Web Site Mining : A new way to spot Competitors, Customers and Suppliers in the World Wide Web

Martin Ester
Simon Fraser University
School of Computing Science
Burnaby BC, Canada V5A 1S6
++1-604-291 4411
ester@cs.sfu.ca

Hans-Peter Kriegel
Institute for Computer Science
University of Munich (LMU)
Oettingenstr. 67, D-80538 Munich, Germany
++49 89 2180 9191
kriegel@dbs.informatik.uni-muenchen.de

Matthias Schubert
Institute for Computer Science
University of Munich (LMU)
Oettingenstr. 67, D-80538 Munich, Germany
++49 89 2180 9321
schubert@dbs.informatik.uni-muenchen.de

ABSTRACT

When automatically extracting information from the world wide web, most established methods focus on spotting single HTML-documents. However, the problem of spotting complete web sites is not handled adequately yet, in spite of its importance for various applications. Therefore, this paper discusses the classification of complete web sites. First, we point out the main differences to page classification by discussing a very intuitive approach and its weaknesses. This approach treats a web site as one large HTML-document and applies the well-known methods for page classification. Next, we show how accuracy can be improved by employing a preprocessing step which assigns an occurring web page to its most likely topic. The determined topics now represent the information the web site contains and can be used to classify it more accurately. We accomplish this by following two directions. First, we apply well established classification algorithms to a feature space of occurring topics. The second direction treats a site as a tree of occurring topics and uses a Markov tree model for further classification. To improve the efficiency of this approach, we additionally introduce a powerful pruning method reducing the number of considered web pages. Our experiments show the superiority of the Markov tree approach regarding classification accuracy. In particular, we demonstrate that the use of our pruning method not only reduces the processing time, but also improves the classification accuracy.

General Terms

Algorithms, Performance.

Keywords

web content mining, web site mining, web site classification, Markov classifiers.

1. INTRODUCTION

In recent years the world wide web (www) has turned into one of the most important distribution channels for private, scientific and business information. One reason for this development is the relatively low cost of publishing a web site. Compared to other ways like brochures or advertisements in newspapers and magazines, the web offers a cheaper and more up to date view on a business for millions of users. So even very small companies are enabled to present their products and services in the www. Furthermore, many companies like online-shops operate via the internet, so presenting themselves there comes naturally. But with dramatically increasing numbers of sites, the problem of finding the ones of interest for a given problem gets more and more difficult.

The problem of spotting new web sites of special interest to a user, is not yet handled adequately. Though directory services like Yahoo[11] and DMOZ[4] can offer useful information, the entries there are often incomplete and out of date due to manual maintenance. Furthermore, the directory services do not support all the categories relevant to some specific user. Since companies need to know who their potential competitors, suppliers and customers are, web-crawlers and spiders have to be enhanced to solve problems where single web pages are not the main focus. For example in the IT-business, where products and services can change quickly, a system that spots special kinds of web sites and offers the opportunity to search them, will turn out to be very useful. Other reasons for focusing on whole sites instead of single pages are: There are much less sites than single pages on the internet so that the search space can be dramatically reduced. The mining for whole web sites can offer a filter step when searching for detailed information. For example when looking for flight prices you might try to spot travel agencies first. One final reason is the higher stability of sites. Sites appear, change and disappear less often then single pages, which might be updated daily. Of course a part of the site is modified too, but in most cases this will not change the class of a site.

The classification of text (and web) documents has received a lot of attention in the research community fore many years. Methods such as Naive Bayes [7] [12] and support vector machines [5]

[12] have been successfully applied to text and hypertext documents. Furthermore, a few special methods have been developed to exploit the hyperlinks to improve the classification accuracy, e.g. [1] and [2]. However, the classification of complete web sites (instead of single web pages) has not yet been investigated. Given a set of site classes C and a new web site S consisting of a set of pages P, the task of web site classification is to determine the element of C which best categorizes the site S.

In this paper, we introduce several approaches of classifying web sites based on different representations:

- Classification of superpages (a web site is represented as a single virtual web page consisting of the union of all its pages, i.e. the web site is represented by a vector of term frequencies).
- Classification of topic vectors (a web site is represented by a vector of topic frequencies).
- Classification of web site trees (a web site is represented by a tree of pages with topics).

The efficiency of web site classification crucially depends on the number of web pages downloaded, since the download of a remote web page is orders of magnitudes more expensive than in-memory operations. Therefore, we also propose a method of pruning a web site, i.e. downloading only a small part of a web site but still achieving high classification accuracy.

The rest of this paper is organized as follows. Section 2 contains a summary of related work. In section 3 we discuss the problem of extracting a web site and introduce a simple approach of web site classification. The fourth section presents several advanced methods of representing and classifying web sites. Section 5 discusses the pruning of web sites. In Section 6, we report the results of an experimental evaluation of the proposed methods. Section 7 summarizes the results and outlines some directions for future research.

2. RELATED WORK

In this section, we briefly review related work on text classification, in particular classification of web pages, and on the classification of sequential data which is related to the classification of web site trees and paths within such trees.

Text classification has been an active area of research for many years. All methods of text classification require several steps of preprocessing of the data. First, any non-textual information such as HTML-tags and punctuation is removed from the documents. Then, stopwords such as “I”, “am”, “and” etc. are also removed. Typically, the terms are reduced to their basic stem applying a stemming algorithm. Most text classification algorithms rely on the so-called *vector-space model*. In this model, each text document is represented by a vector of frequencies of the remaining terms. The term frequencies may be weighted by the inverse document frequency, i.e. a term occurring in fewer documents gets a larger weight. Finally, the document vectors are normalized to unit length to allow comparison of documents of different lengths. The vector-space has a very high dimensionality since even after preprocessing there are typically still several thousands of terms. Due to the high dimensionality, most frequencies are zero for any single document and many of the standard classification methods perform poorly. However, methods that do not suffer so much from high dimensionalities have been very successful in text classification, such as Naive Bayes [7] [12] and support vector machines [5] [12]. While most of the above methods have been applied to pure text documents, an increasing number of publications especially deal with the classification of web pages. Several authors have proposed methods to exploit the hyperlinks to improve the classification accuracy, e.g. [1] and [2]. [2] introduces several methods of relational learning considering the existence of links to web pages of specific classes. [1] presents techniques for using the class labels and the text of neighboring (i.e., linked) web pages. However, all these methods aim at classifying single web pages, not complete web sites.

In our most sophisticated approach to web site classification, we will represent a web site as a so-called web site tree and our classifier will be based on the paths within these trees. Therefore, we briefly survey methods for classification of sequence data. [3] discusses and evaluates several methods for the classification of biological sequence data, e.g. the *k*-nearest neighbor classifier, Markov classifiers and support vector machines. Whereas biological sequences tend to be very long, paths in a web site tree are typically relatively short. Furthermore, in biological sequence classification the data are given and labeled apriori, whereas in web site mining loading and labeling the data is an expensive procedure. Classification algorithms are difficult to apply to sequential data because of the extremely large number of potentially useful features. [6] proposes a sequence mining technique to act as a preprocessor to select features for standard classification algorithms such as Naive Bayes. Several techniques have been introduced in the literature for efficiently determining the frequent sequences within some database, such as [13]. However, these techniques only find the frequent patterns but do not build a classifier.

3. EXTRACTION OF WEB SITES AND A SIMPLE APPROACH OF CLASSIFICATION

The mining of complete web sites is in many aspects different from mining single web pages. Sites may vary strongly in size, structure and techniques used to build them. Another aspect is language. Many professional sites especially in the non-English-speaking areas are at least bilingual to provide international usability. Most page classification projects use only text documents in a single language, which may prove insufficient, when trying to handle whole sites. A further problem occurring on sites is to determine the borders of a site. When classifying single pages it is easy to identify one page as one object of interest. Though there are several interesting projects [1] [2] to soften this natural border, the common approach is based upon page classification. The borders of a web site on the other hand have to be specified first when trying to classify it. Finally, a web site is not just a set of terms or a set of single web pages. To represent the structure within a web site, we use the general concept of graphs which will later be specialized to trees.

The *web site* of a domain *D* is a directed graph $G_D(N,E)$. A Node $n \in N$ represents an HTML-page whose URL starts with *D*. A link between n_1 and n_2 with $n_1, n_2 \in N$ is represented by the directed edge $(n_1, n_2) \in E$.

Thus, every HTML-document under the same domain name is a node in the site graph of the domain and the hyperlinks from and to other pages within the same domain are the connecting edges. This rather simple definition holds for our running application of spotting small and medium-size business sites. Most companies rent their own domain, so the possibility that the actual site starts below the domain is rather low. More complex appearances in the web spreading over several domain names are mostly found in rather large companies, which are already known to most people and which we do not consider in this paper.

To download a site from the web, the following algorithm can be applied. Begin with the page whose URL consists of the domain name only. This is the only page that can be derived from the domain name directly. We call it therefore the *starting page*. After reading it, we can use an HTML-parser to determine the links to the other pages within a site. Note that considering *FRAME*- and *EMBED*-tags as links is necessary here to get an as complete picture of a site as possible. After link extraction, every link to a page, beginning with the same domain name, is followed. It is necessary to mark the pages already visited to avoid circles. Hence, every reachable page is visited and every link found is followed until the picture of the reachable parts of the site is completed.

The most common way to classify single HTML-documents is using the Naive Bayes classifier [7] [12] or support vector machines [5] [12] on a feature space of terms. Here the quality of the results is highly dependent on the right choice of terms. To improve it, several very useful methods have been introduced. The elimination of stop words and stemming (reduction of words to their basic stems) are common techniques to determine the right terms and to reduce the number of terms to be considered. Another interesting possibility is to expand the feature space to include structural components. The number of words and images, the occurrence of forms or frames or the number of links from a page can offer vital information depending on the specified classes.

We define the task of web site classification as follows. Given a set of site classes C and a new web site S consisting of a set (or any other data structure such as a graph) of pages P, the task of *web site classification* is to determine the element of C which best categorizes the site S.

The simplest way of classifying a web site is to extend the methods used for page classification to our definition of web sites. We just generate a single feature vector counting the frequency of terms over all HTML-pages of the whole site, i.e. we represent a web site as a single "superpage". Therefore, we call this simple approach *classification of superpages*. The advantage of this approach is that it is not much more complex than the classification of single pages. You just have to walk through the nodes of the site graph and count terms. Afterwards the vector can be classified by any standard data mining package, e.g. the weka-package[10] we used in our experiments. However, the superpage classifier has several conceptual drawbacks. The right choice of the key terms proves to be very delicate for this approach. As mentioned before, sites can contain documents in several languages. Structural features like the occurrence of frame tags loose most of their significance. Another very important problem here is the loss of local context. Keywords appearing anywhere within the site are aggregated to build up a bag-of-words view of the whole web site graph. As shown in section 6, this simple classifier performed poorly in most experiments.

4. ADVANCED CLASSIFICATION OF WEB SITES

The main reason why the superpage approach does not perform adequately, is the fact that it makes no difference between an appearance within the same sentence, the same page or the same site. However the context plays an important role considering that sites can spread over up to several thousand single HTML-documents, containing information about various topics. For example, the meaning of the terms *'network administration'* and *'services'* on the same page implies that this company offers network administration as one of its services. But without the constraint that both terms appear on the same page the implication gets much weaker. Any company, offering any service and looking for a network administrator, will provide those terms too.

To overcome these problems, we need more natural and more expressive representations of web sites. In this section, we introduce such representations. Then, we present two advanced methods of web site classification based on these representations. In this paper, we use the discovery of potential customers, competitors or suppliers as our running application. However, we would like to argue that all the proposed methods for web site classification are not restricted to corporate websites and have a much broader range of applications.

4.1 Representation of Web sites

Compared to the superpage approach, we change the focus of site classification from single key terms to complete HTML-documents. In order to summarize the content of a single web page, we assign a topic out of a predefined set of topics (page classes) to it. Since the terms only influence the topic of the page they appear on, the local context is preserved. Actually, the preprocessing step can use all mentioned techniques for the selection of terms introduced for page classification without any restriction. To conclude, we introduce *page classes* besides the web site classes.

The choice of topics (page classes) for our experimental evaluation was based upon the observations that we made during the examination of many business sites in several trades. Although their trades varied widely, the following categories of pages are to be found in most classes of business-sites: *company, company philosophy, online contact, places and opening hours, products and services, references and partners, employees, directory, vacancies and other*. The *"other"*-category stands for any topic not specified more precisely. Note that these page

classes are discussed here only for the purpose of illustration, but our method is generally applicable for any web site classes as well as any web page classes. Since the features representing the topic of a page will vary from trade to trade, every category except *"other"* has to be specialized for each trade we want to spot. For example, we distinguished between the products and services of a florist and an IT-service provider (our examples in section 6).

To determine the topic of a page we again used text-classification on terms using the Naive Bayes classifier. Since there is always a classification error for each page, the probability that the complete graph is labelled correctly is rather low. But the average number of correctly labelled nodes is about the mean classification accuracy of the page classification, as can be shown when treating the problem as a Bernoulli chain. We will soon discuss the impact of this effect on our main classification problem.
Based on the labelled pages of a web site, we propose two different representations of a web site:

- *Feature vector of topic frequencies.*

  Each considered topic defines a dimension of the feature space. For each topic, the feature values represent the number of pages within the site having that particular topic. This representation does not exploit the link structure of the site, but it considers a web site as a *set* of labelled web pages.

- *Web site tree.*

  To capture the essence of the link structure within a site, we represent it as a labelled tree. The idea here is, that the structure of most sites is more hierarchic than network-like. Sites begin with a unique root node provided by the starting page and commonly have directory-pages that offer an overview of the topics and the links leading to them. Furthermore, the information in most sites begins very general in the area around the starting page and is getting more and more specific with increasing distance. For example, we observed that pages regarding the company itself are more often to be found within only a few links from the starting page than ones about specific product features.

For building web site trees, we use the minimum number of links as a measure of distance between two pages of a site and build up the tree as the set of minimum paths from the starting page to every page in the graph. Therefore, we perform a breadth-first search through the web site graph and ignore the links to pages already visited. Note, that in the case of two paths to the same page of equal length, the one occurring first is chosen. Though there is no way to tell which path preserves more information, this definition was made to make tree derivation deterministic. The trees in figure 1 are generated by this method.

4.2 Classification of Topic Frequency Vectors

After the transformation of web sites into topic frequency vectors, most general classifiers such as Bayes classifiers and decision tree classifiers are applicable. Especially tree classifiers like C4.5 [9] showed an enormous improvement of classification accuracy compared to the simple superpage approach.

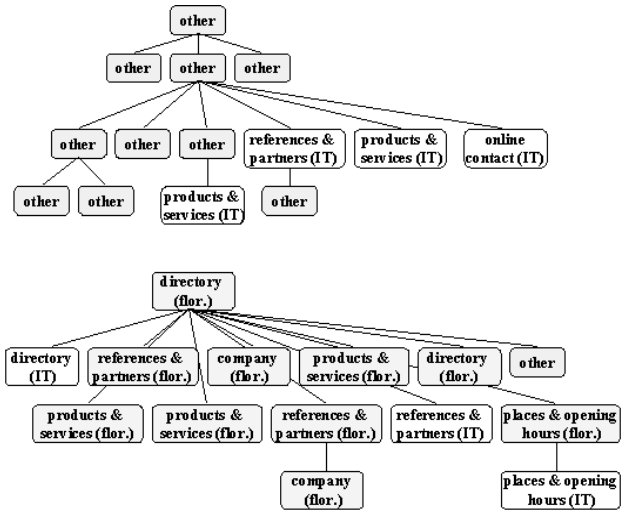


Figure 1: Examples of web site trees. A typical small IT-service provider (above) and a typical small florist site (below)

Note that the dimensionality of the *topic* frequency vectors is much smaller than the dimensionality of the *term* frequency vectors which are used in the superpage approach.

4.3 Classification of Web site trees

In this section, we present an improved method of web site classification based on the web site tree representation, i.e. exploiting the links within a site. Our method is based upon the idea of Markov chains in combination with the Naive Bayes classifier. Therefore, we first follow the Naive Bayes approach and define the most likely (*web site*) class as:

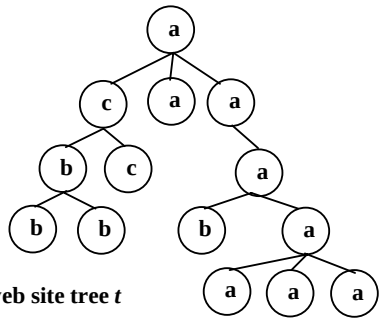
$$(1). \qquad C = maxarg\ p(C_i|S) = maxarg(\ p(C_i)\ p(\ S|C_i\ ))$$

Here the predicted class C of the site S is the class C_i that explains the occurrence of the given site S best. Due to the Bayesian rule the probability $p(C_i|S)$ is the product of the apriori probability $p(C_i)$ and the probability $p(S|C_i)$ that given the model for class C_i the web site tree S would have been constructed. We therefore estimate $p(C_i)$ as the relative frequency of web sites in the class C_i. The approximation of $p(S|C_i)$ depends on the chosen model.

The concept of k-order Markov chains is applied to web site trees using the following procedure. Beginning with the probability for the label of our root node we multiply the probabilities of the transition between the k last nodes and their successor. Note that these *transition probabilities* only use the static link structure of the web pages, they do not use any dynamic click-through probabilities. In the simple case of 1-order Markov chains, for example, the transition probability for the page classes X and Y with respect to site class S is the probability that within a web site of class S a link from a page of class X leads to a page of class Y. Since there can be more than one successor in the tree, we multiply the transition probabilities for every child node,

Transition probabilities for class I:

→	a	b	c
a	**0.2**	**0.7**	**0.1**
b	**0.2**	**0.2**	**0.6**
c	**0.1**	**0.5**	**0.4**
none	**0.1**	**0.6**	**0.3**



web site tree *t*

Transition probabilities for class J:

→	a	b	c
a	**0.2**	**0.5**	**0.3**
b	**0.2**	**0.4**	**0.4**
c	**0.1**	**0.1**	**0.8**
none	**0.1**	**0.1**	**0.8**

```
p(t|I) = 0.1 * 0.1 * 0.5 * 0.2 *       (a-c-b-b)
                           0.2 *       (_-_-_-b)
                     0.4 *             (_-_-c)
               0.2 *                   (_-a)
               0.2 * 0.2 * 0.7 *       (_-a-a-b)
                           0.2 * 0.2 * (_-_-_-a-a)
                               * 0.2 * (_-_-_-_-a)
                                 0.2   (_-_-_-_-a)

p(t|J) = 0.1 * 0.2 * 0.1 * 0.4 *       (a-c-b-b)
                           0.4 *       (_-_-_-b)
                     0.8 *             (_-_-c)
               0.4 *                   (_-a)
               0.4 * 0.4 * 0.4 *       (_-a-a-b)
                           0.4 * 0.4 * (_-_-_-a-a)
                               * 0.4 * (_-_-_-_-a)
                                 0.4   (_-_-_-_-a)
```

Figure 2: The calculation of the model probability *p(t|C)* for 2 site classes (I and J) and 3 page topics (a,b,c) using a 1-order Markov tree.

traversing along every path that a tree contains simultaneously. Note that the probability that a node is reached is only used once and the direction of the process is given by the direction the tree is traversed. This Markov tree model is very similar to the concept of branching Markov chains [8], but does not take branching probabilities into account.

The model for the probability of the site S generated by our model C_i is the following: Let L be the set of page topics extended by the element *"none"*. The *"none"*-topic acts as a placeholder for paths shorter than k and is used to simplify the calculation. Furthermore, let l_t be the label of a node t and let C_i be the i-th site class. The function *pre* (with $pre(k,t) = l$) returns the label of the k-th predecessor of the node t with respect to the web site containing t. If there is no such predecessor, it returns *"none"*. Note that the predecessor is uniquely defined because we have a tree structure. Then the conditional probability of the web site tree S can be calculated as:

$$(2).\ p(S \mid C_i) = \prod_{t \in S} p(l_t \mid pre(k-1,t),...,\ pre(1,t))$$

Thus, for every node t in the site tree S the probability that its label l_t occurs after the occurrence of the labels of its k predecessors is multiplied. Figure 2 visualizes the calculation of $p(S|C_i)$ for two site classes.

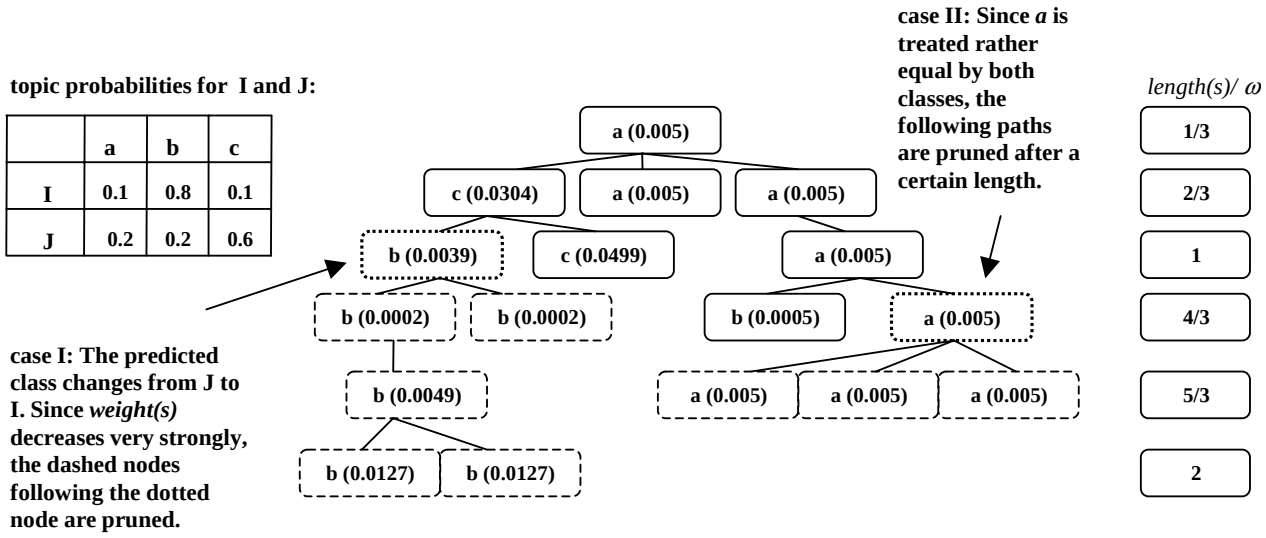
This method does not use the possible correlations between siblings and thus, the context taken into account is limited to the path from the starting page. To estimate the transition probabilities, we calculate the mean distribution of the occurring transition of length k. Note that this is a difference to the relative occurrence of the transitions in class C_i. Due to the classification error in the preprocessing step (when assigning topics to the web pages), the probability for "phantom transitions" that are generated accidentally is rather high. Especially in site classes where the average number of pages is rather high, the absolute number of certain transitions can accidentally match the number in site classes trained on sites having fewer pages. Thus, a problem appears when the matched transition is highly specific for the class consisting of the smaller sites. In this case, the significance gets distorted. To smoothen this effect, the mean distribution uses the size of a site as a measure for the number of transition occurrences and normalizes the number of transition occurrences within a site to one. Therefore, an appearance in a site class, where the total number of transitions is higher, will be given less importance. Note that the information about the number of pages within a site is not taken into account. The use of the mean distribution proves to be superior in every approach based upon the preprocessing step in all experiments.

For the choice of the degree k of the Markov chains we tried out the values zero, one and two. According to our results and a consideration discussed in section 6, a higher degree was not reasonable. For every choice of k this model yields comparably good results to the standard classifiers applied to the distribution vectors. For k=0 it even outperformed the other three approaches.

Since the concept of the 0-order Markov tree shows similarities to the Naive Bayes classifier applied to topic frequency vectors, we examined their differences more closely. For a better understanding of the following comparison, we will present the calculation of the single probabilities in a 0-order Markov tree.

$$(3).\ p(S \mid C_i) = \prod_{n \in S} p_{topic(n)}^{C_i} = (p_1^{C_i})^{r_1} \cdot ... \cdot (p_k^{C_i})^{r_k}$$

given $\sum_{j \in L} r_j = |S|$



web site tree *t* (The nodes are labeled with the topic and the value of *weight(s)*)

Figure 3: The effects of the pruning method on the *0*-order Markov tree classifier with ω =3. The dashed nodes are to be pruned.

Now, let S be the site to be classified, C_i a site class and let $p_{topic(n)}^{C_i}$ be the probability of the occurrence of the topic of the page n in class C_i. Furthermore, let r_j be the number of occurrences of the topic $j \in L$ in the site S and let L be the set of topics. Thus, the probability is calculated by taking the r_j-th power of every topic probability and then multiplying those factors for every topic. This is equivalent to a multinomial process except for the difference that the multinomial coefficient can be neglected due to its equal occurrence in every class C_i.

To explain the different results compared to Naive Bayes, the following differences can be pointed out. Naive Bayes considers the occurrence of a topic to be independent from the occurrences of the other topics. But since Naive Bayes calculates the probability of the total number of occurrences within the site, the probability of just one occurrence is not independent from other occurrences of the same topic. Depending on the used distribution model for a dimension, a further occurrence of a topic that is very typical for a certain site class, will even decrease the probability for that class, if the number of occurences differs strongly from the estimated mean value. The 0-order Markov tree on the other hand always increases the conditional probability $p(S|C_i)$, if a further topic specific to the site class C_i occurs in the site S. A further important difference is the consideration of the number of total pages of a site. Since a large number of occurrences will automatically decrease the number of occurrences in all other topics, the 0-order Markov tree uses knowledge that Naive Bayes does not.

A further interesting property of 0-order Markov trees is the possibility to calculate the probabilities incrementally. For every disjunctive segmentation $(s_1,..,s_2)$ of our site S, the following equation holds:

$$(4). \prod_{s_j \in S} p(s_j \mid C_i) = p(S \mid C_i)$$

In other words, if the probability $p(s_j|C_i)$ for the class C_i *is* higher than for any other class, the subset s_j will increase the probability that $p(S|C_i)$ is also higher than for any other class. This property will play an important role in our pruning method.

5. PRUNING THE IRRELEVANT AREA OF A WEB SITE

The efficiency of web site classification crucially depends on the number of web pages downloaded, since the download of a remote web page is orders of magnitude more expensive than in-memory operations. Therefore, we introduce a classification method, downloading only a small part of a web site, which still achieves high classification accuracy. This method performs some pruning of a web site and applies the classifier only to the remaining web pages. For the purpose of pruning, it is important to spot a limited area that carries the information necessary for accurate classification. The existence of such an area is very likely due to the hierarchical design of most sites. The challenge is to detect the exact border of this area.

For the following reasons, the naive approach of reading the first n pages of a web site does not yield a good accuracy. First, the topology of a web site is a matter of individual design and therefore tends to be very heterogeneous. Many sites contain large amounts of pages providing just structure but not content. For example, animated introductions or frames are instruments of making a site more usable or attractive, but in most cases they do not contain any content recognized by a page classifier. Another

important aspect is how much content is provided on a single page. One could spread the same amount of information over several pages or just use one large page containing all the information. Consequently, the total number of pages already read is not a good indicator for pruning a web site.

The starting page is always the first page to be read and the further ones are linked to it. So the question is, how to traverse the web site and where to stop following the links any further. Breadth first traversal (used already to build the web site trees) seems to be a promising approach. Since this traversal strategy orders the pages with respect to their distance to the starting page, the more general and therefore more important pages are generally visited first. Thus, the traversal offers a reasonable guidance for the classification process. The question therefore becomes, where to prune the subtrees. Note that the site tree is derived from the site graph during classification and that its topology depends on the pruning criteria. Thus, a node can only be ignored, when every path leading to it is pruned. Therefore, the trees derived by the breadth-first traversal in combination with pruning can vary from those derived by a pure breadth-first traversal.

Our pruning method is based on the following two propositions about the paths from the starting page to the following pages:

- Case I: The membership of a complete path in some site class is strongly depending on the topics of the pages closest to the starting page. As mentioned before, general information about the class of a web site is most likely placed within a few links from the starting page. If a new topic follows, it appears in the context of the former topic.
- Case II: There are cases where a whole subtree and the path leading to it does not show any clear class membership at all. Though it is obvious to a human that its impact on classification is rather low, recognizing this kind of subtree is a difficult problem. A tree could always become highly specific after the next node. But after a reasonable length of the path, the probability that the meaning of the subtree is of general nature tends to be rather low. So the strength of the class information has to be measured by the length of a path it appears on.

To exploit these propositions, it is necessary to measure the degree of class membership for a path and its impact on site classification. Here the ability of the 0-order Markov tree to calculate the class membership incrementally is very useful. Since the methods based on the established classifiers cannot predict the impact on the site class without knowing about the rest of the topic distribution, they are not suitable in the context of pruning. The 0-order Markov tree on the other hand can calculate the probability for the occurrence of a path for each class though it was trained on complete sites.

The conditional probabilities $p(s \mid C_i)$ yield the information about the degree that a path *s* supports a site class C_i. Since the focus lies on the importance of a path for site classification, the actual class or classes it supports are not relevant here. To quantify the importance for the complete site, we use the variance of the conditional probabilities over the set of all web site classes. Since the variance is a measure for the heterogeneity of the given values, it mirrors the ability of a path to distinguish between the different site classes. A high variance of the probabilities of the web site classes indicates a high distinctive power of that particular path. Let *s* be any path in the site tree *S* and let $p(s|C_i)$ be the probability that *s* will be generated by the model for class C_i, then

$$weight(s) = \operatorname{var}_{C_i \in C}\left(p(s \mid C_i)^{\frac{1}{length(s)}} \right)$$

is a measure for the importance of the path s for site classification. Let *length(s)* be the number of nodes in path *s*. The (*1/length(s)*)*th*-power is taken to normalize *weight*(s) with respect to *length*(s). This is necessary for comparing weights of paths of varying length.

To determine *weight(s)* according to the above propositions (cases I and II), we have to show that we can recognize a change in the class membership and recognize the occurrence of unimportant paths. The last requirement is obvious since a low variance means that the path is treated similar by the model of any class. The first requirement is not as easy to fulfil, but is provided with high probability after a certain length of the path is reached.

With increasing *length(s)*, $p(s \mid C_i)^{\frac{1}{length(s)}}$ becomes less sensitive to the multiplication of a further factor within in the calculation of $p(S|C_i)$. The chance of a single node changing the predicted class and keeping up the variance at the same time is therefore decreasing with increasing *length(s)*. A topic that is more likely to be found in a site class different from the currently predicted class of *s*, will most likely decrease the variance. Thus, after a few nodes on a path *s* a decreasing value of *weight(s)* indicates the appearance of a topic less specific for the current site class. Now our first proposition can be applied, i.e. the path can be pruned.

Due to the important role *length(s)* plays for estimating the importance of the observed path *s*, it is an essential ingredient for the pruning criterion. Let s_1, s_2 be paths within the site tree *S*, where s_2 is an extension of s_1 by exactly one node. Then we stop the traversal at the last node of s_2 iff:

$$weight(s_2) < weight(s_1) \cdot \frac{length(s_2)}{\omega} \text{ with } \omega \in \Re^+.$$

For suitable values of ω ($\omega \geq 3$), the criterion will very likely allow the extension of shorter paths, which should not be pruned for the following reasons. Due to the small number of nodes, the membership can be influenced strongly by the classification error of the preprocessing step. Furthermore, our weight function can not recognize a change in the membership in very short paths. Last, the question of the importance of those paths for site classification cannot be decided in such an early state. Thus, applying the pruning rule makes no sense until descending some nodes along every path. Figure 3 illustrates the proposed pruning method on a small sample web site tree, in particular it shows the *weight*(s) values for each path *s*.

For a path *s* with *length(s)* smaller than ω, this rule will stop the traversal only, if a relevant decrease in variance is observed. As mentioned above, this is interpreted as a change of the site class and we can prune any following nodes due to our first

Table 1: Results on complete web sites using 10-fold cross-validation.

classifier	accuracy	precision other	recall other	precision IT-service provider	recall IT-service provider	precision florist	recall florist
classification of superpages	55.6 %	0.800	0.321	0.475	0.892	0.565	0.619
2-Order Markov tree	76.7 %	0.729	0.919	0.852	0.622	0.833	0.476
Naive Bayes	78.7 %	0.741	0.946	0.882	0.608	0.923	0.571
1-Order Markov tree	81.7 %	0.791	0.953	0.850	0.557	1.000	0.917
C4.5	82.6 %	0.802	0.902	0.831	0.730	1.000	0.762
0-order Markov tree	86.0 %	0.827	0.938	0.889	0.757	1.000	0.810
0-order Markov tree pruned (57 % of all pages)	87.0 %	0.840	0.938	0.956	0.770	1.000	0.857

proposition. For *length(s)* $\geq \omega$, the criterion is likely to prohibit the extension of a path unless a topic occurs that can significantly raise the variance. With increasing *length(s)* it is getting more and more unlikely that an additional factor can increase the variance strongly enough. Due to the required growth of variance and the decreasing influence of the additional factor, most paths are cut off after a certain length. This corresponds to the requirement made by our second proposition that a path will not provide general information about the class of the web site after a certain length. Hence, we avoid reading large subtrees without any impact on site classification.

The parameter ω is used to adjust the trade-off between classification accuracy and the number of web pages downloaded. Since the tolerance for the change of *weight(s)* depends on the ratio $\frac{length(s)}{\omega}$, ω is the length from which on an extension of the path is only permitted, if the variance increases. Thus, a good estimate for ω is the distance from the starting page in which the relevant information is assumed. Our experiments (see section 6) will show that the interval of reasonable values for ω is relatively wide and that the choice of ω is not delicate.

Pruning is not only a means of increasing the efficiency of web site classification, but it can also improve the classification accuracy. When classifying a complete web site, all of the introduced methods (with the exception of Markov trees with k≥1) consider all pages equally important and independently from their position within the site. Thus, unspecific subtrees can drive the classification process into the wrong direction. By providing an effective heuristic to disregard areas that are unlikely to contain the relevant information, the classifier gets a better description of the web site and therefore, it will offer better accuracy. To conclude, the introduced pruning rule tries to cut off misleading areas from the web site tree and thus, can reduce the processing time and also increase the classification accuracy.

6. EXPERIMENTAL EVALUATION

This section presents the results of our experimental evaluation of the proposed methods of web site classification. As mentioned before our running application is the categorization of corporate web sites belonging to 2 different trades. Our testbed consisted of 82,842 single HTML-documents representing 207 web sites. For the considered trades we chose florists and IT-service providers to have a significant distinction in the business. The distribution of the web site classes was: 112 “other”, 21 “florist” and 74 “IT-service provider”. The “other”-sites were taken randomly from various categories in yahoo[11]. To make the experiments reproducable, the downloaded information was stored locally. To classify the pages into the topics listed in section 4, we labelled about 2 % of the pages in the testbed and obtained a classification accuracy of about 72 % using 10-fold cross-validation with Naive Bayes on the manually labelled pages. As implementation for this and the rest of the standard algorithms, we used the well-implemented weka-package [10]. The remaining 98 % of the pages were labelled by Naive Bayes based upon this training set.

The evaluation of our methods consisted of two main parts. First, we compared the methods with respect to their classification accuracy on complete sites. The second part shows the impact of our pruning method and its parameter ω on the classification accuracy using the 0-order Markov tree.

Table 1 shows the overall classification accuracy as well as precision and recall for the single site classes. Since the superpage approach provided only an accuracy of about 55 %, it seems not to be well-suited for web site classification. On the other hand, all approaches based on the preprocessing step (introducing page class labels etc.) obtained reasonable results. The best method using the complete web site turned out to be the 0-order Markov tree which yielded 3.4 % more classification accuracy than the C4.5 [9] decision tree classifier. It also clearly outperformed the 1-order Markov tree by 4.3%. As a comparison the 0-order Markov tree, applying the introduced pruning method, increased the accuracy by one percent to 87 % on reading only 57 % of the data.

Our experimental evaluation demonstrates that using higher values than 0 for the order *k* did not improve the results, when applying a Markov tree classifier. This is due to the following reasons. First of all, the above mentioned problem of “phantom paths” increases with the length of the considered context (represented by the order k), depending on the error rate of page classification. We already noted that the overall error rate *p* of wrongly recognized pages in the site is about the same as the classification error for the single pages. But the probability of a correctly observed transition is only $(1-p)^2$, since it takes two correctly classified pages to recognize a transition. This problem gets worse with increasing order *k*. Thus, a distribution based

upon observed transitions will model reality only poorly. A further reason is founded within the specific characteristics of the application. The question of the class membership of a site is mostly decided in the area around the starting page. Since the nature of the information specifying the business is rather general, most designers are placing it near the starting page. Hence, the area relevant for the site classification is not characterized by long paths and the use of considering them is questionable here.

Table 2: Comparison of the accuracy decrease from classifying the test set and employing 10-fold cross-validation.

classifier	accuracy on training set	10-fold cross-validation	difference
0-order Markov tree	87.9 %	86.0 %	1.9 %
Naive Bayes	82.1 %	78.7 %	3.4 %
1-order Markov tree	85.5 %	81.7 %	3.8 %
2-order Markov tree	80.6 %	76.7 %	3.9 %
C 4.5	89.4 %	82.6 %	6.8 %
classification of superpages	65.7 %	55.6 %	10.1%

To get a notion of the tendency for overfitting of the employed classifiers, we compared the achieved accuracy, when classifying the training set itself, to the results on 10-fold cross-validation. The observed difference can give a clue of the accuracy loss, when using our methods on large amounts of so far unknown data.
The values displayed in table 2 show the accuracy achieved when training and testing is performed on the same data set. The next column shows the accuracy using 10-fold cross-validation, already presented in table 1. The largest difference between both values is observed for the superpage approach. This is a hint to a further problem of this method. Due to the large dimensionality of the used feature space, the number of training instances needed is rather high. Hence, decreasing them results in a large loss of accuracy. The further results show that the tendency for overfitting is the second largest in the decision tree approach of the C4.5 classifier. This is explained by the more accurate description of the training set of almost 90 %. Due to the heterogeneity of web sites this outstanding result decreases rapidly when using 10-fold cross-validation. The Markov tree approaches of orders 1 or 2 show a loss of about 4 %, which is most probably based on their higher demand of training instances. The best results here again achieved the 0-order Markov tree followed by Naive Bayes. The reason for the rather small loss of accuracy here is based on measuring all occurring features equally which fits our application best. Furthermore, the number of instances necessary for a representative training is rather low.

The second part of our experiments demonstrates the effects of the pruning method. Figure 4 shows the percentage of the downloaded web pages for varying values of ω. Additionally, the corresponding classification accuracy for each ω is depicted. Note that for values of $5 \leq \omega \leq 15$ the achieved accuracy exceeds 86 %, which is the accuracy when reading all web pages. The accuracy

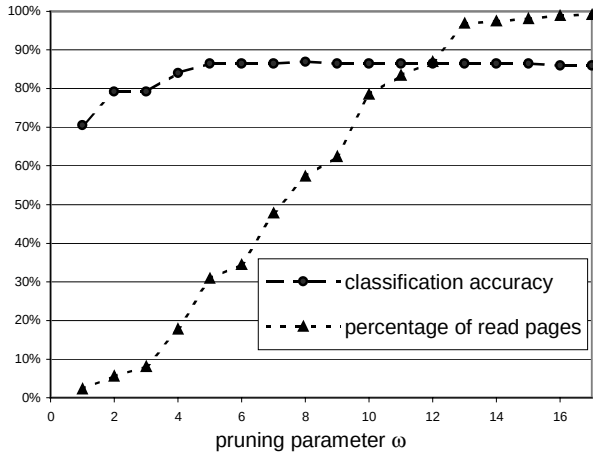


Figure 4: Effect of the parameter ω on the classification accuracy and the percentage of downloaded web pages.

for these values is around 86.4 %. For $\omega = 8$ it even reaches almost 87.0 % (table 1 shows the exact results). Thus, pruning a web site tree has the advantage of improving the accuracy of the 0-order Markov tree classifier.

The efficiency of the method can obviously be improved too. When reading only 30 % of the pages ($\omega = 5$), the classifier already provides the accuracy observed on the complete data or even exceeds it. Thus, reading more pages is not necessary. Even when choosing $\omega = 4$, i.e. reading only 17% of the pages, this classification method still outperforms the second best of the introduced approaches (84.1% against 82.4 % for C4.5). Note that reading only 17 % of the pages implies a speed-up factor of more than 5, since loading the web pages is the major cost of web site classification.

Another interesting point is the strong decrease of accuracy for very small numbers of read pages. A further experiment that tried to predict the class of a web site from the topic of the starting page alone, only achieved an accuracy of about 63 %. Hence, an area around the starting page large enough must be considered to achieve a good accuracy.

Although the effect of the chosen ω cannot be predicted, determining a reasonable choice is not very difficult. Since the accuracy did not react very sensitive to the varying values for ω after a reasonable value (about 4) was reached, it is relatively easy to make a choice that favours accuracy and/or efficiency.

The 0-order Markov tree classifier employing the introduced pruning rule therefore is able to offer superior classification accuracy, needing only a minor part of the I/O operations for reading complete web sites. Thus, the 0-order Markov tree with pruning outperforms all other methods by large factors with respect to efficiency.

7. CONCLUSIONS

In this paper, we introduced a new approach of web site mining for spotting interesting information on the world wide web. The idea was to change the focus from the consideration of single HTML-documents to the more complex objects of complete web sites, which are the objects of interest in various applications.

To spot the web sites we introduced several approaches for classification. The superpage approach is based upon the idea of treating a site as a single superpage merged from all the pages of the site. This superpage then can be classified like a normal web page.

Furthermore, we introduced a preprocessing step transforming a raw web site into a labeled web site tree. The preprocessing step aggregates the information on the single pages to one of several specified topics. To determine the topic of a page we used well known techniques for text document classification. The topics were used for further classification afterwards and increased the classification accuracy significantly.

Based upon this preprocessing step, we presented two advanced approaches of web site classification. The first is the use of standard classifiers such as Naive Bayes or decision trees on the topic frequency vectors of a site. The second one was called *k*-order Markov tree classifier and was based on the concept of branching *k*-order Markov chains. Due to the classification error occurring while determining the topics, it turned out that the value for *k*, providing the best results, was 0. Though the 0-order Markov tree does not use any knowledge of the context of a page, it outperformed all other classifiers discussed here.

To reduce the number of pages that have to be considered during classification, we derived a method for pruning subtrees within a web site tree. This method is based on the following two propositions. First, if the probability of a site class decreases after the extension of the path, the following pages are not of general interest anymore and can be pruned. Second, a path not favoring any site class can be cut after some length, since its children will not provide any general information. These considerations were applied to the Markov tree approach which is the only approach discussed that can calculate predictions incrementally.

We would like to outline several directions for future research. The first one is the improvement of the proposed classification methods. Especially the better use of structural information may permit an increase of both classification accuracy and pruning capability. As shown, one factor complicating the exploitation of structure is the uncertainty of the labels within the site trees. To solve this problem, we plan to adapt hidden Markov models to handle the uncertainty within the site classifier itself. Another possible way to improve the accuracy of site classification is the introduction of taxonomies for site classes. Clearly, this could yield the same advantages as it did in page classification.

To raise the usability of our approach it will be essential to avoid manually labelling a huge amount of example pages. Here methods adapted from unsupervised learning or the automatic extraction of example pages from directory services like yahoo[11], may offer a less expensive method to determine the page classes.

Another direction for future work is the development of systems that use web site classification. One promising application is the use of web site classification in systems trying to detect special kinds of sites in the world wide web. Therefore, we plan to develop a focused web site crawler that is specialized in detecting complete web sites. Since many of the well-known techniques are not applicable on whole web sites, new techniques are required to solve the new problems and to exploit the new chances of this field. A further goal of a focused web site crawler is the use of the predicted site class as a filter when downloading single web pages from a site.

8. REFERENCES

[1] Chakrabarti S., Dom B. and Indyk P.: Enhanced hypertext categorization using hyperlinks, *Proceedings ACM SIGMOD*, 1998.

[2] Craven M., DiPasquo D., Freitag D., McCallum A., Mitchell T., Nigam K., and Slattery S.: Learning to Construct Knowledge Bases from the World Wide Web, *Artificial Intelligence*, Elsevier, 1999.

[3] Deshpande M., Karypis G.: Evaluation of Techniques for Classifying Biological Sequences, *Proceedings PAKDD*, 2002.

[4] DMOZ open directory project, http://dmoz.org/

[5] Joachims T.: Text Categorization with Support Vector Machines: Learning with Many Relevant Features, *Proceedings European Conference on Machine Learning*, 1998.

[6] Lesh N., Zaki M. J., Ogihara Mitsunori: Mining Features for Sequence Classification, *Proceedings ACM SIGKDD* , San Diego, CA, August 1999.

[7] McCallum A., Nigam K.: A Comparison of Event Models for Naive Bayes Text Classification, *Proceedings of AAAI-98 Workshop on Learning for Text Categorization*, 1998.

[8] Menshikov M.V., Volkov S.E.: Branching Markov Chains: Qualitative Characteristics, 1997, *Markov Processes Relat. Fields. 3 1-18.*

[9] Quinlan J.R.: C4.5 : Programs for Machine Learning, 1993, *Morgan Kaufmann*, San Mateo, CA

[10] Witten I. H., Eibe F.: Data Mining: Practical Machine Learning Tools and Techniques with Java Implementations, 1999, *Morgan Kaufmann*, http://www.cs.waikato.ac.nz/ml/weka/

[11] Yahoo! Directory Service, http://www.yahoo.com/

[12] Yang Y., Liu X.: A Re-Examination of Text Categorization Methods, *Proceedings ACM SIGIR*, 1999.

[13] Zaki M. J.: SPADE: An Efficient Algorithm for Mining Frequent Sequences, *Machine Learning Journal*, pp 31-60, Vol. 42 Nos. 1/2, Jan/Feb 2001