more dramatic than for the *Health* hosts.

In conclusion, category influence scores grouped in profiles seem to complement text features in host categorization. While, in general, link-based features alone are no better or worse than text-based features alone, combining the two yields classifiers with significantly improved performance.

7. ADDITIONAL RESULTS

In addition to our main results on the Yahoo! data set, we also have several findings derived from experiments using a second, smaller web graph of `.uk` hosts. Our data set is based on a web crawl performed at the Università degli Studi di Milano using Ubicrawler [2] in May 2006. The graph consists of 11,401 hosts and 730,744 links. As before, we used the category information available in Open Directory. For the purposes of our experiments, out of the 3,584 `.uk` hosts listed in the Open Directory under the same 14 top-level categories as before, we used the 2,389 non-ambiguous ones (listed under a single category) to generate seeds and training and evaluation sets for machine learning. Overall, we followed the same experimental procedure described for the Yahoo! data set in Section 5. However, the limited size of the `.uk` data enabled us to perform a large number of small-scale experiments. In the interest of space, we only summarize some of our findings in this paper.

- **Weighted influence.** For both regular and reverse influence, we experimented with PageRank computations on an unweighted host-level web graph as well as a host-level graph with edges weighted by the number of page-level links between hosts. Unweighted PageRank yielded better binary classification results for some categories and worse for some others. We found no conclusive evidence that weighted graphs would be more useful than unweighted ones, or vice versa.

- **Reverse influence.** In addition to the classifiers based on regular node profiles, we also built ones for profiles consisting of reverse influence scores. As with text features, regular profiles seem to perform better for some categories and reverse profiles for the others. At the same time, the combination of the two consistently yields superior classification results. For instance, binary classifiers for the category *Computers* had an average error rate of 27% for regular profiles and 34% for reverse profiles. However, for profiles combining both regular and reverse influence, the average error rate was as low as 9.7%. Our results indicates that regular and reverse influence are complementary pieces of information.

- **Limited-distance influence.** Our most interesting findings concern limited-distance influence. For most binary classifiers, using distance-k influence seems to lead to better classification results than using influence computed over all paths. The box-and-whisker plot in Figure 11 illustrates the point for the binary classification of *Computers* hosts. The horizontal axis indicates the number of PageRank iterations performed (*Conv* means convergence, reached in 102 iterations), while the vertical axis represents the classification error rate, computed through 5-fold cross-validation. Running PageRank for $k = 3$ iterations only produces the best classification results with an average error rate around 25%. The error rates increase with the number of iterations, leveling off around 30% for 9 and more iterations. We noticed a similar phenomenon for other categories as well, for both regular and reverse influence, though the optimal choice of k varied between 1 and 9. Our findings are in line with the common belief that score propagation on the web graph reaches a diffusion point after a few steps, due to central hub nodes that distribute their incoming score indiscriminately through a large number of outlinks.

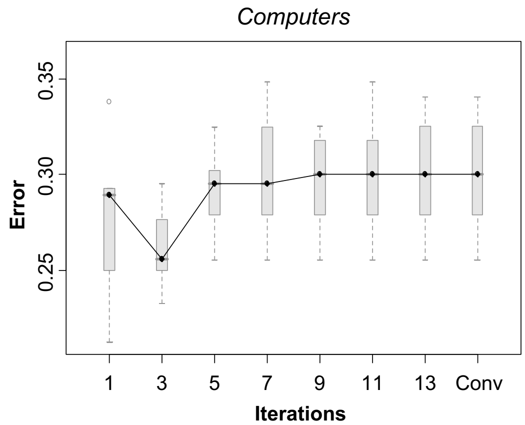

FIG. 11: Classification error rates.

8. RELATED WORK

This paper rests on the foundations provided by two orthogonal research directions:

(Hyper)text categorization. Document classification has an extensive literature dating back to the '60s and '70s. In the early years of the World Wide Web naïve Bayes classifiers represented the preferred approach to hypertext categorization, and achieved considerable success as reported, for instance, in [5]. Over the last decade, the information retrieval community has also turned to emerging robust generic classification techniques, such as boosted [7, 8] decision trees [3] and support vector machines [14, 15, 21].

While the choice of the machine learning technique can have a significant impact on performance, the prediction models can only be as good as the data that they use. Researchers have built effective classifiers relying not only on the *local* terms in documents, but on (some of the) terms in the linked documents as well [1, 5, 9].

The most direct use of the link structure in previous work was proposed in [5]. The authors present an iterative relaxation labeling scheme to gradually refine the predicted category of a hyperlinked document and its neighbors based on local text features and the categories predicted in the previous iteration. Relaxation labeling allows the authors to successfully classify documents even if the category information of most neighbors is missing. However, the classification model employed still requires a training (sub)graph that has category labels for *all* its nodes connected by links. Such a training graph is not available for a non-trivial part of the Web and creating one by manual labeling would represent a huge undertaking. Without adequate web training data, we are unable to perform a direct comparison between the relaxation labeling scheme and our approach. Also, note that in order to predict the category of any one document, relaxation labeling repeatedly classifies both the document and all its neighbors. Accordingly, the average number of classification steps per document can easily be in the order of hundreds, making the scheme computationally expensive for the web. In this context, our paper proposes an alternative, efficient approach to learning, which is based on a more relaxed notion of connectedness and a more limited knowledge of the category of some documents.

PageRank. The PageRank algorithm was proposed in [18] for query-independent link-based ranking of web documents. Topic-sensitive PageRank [12] allows for topical bias in ranking, essentially by computing and combining category influence scores, where the categories are possible topics. Our paper makes the additional step of using influence scores for categorization purposes instead of ranking.

9. CONCLUSIONS

We proposed an approach to the automatic categorization of web documents based on link structure information and the *a priori* knowledge of the category of a (small) group of documents. Link-based categorization diverges considerably from the traditional, well-studied research direction of classifying documents based on their textual contents.

Our approach is that of *feature engineering*: We do not introduce a new machine learning model that directly takes the web graph as input. Instead, we compute a set of category influence vectors and produce compact profiles that can act as input features of efficient generic machine learning algorithms, such as support vector machines. The individual algorithmic building blocks that we use scale well to large data sets.

Our experiments indicate that link-based categorization yields performance figures comparable to those of text-based categorization, i.e., an error rate of around 10-15% for binary topic classifiers. Furthermore, combining link and text features results in superior performance, in particular, error rates around 5%.

The use of features derived from the link structure between documents may enable automatic categorization even in settings when text features are unavailable. For instance, consider a photo collection in which individual pictures are linked through various metadata, such as matching creation time or location, or camera serial number, but do not have text descriptions attached. As soon as a human editor manually categorizes some subset of the photos, our solution enables the automatic categorization of the rest of the collection. Similarly, link-based features might improve the categorization accuracy for web documents without local textual information, for instance, sites built using Macromedia Flash that search engines cannot index.

Node profiles can also be valuable in situations when either the text features are unreliable (for instance, when the content of a web page and/or the anchor text are heavily spammed [10]) or documents belonging to different categories use the same terminology (for instance, the text-based separation of pro-life and pro-abortion web pages is often challenging as they tend to use identical language).

---

[1] Avrim Blum and Tom Mitchell. Combining labeled and unlabeled data with co-training. In *Proceedings of the* 11^{th} *Conference on Computational Learning Theory (COLT)*, 1998.

[2] Paolo Boldi, Bruno Codenotti, Massimo Santini, and Sebastiano Vigna. Ubicrawler: A scalable fully distributed web crawler. *Software Practice and Experience*, 5(3), 2002.

[3] Leo Breiman, Jerome Friedman, Richard Olshen, and Charles Stone. *Classification and Regression Trees*. Wadsworth, 1984.

[4] Soumen Chakrabarti. *Mining the Web*. Morgan Kaufmann, 2002.

[5] Soumen Chakrabarti, Byron Dom, and Piotr Indyk. Enhanced hypertext categorization using hyperlinks. In *Proceedings of the ACM SIGMOD Conference*, 1998.

[6] Susan Dumais, John Platt, David Heckerman, and Mehran Sahami. Inductive learning algorithms and representations for text categorization. In *Proceedings of the Conference on Information and Knowledge Management (CIKM)*, 1998.

[7] Yoav Freund and Robert Schapire. Experiments with a new boosting algorithm. In *Proceedings of the* 13^{th} *International Conference on Machine Learning (ICML)*, 1996.

[8] Jerome Friedman. Greedy function approximation: A gradient boosting machine. *The Annals of Statistics*, 29(5), 2001.

[9] Eric Glover, Kostas Tsioutsiouliklis, Steve Lawrence, David Pennock, and Gary Flake. Using web structure for classifying and describing web pages. In *Proceedings of the* 11^{th} *International Conference on World Wide Web*, 2002.

[10] Zoltán Gyöngyi and Hector Garcia-Molina. Web spam taxonomy. In *Proceedings of the First International Workshop on Adversarial Information Retrieval (AIRWeb)*, 2005.

[11] Zoltán Gyöngyi, Pavel Berkhin, Hector Garcia-Molina, and Jan Pedersen. Link spam detection based on mass estimation. In *Proceedings of the* 32^{nd} *International Conference on Very Large Data Bases (VLDB)*, 2006.

[12] Taher Haveliwala. Topic-sensitive PageRank. In *Proceedings of the* 11^{th} *International Conference on World Wide Web*, 2002.

[13] Glen Jeh and Jennifer Widom. Scaling personalized web search. In *Proceedings of the* 12^{th} *International Conference on World Wide Web*, 2003.

[14] Thorsten Joachims. Text categorization with support vector machines: Learning with many relevant features. In *Proceedings of the* 10^{th} *European Conference on Machine Learning (ECML)*, 1998.

[15] Thorsten Joachims. A statistical learning model of text classification for support vector machines. In *Proceedings of the ACM SIGIR Conference*, 2001.

[16] Tom Mitchell. *Machine Learning.* McGraw-Hill, 1997.

[17] Lan Nie, Brian Davison, and Xiaoguang Qi. Topical link analysis for web search. In *Proc. of SIGIR*, 2006.

[18] Lawrence Page, Sergey Brin, Rajeev Motwani, and Terry Winograd. The PageRank citation ranking: Bringing order to the web. Technical report, Stanford University, 1998.

[19] Feng Qiu and Junghoo Cho. Automatic identification of user interest for personalized search. In *Proceedings of the* 15^{th} *International Conference on World Wide Web*, 2006.

[20] Pang-Ning Tan, Michael Steinbach, and Vipin Kumar. *Introduction to Data Mining.* Pearson/Addison-Wesley, 2006.

[21] Vladimir Vapnik. *The Nature of Statistical Learning Theory.* Springer, 1995.

[22] Yiming Yang and Jan Pedersen. A comparative study of feature selection in text categorization. In *Proceedings of the* 14^{th} *International Conference on Machine Learning (ICML)*, 1997.