Mining HTML Pages to Support Document Sharing in a Cooperative System

Donato Malerba, Floriana Esposito, and Michelangelo Ceci

Dipartimento di Informatica, Università degli Studi
via Orabona, 4 - 70126 Bari - Italy
{malerba, esposito, ceci}@di.uniba.it

Abstract. In this paper, the problem of classifying HTML documents is investigated in the context of a client-server application, named WebClass, developed to support the search activity of a geographically distributed group of people with common interests. The two main issues studied in the paper are the selection of some features to represent HTML documents and the construction of the classifiers. A new feature selection technique is presented and its interaction with different classifiers is experimentally studied. Results show that performance improves even with simple classifiers and the proposed feature selection technique compares favorably with respect to other well-known approaches.

Keywords: mining from semi-structured data, cooperative systems, feature extraction and selection.

1 Introduction

There is an increasing interest among researchers in the possibility of sharing information among distributed work groups in organizations (*virtual groups*) – providing some form of shared or common information "space". However, this is only possible if it is based on a shared understanding of concepts and there is a shared language of discourse. An *ontology* can be used as such a basis for knowledge sharing since it formally represents the vocabulary and conceptual structure of the domain. Formal ontologies were originally developed by the knowledge engineering community to model domain knowledge and have recently been put to a variety of different uses, e.g. in the development of large distributed electronic trading groups, or in the classification of some Web pages such as those in Yahoo.

Ontologies enable knowledge sharing and re-use, but suffer from an imbalance between efforts and benefits from the individual's point of view. The ontological annotation of documents to be shared with other users facilitates information retrieval and integration, but this is an effective approach only when information providers have a thorough comprehension of the underlying shared ontology. End users find it difficult to understand ontologies codified in formal languages, especially abstract, top-level definitions of concepts, relations and properties. An experimental study on manual indexing for Boolean information retrieval systems has shown that the degree of overlap in the keywords selected by two similarly trained people to represent the

A.B. Chaudhri et al. (Eds.): EDBT 2002 Workshops, LNCS 2490, pp. 420-434, 2002.

same document is not higher than 30% on average [4]. This is an important limitation on the development of cost effective and highly scalable co-operative information repositories.

Data mining techniques give workspace users the power of automated classification of documents according to a given ontology. Since members of a virtual group are usually able to provide sets of pre-classified documents, but no operational definition of how a document should be categorized, the classification of documents can be delegated to an intermediary that is properly trained on the set of given documents (see Figure 1).

Data mining techniques are usually designed to work on relational databases, that is, on structured data. Therefore, the application of data mining techniques to unstructured or semi-structured data poses additional research problems. In particular, the application of data mining techniques to the discovery of useful information from the Web contents is referred to as *Web content mining*. Web content data mostly consist of semi-structured data such as HTML documents. Although multimedia components of HTML documents, such as pictures, videos, animations and audios, take 80% of a typical Web site, most of the work in Web content mining has been focused on free or semi-structured text.

In this paper, we present WebClass, a client-server application that performs the automated classification of Web pages on the basis of their textual content. This application has been designed to support the search activity of a geographically distributed group of people with common interests. Two categories of remote users are allowed in WebClass: Administrators and final users. The former are allowed to train the system by providing a set of pre-classified Web pages in HTML format, where classes correspond to topics of interests for a final user. Training is performed by means of a graphical user interface that supports browsing functions together with functions typical of a learning system, such as parameter setup, feature extraction and

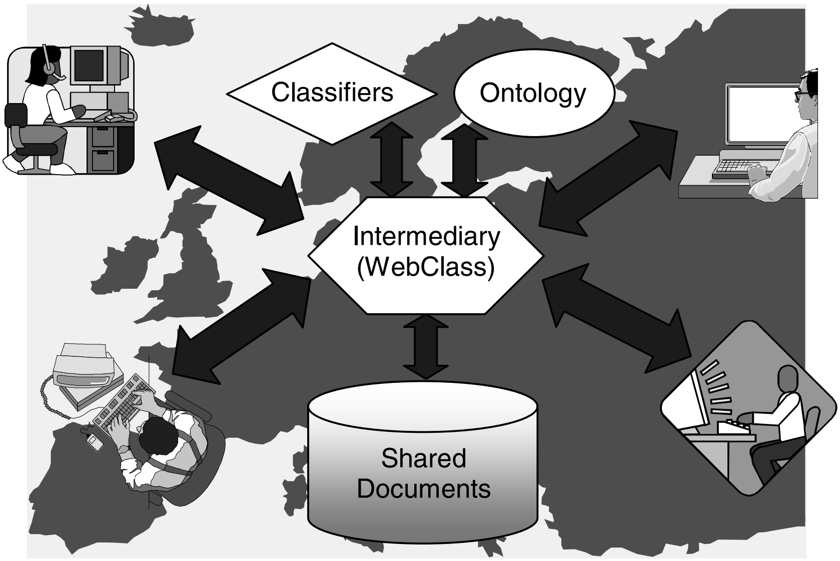


Fig. 1. An intermediary-based architecture for computer-supported collaborative work.

selection, definition of training and test set, classifier generation and testing. Learned classifiers are subsequently used by final users while browsing the Web. Final users can either decide to store the document in a document base or discard it. In the former case, the document is automatically classified by WebClass according to the pre-defined set of classes.

The automated classification of Web pages requires the solution of two problems:

1. The definition of a representation language for HTML pages, and
2. The construction of a classifier that is able to categorize new Web pages on the basis of their representation.

As to the first problem, WebClass adopts a feature vector representation of Web documents, where each single feature corresponds to a distinct term extracted from the training documents. Features are determined by means of a complex pre-processing phase, which includes both a term extraction and a term selection process.

For the second problem, Webclass has four alternative ways of assigning a Web page to a class:

1. By sequentially testing feature values according to a *decision tree*.
2. By computing the distance from the *centroids* of the different classes.
3. By computing a posterior probability by means of a naive Bayes classifier.
4. By computing the distance from all training documents (*k-nearest-neighbor*).

In the first three ways, a training phase is necessary to build either the decision tree or the centroids or the prior probabilities. In the fourth way, specific instances rather than previously built classifiers are used during the prediction task.

In this paper, some issues in the construction of *WebClass* classifiers are presented. After a brief introduction to both WebClass client-server architecture and the logical design of the underlying database, a novel technique for the selection of relevant words in the preprocessing phase is introduced in Section 3, while the different approaches adopted for the automated categorization are described in Section 4. Empirical results on different training sets are reported and discussed in Section 5.

2 The WebClass System

The architecture of WebClass is based on the classical three-tier model (see Figure 2). The first tier is a Java-enabled Web browser, through which it is possible to run a java applet. The applet supports several user activities, namely browsing the Web, classification and storage of interesting documents, query and retrieval of stored documents, appropriate presentation of retrieved documents on the medium, construction of classifiers (training phase), as well as typical management functions (create/modify/delete system users). The second tier is a servlet middleware, namely a Web server running Java servlets. The Java servlet is able to access the database and return dynamically generated HTML pages. The third tier is the back-end database server. The Java servlet can access information stored by means of the JDBC-ODBC driver integrated in the development environment JDK 1.3.

Users can query the database either by writing a sentence in English or by browsing documents in a class. In the former case, the system returns a list of document identifiers (ID) and titles, ranked according to a relevance measure. By clicking on a row, the document class (category) and the original URL of the Web document are displayed (see Figure 3), and by clicking on "View This Document" it

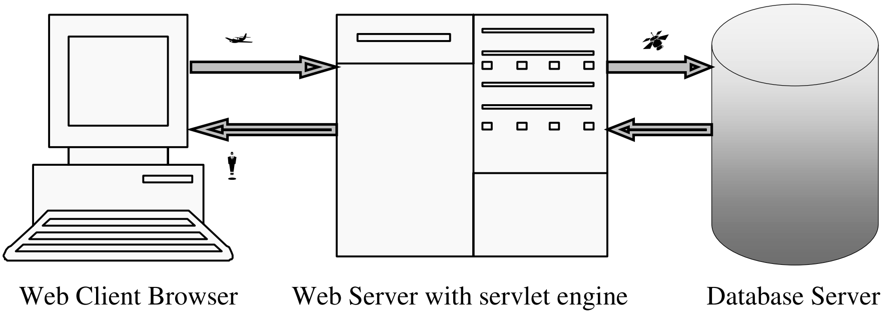
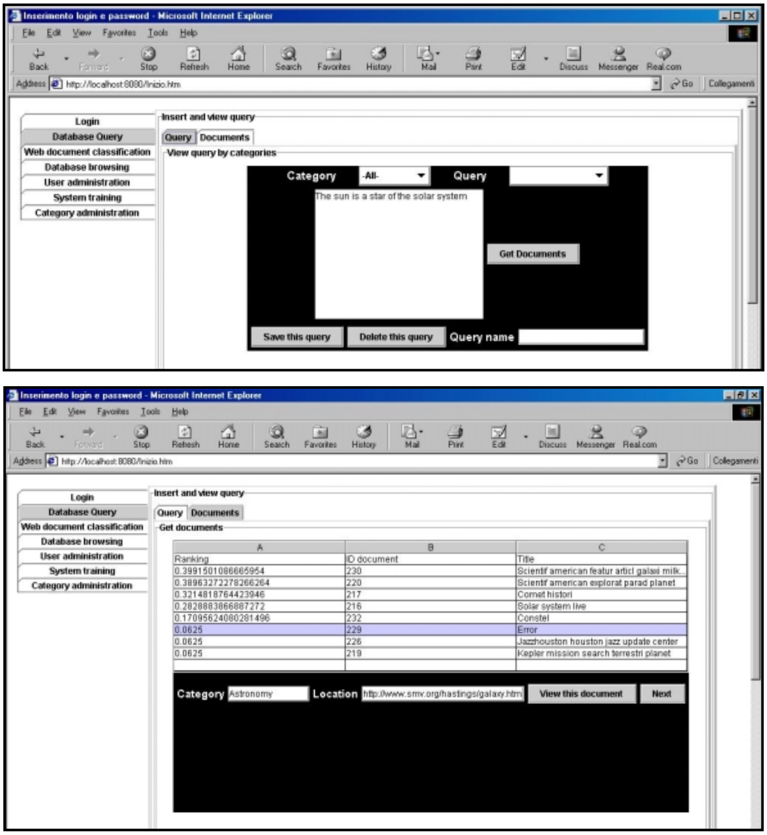


Fig. 2. WebClass three-tier architecture. In the client-side, the user interacts with the system by means of an applet. The applet can send parameterized requests to the appropriate Java servlet running on the Web server, namely jsdk2.1 (step 1). The Java servlet interprets the request, processes it and constructs an SQL statement, which is passed to the database server using the Java Database Connection (JDBC) (step 2). The database server executes the SQL statement and returns a result set to the Java servlet (step 3). The Java servlet processes the result set and communicates the results to the applet.


Inserimento login e password - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Forward Stop Refresh Home Search Favorites History Mail Print Edit Discuss Messenger Real.com
Address
Go
Collegamenti
Login
Database Query
Web document classification
Database browsing
User administration
System training
Category administration
Insert and view query
Query
Documents
View query by categories
Category
Query
Get Documents
Save this query
Delete this query
Query name

Address: http://localhost:8080/inizio.htm

Category: -All-

The sun is a star of the solar system

Inserimento login e password - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Forward Stop Refresh Home Search Favorites History Mail Print Edit Discuss Messenger Real.com
Address
Go
Collegamenti
Login
Database Query
Web document classification
Database browsing
User administration
System training
Category administration
Insert and view query
Query
Documents
Get documents
View this document
Next

Address: http://localhost:8080/inizio.htm

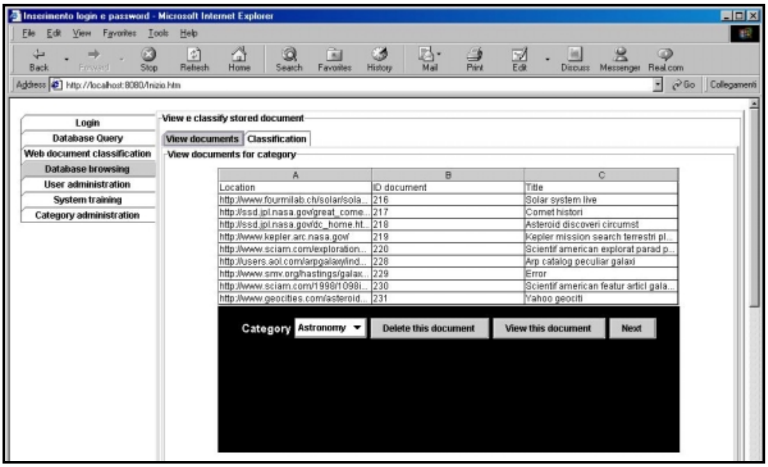
A	B	C
Ranking	ID document	Title
0.3991501086665954	230	Scientif american featur articl galaxi milk...
0.3896327227826264	220	Scientif american explorat parad planet
0.3214818764423946	217	Comet histori
0.2828883866887272	216	Solar system live
0.17095624080281496	232	Constel
0.0625	229	Error
0.0625	226	Jazzhouston houston jazz update center
0.0625	219	Kepler mission search terrestri planet

Category: Astronomy
Location: http://www.smv.org/hastings/galaxy.htm

Fig. 3. An example of textual query (above) and the set of retrieved documents (below).

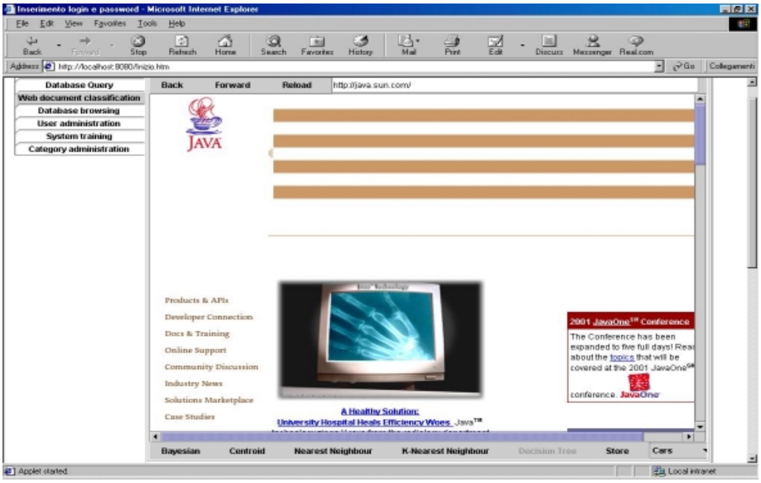
is possible to view the stored document. Textual queries can be saved in the user workspace for future use.

The second mode of database querying requires the specification of a class of documents to be browsed (see Figure 4). WebClass works as an intemediary when users browse the Web through the system and categorize documents by means of one of the classification techniques available (see Figure 5). The extraction of features to be used for document representation and classification and the possible construction of classifiers is another functionality made available to remote users with administration rights. In this case it is necessary to select training documents and possibly test documents either from the database or from the Web (see Figure 6). Once the training and test sets have been fully defined, it is possible to activate the


Inserimento login e password - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Forward Stop Refresh Home Search Favorites History Mail Print Edit Discuss Messenger Real.com
Address http://localhost:8080/Inizio.htm
Go Collegamenti
Login
Database Query
Web document classification
Database browsing
User administration
System training
Category administration
View e classify stored document
View documents
Classification
View documents for category
Category
Astronomy
Delete this document
View this document
Next

A	B	C
Location	ID document	Title
http://www.fourmilab.ch/solar/sola...	216	Solar system live
http://ssd.jpl.nasa.gov/great_come...	217	Comet histori
http://ssd.jpl.nasa.gov/dc_home.ht...	218	Asteroid discoveri circumst
http://www.kepler.arc.nasa.gov/	219	Kepler mission search terrestri pl...
http://www.sciam.com/exploration...	220	Scientif american explorat parad p...
http://users.aol.com/arpgalaxy/ind...	228	Arp catalog peculiar galaxi
http://www.smv.org/hastings/galax...	229	Error
http://www.sciam.com/1998/1098i...	230	Scientif american featur articl gala...
http://www.geocities.com/asteroid...	231	Yahoo geociti

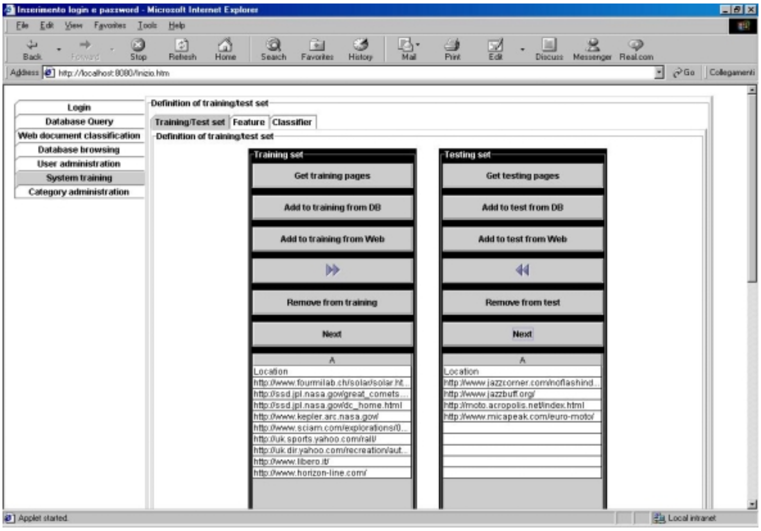
Fig. 4. Browsing documents by categories.



2001 JavaOne℠ Conference

The Conference has been expanded to five full days! Rea about the topics that will be covered at the 2001 JavaOne℠ conference. JavaOne

Fig. 5. Browsing documents in the Web and tools for automated classification (see buttons below). The class is reported in the bottom right-hand corner.



A
Location
http://www.fourmilab.ch/solar/solar.ht...
http://ssd.jpl.nasa.gov/great_comets...
http://ssd.jpl.nasa.gov/dc_home.html
http://www.kepler.arc.nasa.gov/
http://www.sciam.com/explorations/0...
http://uk.sports.yahoo.com/rall/
http://uk.dir.yahoo.com/recreation/aut...
http://www.libero.it/
http://www.horizon-line.com/

A
Location
http://www.jazzcorner.com/noflashind...
http://www.jazzbuff.org/
http://moto.acropolis.net/index.html
http://www.micapeak.com/euro-moto/

Fig. 6. Browsing documents by categories.

feature extraction process and, possibly, the construction of some classifiers. This activity also requires the specification of some parameters by means of the interface. The required parameters and their meanings will be explained in the next sections.

3 The Learning Issues

Many issues are related to the goal of building a Web page classifier. Firstly, it is necessary to determine which kind of information in a Web page is relevant. In this work, it is assumed that the relevance of a page for the group of users depends exclusively on both the *textual content* and the HTML *layout structure* of the Web page. Hence, the following factors are not considered in this study:

- External factors, such as the novelty or the reliability of the information in the document.
- Multimedia components of a page, such as pictures, videos, animations and audios.
- Contextual information reported in a link to an HTML document [2].
- Possible dependencies among classes of Web documents. This means that classes considered in this study are not hierarchically related in the user ontology.

Thus, the learning problem can be formulated as follows:

Given

- a set $\mathcal{C}$ of classes of documents $C_1, C_2, \ldots, C_r$,
- a finite Web space W structured as a directed graph over Web documents,
- a set of Web documents $D \subseteq W$ described in a language L_D,

Find
a model or hypothesis H which maps Web documents to a class and maximizes predictive accuracy on possibly unseen documents in the Web space W.

The second issue concerns the representation language L_D adopted for Web documents. Typically, documents are represented as feature vectors, where features correspond to specific words or groups of words. This representation, also known as *bag-of-words*, is that adopted by WebClass, although two variants have been tested: *Plain* frequency vector representation and *emphasized* representation. The latter takes into account the information related to all the HTML tags enclosing occurrences of words.

The selection of features is the third issue. Thousands of features can be generated, even from a small set of pages. Generally, only a subset of features is relevant, and their selection has been proven beneficial for the computational complexity and the accuracy of the page classifiers [8]. In the next section, a feature selection process that extracts orthogonal sets of features for each document class is described and a novel scoring measure, appropriate for feature selection in multi-class text categorization problems, is presented.

The last issue concerns the construction of classifiers. Once again, several solutions have been proposed in the literature: Bayesian classifiers [14], decision trees [1], some adaptations of Rocchio's algorithm to text categorization [7], and *k*-nearest neighbor [11]. In this study four classifiers are considered. They are based on three different views of classes: *k*-nearest neighbor (extensional view), decision trees and Bayesian classifiers (classical intensional view), and centroids (exemplar intensional view).

3.1 The Preprocessing Phase

Initially, all training documents are tokenized, and the set of tokens (words) is filtered in order to remove HTML tags, punctuation marks, numbers and tokens of less than three characters. The basic idea is to select relevant tokens to be used in the bag-of-words representation. These tokens will be called *features*. As already observed in Luhn's seminal work [10], when distinct words in a textual document are arranged in decreasing order of their frequency of occurrence, the distribution satisfies Zipf's Law [20], that is, the product *rank*frequency* is constant. Luhn conjectured that the relevant words extracted from a document text would peak in the middle range, and further proposed to use words with medium frequency, because high- and low-frequency words are not good content identifiers. These considerations lead to a standard procedure in text pre-processing, which is implemented in WebClass:

1. Removal of words with high frequency, or stopwords, such as articles, prepositions, and conjunctions.
2. Removal of suffixes, such as those used in plurals (e.g., *-s*, *-es* and *-ies*), gerund (*-ing*), simple past (*-ed*), and so on.
3. Determination of equivalent stems (stemming), such as *analy* in the words *analysis*, *analyses*, *analyze*, *analyzing* and *analyzer*.

For the first step, stopwords used by WebClass have been taken from Glimpse (glimpse.cs.arizona.edu), a tool used to index files by means of words, while for the last two steps, the algorithm proposed by Porter [15] has been implemented.

Many approaches have been proposed in the literature on information retrieval for the identification of relevant words to be used as index terms of documents [16]. Most of them simply score words according to some measure and select the best firsts. However, techniques proposed for information retrieval purposes are not always appropriate for the task of document classification, also known as text categorization. Indeed, we are not interested in words characterizing each single document, but we look for words that distinguish a class of documents from other classes. Generally speaking, the set of words required for classification purposes is much smaller than the set of words required for indexing purposes.

For two-class problems, Mladenic [12] compared scoring measures based on the *Odds ratio*[1] and those based on *information gain*,[2] leading her to favor the former. For multi-class problems, as in the case of WebClass, an extension of the well-known TF-IDF measure to text categorization, originally proposed for information retrieval purposes [17], has been suggested [6]. However, this measure is not intended for feature-selection purposes, but for the definition of a probabilistic classifier of documents.

The feature selection algorithm implemented in WebClass is based on a variant of TF-IDF. Given the training document *d* of the *i*-th class, for each token *t* the frequency *TF(i,d,t)* of the token in the document is computed. Then, for each class *i* and token *t*, the following statistics are computed:

- *MaxTF(i,t)*, the maximum value of *TF(i,d,t)* on all training documents *d* of class *i*;
- *PF(i,t)*, the *page frequency*, that is, the percentage of documents of class *i* in which the token *t* occurs.

The union of sets of tokens extracted from Web pages in one class defines an "empirical" *class dictionary* used by documents on the topic specified by the class. By sorting the dictionary with respect to *maxTF(i,t)*, words occurring frequently only in one long HTML page might be favored. Indeed, Web page authors usually "hide" a number of occurrences of "key" words in the HTML code, in order to force search engines to rank that page in the first positions of the returned lists of Web references. By sorting each class dictionary according to the product $maxTF(i,t)*PF(i,t)^2$, briefly denoted as $MaxTF\text{-}PF^2$ (Max Term Frequency - Square Page Frequency) measure, the effect of this phenomenon is kept under control.[3] Moreover, common words used in documents of a given class will appear in the first entries of the corresponding class

[1] For a given token *t*, $Odds\text{ - }ratio(t) = \log\frac{odds(t|C_1)}{odds(t|C_2)} = \log\frac{P(t|C_1)(1-P(t|C_2))}{(1-P(t|C_1))P(t|C_2)}$.

Singularities are handled as in [18].

[2] For a given token *t*, *Information-Gain(t) = H(C) – H(C|t)*, where *H(C) (H(C|t))* is the entropy in the document set before (after) the usage of the token *t* for splitting the document set into subsets.

[3] The plain *PF(i,t)* factor was also used, but was found to reduce performance slightly. For the small sets of training documents we considered in our experiments, the term *PF(i,t)* might not be small enough to reduce the effect of very frequent words in single documents.

dictionary. Some of these words are actually specific to that class, while others are simply common English words (e.g., "information", "unique", "suggestion", "time" and "people") and should be considered as *quasi-stopwords*. In order to move quasi-stopwords down in the sorted dictionary, the $MaxTF\text{-}PF^2$ of each term is multiplied by a factor $ICF=1/CF(t)$, where $CF(t)$ (*category frequency*) is the number of class dictionaries in which the word t occurs. In this way, the sorted dictionary will have the most representative words of each class in the first entries, so that it will be sufficient to choose the first N words per dictionary, in order to define the set of attributes. Once the class dictionaries are determined, a unique set of features is selected to represent documents of all classes. Currently, the administrator defines the number of features.

Once the set of features has been determined, training documents can be represented as feature vectors. WebClass can adopt two different representations: Plain and emphasized. In the latter representation, the frequency of a word is multiplied by an *emphasis factor*, which is computed in two different ways:

1. *Additive* emphasis: The emphasis factor of a word is computed by summing weights associated to the HTML layout structures where the word occurs;
2. *Multiplicative* emphasis: The product of weights is considered.

The effect of the emphasis on the *classification* of Web pages is reported in [6]. Experimental results do not lead to a clear conclusion that any classification technique generally benefits from the HTML structural information. In other words, the partial structure existing in HTML pages does not seem to convey information useful for the classification. Similar results have bee reported in [5], where Hidden Markov Models have been applied to capture information possibly conveyed by the HTML tags. For this reason, only the plain representation will be considered in the section on experimental results. The importance of the HTML structure in document *understanding*, that is the interpretation of the content of some parts of an HTML page, is still an open problem. Some results obtained with systems developed to convert HTML pages into XML documents [3] seem to indicate a certain relevance of fonts, colors, and table structure.

3.2 The Classification Methods

WebClass has four alternative ways of assigning a Web page to a class:

1. By sequentially testing feature values according to a *decision tree*.
2. By computing the distance from the *centroids* of the different classes.
3. By computing the Bayesian posterior probability.
4. By computing the distance from all training documents (*k-nearest-neighbor*).

In the first three ways, a training phase is necessary to build either the decision tree or the centroids or to estimate probability distributions. In the fourth way, training instances are used only when the system is asked to classify a new page.

WebClass generates a univariate decision tree, by means of the system OC1 [13], for each class C_i, by considering training documents of class C_i as positive examples and all remaining documents as negative examples.[4] Therefore, the result of the classification process can be: 1) no classification (the document is not stored); 2) single classification (the document is stored as an instance of the corresponding class); 3) multiple classification (the document is stored as an instance of more than one class).

According to the Bayesian theory, the optimal classification of a document *d* assigns *d* to the class $C_i \in \mathcal{C}$ maximizing the posterior probability $P(C_i|d)$. Under the assumption that each word in *d* occurs independently of other words, as well as independently of the text length, it is possible to estimate the posterior probability as follows [7]:

$$P(C_i \mid d) = \frac{P(C_i) * \prod_{t \in F} P(t \mid C_i)^{TF(d,t)}}{\sum_{C' \in \mathcal{C}} P(C') * \prod_{t \in F} P(t \mid C')^{TF(d,t)}} \quad (1)$$

where *F* is the set of features selected in the pre-processing phase, and *TF(d,t)* is the frequency of the word *t* in *d*. The prior probability $P(C_i)$ is estimated as the percentage of documents in C_i, while the likelihood $P(t \mid C_i)$ is estimated according to Laplace's Law of succession:

$$\hat{P}(t \mid C_i) = \frac{1 + TF(i,t)}{\mid F \mid + \sum_{t' \in F} TF(i,t')} \quad (2)$$

with *TF(i,t)* denoting the frequency of word *t* in all documents of class C_i. During the training phase, all probabilities $P(C_i)$ and $P(t \mid C_i)$ are estimated on the training set, so that they can be readily used when a new document *d* has to be classified.

The computation of centroids is based on a simple formula. Let $v_1, v_2, \ldots, v_{ni}$ be the feature vectors corresponding to the n_i training pages of class C_i. Let $v_j(k)$ denote the *k*-th component of v_j, *k*=1, 2, …, *N*. Then the *centroid* is an *N*-dimensional feature vector, whose components are computed by averaging on the corresponding components of the training documents. In order to classify a document, the centroid most similar to the document description has to be found. The similarity measure considered is the *cosine correlation* [9], which computes the angle spanned by two vectors (the document and the centroid). The mathematical formulations of both the centroid and the similarity measure are the following:

$$P(k) = \frac{\sum_{j=1}^{n_s} v_j(k)}{n_j} \qquad sim(P,V) = \frac{\sum_{k=1}^{N} P(k)V(k)}{\sqrt{\sum_{k=1}^{N} P(k)^2 \sum_{k=1}^{N} V(k)^2}} \quad (3)$$

[4] It is worthwhile observing that the construction of the classifiers may be cast as a set of two class problems, even though the feature selection process is based on a multi-class approach. Indeed, thanks to the common set of features for all classes, it is possible to perform a centroid-based classification and to compare it with more sophisticated techniques, such as decision trees.

The cosine correlation returns a particularly meaningful value when vectors are highly dimensional and features define orthogonal directions. In WebClass both conditions are satisfied, though orthogonality refers to the group of features extracted from each class dictionary rather than to the individual features. The cosine correlation is also used in the fourth classification approach, namely the *k*-nearest-neighbor.

All the above classifiers assign a document *d* to one of the "meaningful" classes $C_i \in \mathcal{C}$. Nevertheless, in some application domains some documents cannot or should not be meaningfully assigned to any class. Although the problem can be operatively dealt with by adding a "reject" class to $\mathcal{C}$, where all such documents might be collected, from the point of view of document classification this is not always correct. This reject class represents "the rest of the world" and the computation of a posterior probability, as well as the computation of a centroid, would make little sense. By assuming that documents in the reject class have a low posterior probability, compared with other classes, or are very distant from the centroids/examples of other classes, the problem can be reformulated in a different way, namely how to define some thresholds for both posterior probabilities and distances. WebClass uses documents of the reject class in the training set to compute the optimal thresholds. Optimality is evaluated with respect to *F-score*,[5] a measure that synthesizes two important parameters used in information retrieval, namely *recall* and *precision*.

4 Design of the Experiments and Results

A set of experiments was designed to test the performance of WebClass. Documents used in the test are a subset of the collection known as "Reuters-21,578, Distribution 0.1" (http://www.research.att.com/~lewis/reuters21578.html) This is a set of 21,578 news items provided by the German agency Reuters, manually catalogued by personnel from Reuters Ltd. and Carnegie Group Inc. There are 135 distinct classes, mainly on economical topics, but in our experiments we considered only ten of them, namely *acq*, *coffee*, *crude*, *earn*, *interest*, *money-fx*, *money-supply*, *ship*, *sugar* and *trade*. They are the most represented classes associated to the first 4,000 documents of the collection. All other classes are grouped in a generic *reject* class. From the first 4,000 news items in the collection about 3,500 documents are selected for the test set, by discarding empty documents as well as documents with multiple classifications. From the subsequent 10,000 documents three training sets of different size (about 2,000, 3,000 and 4,000, respectively) are extracted according to the same criterion applied for the test set (see Table 1). It is noteworthy that examples are unevenly distributed among classes. The first 14,000 news items in the Reuters corpus actually contain examples of many other classes, which are dropped because they are too poorly represented to lead to meaningful conclusions on system performance.

Fifty features are extracted for each class, totaling up to five hundred features for each experiment. Since there are about 22,500 unique terms in the Reuters collection (after performing stemming, stopword removal and conversion to lower case), the reduction is about 97,8%. This choice is coherent with the best result observed by Yang and Pedersen [19] for information gain based feature selection. Twenty-four

[5] F-Score=2*Precision*Recall/(Precision+Recall)

experiments were performed by varying the feature selection measure (*MaxTF-PF²-ICF* vs. average mutual information (AMI)[6] vs Odds-ratio), the training set size (2,000, 3,000, 4,000) and the method used to classify documents (centroids, nearest neighbor, k-nearest neighbor, Bayesian classifier). In this experimentation decision trees could not be tested because of the OC1 limit on processing examples described by no more than one hundred features.

The experimental results are shown in Figure 7, where the level of results is measured on the basis of an F-score. It is noteworthy that quite a simple classification method, based on the computation of a distance from class centroids, performs quite well when features are extracted by means of *MaxTF-PF²-ICF*. This empirically confirms that the set of selected features for each class has a good discriminant power. The combination information gain and Bayesian classifier outperform all methods only in the experiment on 2,000 training pages. Our results are at divergence with those reported in the work by Mladenic [12], according to which information gain performs worse than random feature selection. A closer look at words sorted by information gain shows that almost all best words are characteristic of a negative class value. These negative results prevented Mladenic from considering other measures similar to information gain, such as mutual information. Nevertheless, from our results we observe that average mutual information performs better than Odds ratio in some experiments. The difference might be due to the fact that, in our approach, features are extracted and selected by processing documents class by class, while Mladenic merged together words of both positive and negative examples.

Table 1. Distribution of selected Reuters news in training and test sets

Class	Training 2000	Training 3000	Training 4000	Test
acq	260	390	520	458
coffee	10	15	21	19
crude	44	66	88	77
earn	450	675	900	794
interest	20	30	39	33
money-fx	8	12	16	16
money-supply	20	30	40	35
ship	16	24	32	27
sugar	10	15	20	18
trade	18	27	37	32
Subtotal	***856***	***1284***	***1713***	***1506***
reject	1132	1690	2260	2017
Total	***1988***	***2974***	***3973***	***3526***

[6] Let W_i be a random variable denoting the presence/absence of the word (token) w_i in a document. It can take two values v_i, where $v_i \in \{w_i, \neg w_i\}$. Let C be a random variable with values in the set of user classes. The average mutual information is defined as follows [19]:

$$I(C,W_i) = H(C) - H(C|W_i) = -\sum_{c\in C} P(c)\log(P(c)) + \sum_{v_i\in\{w_i,\neg w_i\}} P(v_i)\sum_{c\in C} P(c|v_i)\log(P(c|v_i)) =$$

$$= \sum_{v_i\in\{w_i,\neg w_i\}}\sum_{c\in C} P(c,v_i)\log\frac{P(c,v_i)}{P(c)P(v_i)}$$

	Test 2000 (50 feature x category)		
	AMI	MaxTF-PF^2-ICF	Odds Ratio
Centroids	46.55	50.54	37.84
NN	44.17	43.84	34.30
7-NN	49.98	50.25	37.66
Naive-Bayes	52.83	47.32	35.57
	Test 3000 (50 feature x category)		
	AMI	MaxTF-PF^2-ICF	Odds Ratio
Centroids	47.80	53.69	42.59
NN	43.52	43.19	32.79
7-NN	49.23	48.77	37.70
Naive-Bayes	52.52	47.55	35.86
	Test 4000 (50 feature x category)		
	AMI	MaxTF-PF^2-ICF	Odds Ratio
Centroids	48.14	51.79	40.55
NN	44.96	43.17	28.20
7-NN	49.07	48.45	37.09
Naive-Bayes	50.16	47.88	36.28

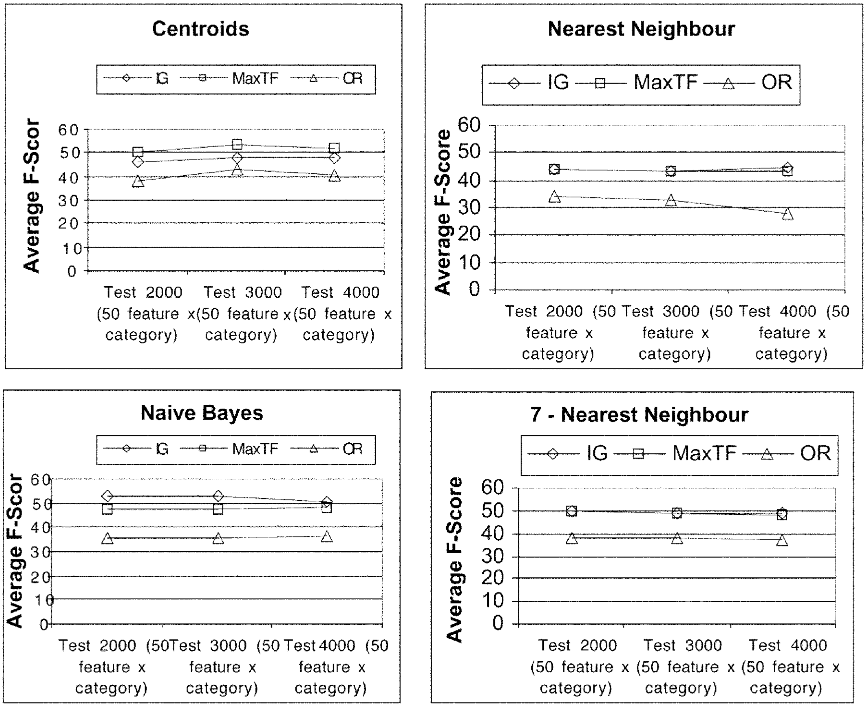


Fig. 7. Experimental results on the Reuters data.

Finally, we performed a further test according to the standard experimental design proposed by Lewis also known as "Modified Apté" (www.research.att.com/~lewis/). In this case, 5,754 cases were used for the training set, 2,861 for the test set and 722 for the reject set. The number of selected features was kept the same. Table 2 shows that the proposed feature selection based on *MaxTF-PF*^{2}*-ICF* outperforms the information gain in almost all experiments, with the exception of 7-NN.

Table 2. Experimental results on the Reuters data – "Modified Apté" test.

	IG	MaxTF-PF^2-ICF	Odds Ratio
Centroids	60.09	61.44	45.99
NN	66.12	66.48	44.10
7-NN	67.88	67.62	44.36
N. Bayes	62.76	67.58	11.64

5 Conclusions

In this paper, the problem of automatically classifying HTML documents, according to a set of pre-defined classes, has been investigated in the context of a client-server application developed to support Web document sharing in a group of users with common interests. The two main issues studied in the paper are the selection of some features to represent HTML documents and the construction of the classifiers. For the first issue, a novel technique for the selection of relevant features from training pages has been presented, while for the second issue several classifiers have been considered and a thresholding algorithm has been proposed in the case of a reject class. The interaction of the feature selection technique with different classifiers has been experimentally studied. Results show that performance improves even with simple classifiers and the proposed feature selection technique compares favorably with respect to other well-known approaches.

We are currently investigating some research issues related to the classification of HTML documents in a set of hierarchically related classes. An experimental study of a general method defined for the construction of a hierarchical classifier is planned for the near future, together with an experimental study on the document retrieval technique implemented in WebClass and not described in this paper.

References

1. C. Apté, F. Damerau, & S.M. Weiss (1994). Automated learning of decision rules for text categorization. *ACM Transactions on Information Systems*, 12(3), 233-251.
2. G. Attardi, S. Di Marco, D. Salvi, & F. Sebastiani (1998). Categorisation by context. *On-line Proceedings of the 1st International Workshop on Innovative Internet Information Systems*, http://www.idt.ntnu.no/~monica/iii-98/proceedings_on_line.html.

3. R. Baumgartner, S. Flesca, G. Gottlob (2001). Supervised Wrapper Generation with Lixto. *Proc. of the 27th Int. Conf. on Very Large Data Bases*, 715-716.
4. C. Cleverdon (1984). Optimizing convenient online access to bibliographic databases. *Information Services and Use*, 4, 37-47.
5. M. Diligenti, M. Gori, M. Maggini & F. Scarselli (2001). Classification of HTML Documents by Hidden Tree-Markov Models. *Proc. of the 6th Int. Conf. on Document Analysis and Recognition ICDAR'01*, IEEE Computer Society Press, Los Vaqueros, CA.
6. F. Esposito, D. Malerba, L. Di Pace, & P. Leo (2000). A Machine Learning Approach to Web Mining, In E. Lamma & P. Mello (Eds.), *AI*IA 99: Advances in Artificial Intelligence*, Lecture Notes in Artificial Intelligence, Vol. 1792, 190-201, Berlin: Springer.
7. T. Joachims (1997). A probabilistic analysis of the Rocchio algorithm with TFIDF for text categorization. *Proc. of the 14th Int. Conf. on Machine Learning,* 143-151.
8. D. Koller & M. Sahami (1996). Toward optimal feature selection. *Proc. of the 13th Int. Conf. on Machine Learning ICML'96,* 284-292.
9. D.D. Lewis, R.E. Schapire, J.P. Callan, & R. Papka (1996). Training algorithms for linear text classifiers. In H.-P. Frei, D. Harman, P. Schauble, & R. Wilkinson, (ed.), *Proc. of the 19th Annual Int. ACM SIGIR Conf. on Research and Development in Information Retrieval*, 298-306.
10. H. Luhn (1958). The automatic creation of literature abstracts. *IBM Journal of Research and Development,* 2(2):159—165.
11. B. Masand, G. Linoff, & D. Waltz (1992). Classifying new stories using memory based reasoning. *Proc. SIGIR'92,* 59-65.
12. D. Mladenic (1998). Feature subset selection in text-learning. In C. Nédellec, & C. Rouveirol (Eds.), *Machine Learning: ECML-98*, Lecture Notes in Artificial Intelligence, 1398, 95-100, Berlin: Springer.
13. S.K. Murthy, S. Kasif & S. Salzberg (1994). A system for induction of oblique decision trees. *Journal of Artificial Intelligence Research,* 2, 1-32.
14. M. Pazzani & D. Billsus (1997). Learning and revising user profiles: The identification of interesting web sites. *Machine Learning Journal,* 23, 313-331.
15. M. F. Porter (1980). An algorithm for suffix stripping. *Program*, 14(3) : 130-137.
16. G. Salton (1989). *Automatic text processing: The transformation, analysis, and retrieval of information by computer*. Reading, MA: Addison-Wesley.
17. G. Salton & C. Buckley (1988). Term weighting approaches in automatic text retrieval. *Information Processing and Management,* 24(5), 513-523.
18. W.M. Shaw Jr (1995). Term-relevance computations and perfect retrieval performance. *Information Processing & Management*, 31(4), 491-498.
19. Y. Yang & J.O. Pedersen (1997). A Comparative Study on Feature Selection in Text Categorization. *Proc. of the 14th Int. Conf. on Machine Learning ICML-97*, 412-420.
20. G.K. Zipf (1949). *Human Behavior and the Principle of Least Effort*. Reading, MA: Addison-Wesley.