Measuring Contribution of HTML Features in Web Document Clustering

Esteban Meneses
Costa Rica Institute of Technology, Computing Research Center,
Cartago, Costa Rica, P.O.Box 159-7050
esteban.meneses@acm.org

and

Oldemar Rodríguez-Rojas
University of Costa Rica, School of Mathematics,
San José, Costa Rica, P.O.Box 2060
oldemar.rodriguez@predisoft.com

Abstract

Documents in HTML format have many features to analyze, from the terms in special sections to the phrases that appear in the whole document. However, it is important to decide which feature contributes the most to separate documents according to classes. Given this information, it is possible not to include certain feature in the representation for the document, given that it is expensive to compute and doesn't contribute enough in the clustering process. By using a novel representation model and the standard *k-means* algorithm, we discovered that terms in the body of document contributes the most, followed by terms in other sections. Suffix tree provides poor contribution in that scenario, while term order graphs influence a little the partition. We used 4 known datasets to support the conclusions.

Keywords: Web Mining, Symbolic Objects, Web Document Clustering.

Resumen

Los documentos en formato HTML tienen muchas características por analizar, desde los términos en secciones particulares hasta las frases que aparecen en todo el documento. Sin embargo, es importante decidir cuáles son las características que más contribuyen a separar los documentos en ciertas clases. Dada esta información, es posible excluir una característica de la representación de un documento, debido a que es computacionalmente onerosa y no contribuye lo suficiente en el proceso de agrupamiento. Usando una nueva representación y el algoritmo *k-means*, descubrimos que los términos en el cuerpo del documento tienen la mayor contribución, seguido por términos en otras secciones. Los árboles de sufijos proveen una contribución pobre en ese escenario, mientras que los grafos de orden en los términos influencian un poco la partición. Usamos 4 colecciones conocidas para darle soporte a las conclusiones.

Palabras clave: Análisis Web, Objetos Simbólicos, Agrupamiento de Documentos Web.

1 Introduction

The *World Wide Web* is conceived as the biggest information system in the world. It is not just its size what is more impressive, but its rapid growing. This makes web analysis a really hard task. Dealing with thousands of documents for indexing, consulting and clustering is an effort that requires the best approaches to be considered and tested over and over.

One task that has received close attention recently is the named *web clustering* [17, 29]. The whole idea is to group web objects in the natural partitions of population. Applications of that practice are included into indexing, ranking and browsing processes. Part of this job consist in separate web documents from a given collection. Then, similar documents will form families for which several analysis can be done. Although this classification can be manually done as in the ODP [1] or YAHOO! [2] directories, some subtasks are susceptible to be automatized.

Obtaining such groups depend on what representation, distance measure and clustering algorithm is used. This paper discusses a new representation method and how it can be used to determine which HTML feature contributes the most to separate a web document collection. Although several techniques have been proposed for mapping web documents [4, 7, 8, 14, 24, 25, 34], there is an opportunity to include the best of every world and integrate them into a single array: *a symbolic object* [5]. This abstraction consist in an array that can include entries from every data type: real values, intervals, sets, histograms, graphs, trees, and many more. Hence, symbolic objects supersedes the vector model by offering a more general and flexible representation. Each entry is not restricted to be a real value.

By using the well known *k-means* algorithm [12] and the best distances measures for each data type [14, 25, 28], symbolic objects can effectively address the problem of discovering which feature is more important in the clustering process.

The paper is structured in the following way. First, the past and related work is revisited in Section 2. Then, the web document clustering technique is presented in Section 3 as well as the evaluation criteria. Section 4 offers the novel representation for a web document. Symbolic objects and their properties are explained there in first place, leaving for the last subsection the strategy for analyzing contribution in clustering processes. For supporting conclusions, Section 5 presents some results with several datasets. Finally, conclusions and a roadmap for future work is offered in Section 6.

2 Related Work

In his seminal paper [23, 24], Gerard Salton showed a simple, but powerful representation for documents. The basic idea is to depict each document as a real-valued vector in a high dimensional space, where each entry stands for the importance of a given term in the document. Although there are many formulations [3, 4, 8], the fundamental description says that document d_i in any document collection is conceived as $\langle w_{i,1}, w_{i,2}, ..., w_{i,m} \rangle$. The value $w_{i,j}$ is the weight of term t_j for document d_i and m is the size of dictionary (i.e., the number of different terms considered).

One problem arises when it is needed to compute these weights. A typical solution is just count terms up in the document. However, there is a technique for computing weights of terms in documents from a collection. This is called the *TF-IDF* model [8, 32]. The first part stands for *term frequency* while the second for *inverse document frequency*.

The vector space model or the *bag of words* has been, for long time, the classical approach to model web documents. Although some adjustments must be made if HTML tags are considered, the scheme remains basically in the same shape: a real-valued vector [8] or a four tuple of them [13].

On the other hand, Schenker et al [25] explain how a web document can be described by a graph. They claim this approach has the advantage of maintaining the structure of the document, instead of just a counting of terms. Also, basic classification algorithms can be adapted to work with such a data structure [20, 26]. The basic idea is to create as many nodes as terms appear in a dictionary. Then, links between adjacent terms in documents are also mapped into the graph. So, if term t_j appeared just before term t_k in document d_i, then in the graph that symbolizes d_i there must be a link between node t_j and node t_k. Every link will have a tag, regarding which section of the web document the relation comes from. Hence, if relation appears inside a bold tag, the link will have that tag.

In their germinal paper, Zamir and Etzioni [34] presented an innovative method for clustering web documents using a suffix tree. Their method, called STC (Suffix Tree Clustering) was eventually implemented

into a web search engine [35] and called it Grouper. The basic strategy consists in analyzing the phrases that appear into a document. Then all the suffixes of those phrases are used to build a suffix tree. This tree has words as tags in the links, despite the common use of letters in that position. In that tree many characteristics can be saved, as the web document section and the amount of repetitions for that phrase. Nodes appear to be a good place to store such information. Several proposals have been made to extend this basic representation [9, 14].

An important mention must be made about this model. A single suffix tree is usually made for an entire collection in such a way that the tree also stores which document has each phrase. This permits a great efficiency when computing the distance measure [14].

Nevertheless, more information might be used in classification of web documents. Calado et al [7] combined structure and content information. In their approach the link structure of a web collection is included to help in the separation task.

More recently, Meneses and Rodríguez-Rojas [21] proposed the symbolic data approach to model web documents. They build a symbolic vector with several histograms, each standing for a different HTML tag.

Related to determining which HTML tag contribute the most in a clustering process, we must mention Fathi et al [13]. They conducted several experiments on web document classification by means of a extended vectorial representation, where terms in text, anchor, title and metadata sections were considered. They kept 4 vectors, one for each section and run the classification scheme with this representation. On their results, it is clear that terms in metadata contribute more in classification than terms in title.

3 Web Document Clustering

This section starts with a mention of some preprocessing tasks and reviews the basic algorithm for clustering web documents and the evaluation criteria that will measure how effective a given approach is.

3.1 Preprocessing

There are two basic tasks that must be run before clustering is performed: *stopword removal* and *stemming*.

In a text document not every word is as significant as any other. Words which are too frequent among the documents in a collection are not good discriminators. Indeed, a word that appears in 80% of the documents in the collection is useless for retrieval purposes [3]. That kind of words are commonly referred as *stopwords* and they are eliminated for further analysis of the text. Conjunctions, articles, prepositions and connectors are good candidates for conforming a stopword list.

There are many occasions where a user is searching for documents containing certain word. Nevertheless, information retrieval systems can find documents with variants of the word. This is done thanks to a process call *stemming*, which consists in always extracting the root of the word.

Plurals, past tense suffixes and gerund forms are examples of syntactical variations that can prevent a system to find an appropriate document for a user query, that is why a substitution of a word by its stem can be potentially advantageous [3]. A *stem* is the portion of the word that is kept after removing its affixes (suffixes and prefixes). A good example is *connect* which is the stem for a big list of variations: connected, connecting, connection and connections [22]. Stems are conceived to be useful to improve retrieval performance, because they reduce variants of the same root word to a common concept. Moreover, stemming has the benefit to reduce the size of the indexing structure.There are several algorithms, but Porter's [22] is probably one of the most famous.

3.2 Dynamic Clustering Algorithm

Several algorithms have been proposed for clustering and classification of web document collections [6, 9, 15, 16, 18, 31, 33]. However, the most basic and fastest is probably the well known *k-means*[12] or *nuées dynamiques* [11]. The basic idea of this method is to select some special patterns as *centers of gravity* that will attract other patterns and eventually form a cluster. Then, initially, k patterns are randomly selected as centers and iteratively centers will be changing according to the patterns they draw in.

Figure 1 shows the basic steps for *k-means* algorithm to cluster a collection of patters $\mathcal{S}$ into k clusters. It is assumed that we have way to compute the distance between any two elements by means of distance measure function δ. This algorithm has linear complexity time related to the number of patterns n: $O(nk)$.

k-means($\mathcal{S}$,k)

Step 1: Initialization

choose a random partition $\mathcal{P} = (\mathcal{C}_1, \mathcal{C}_2, ..., \mathcal{C}_k)$ or
randomly select k prototypes $\{g_1, g_2, ..., g_k\}$ among element of $\mathcal{S}$
(a step of assignment is required in this last case)

Step 2: Representation

for $i = 1$ to k
  compute the prototype g_i by minimizing the criterion $\sum_{x \in \mathcal{C}_i} \delta(x, g_i)$

Step 3: Assignment

$change \leftarrow$ false
for $i = 1$ to n
  $m \leftarrow$ cluster number to which x_i is assigned to
  assign x_i to $\mathcal{C}_j$, where $j = \min_{l=1,2,...,k} \delta(x_i, g_l)$
  if $m \neq j$ $change \leftarrow$ true, $\mathcal{C}_m \leftarrow \mathcal{C}_m - \{x_i\}$, $\mathcal{C}_j \leftarrow \mathcal{C}_j \cup \{x_i\}$

Step 4: Convergence

if $change$ go to step 2 else STOP

Figure 1: The k-means algorithm.

Extensions have also been made on *k-means* algorithm. In one hand, *bisecting k-means* [27] consists in separate the less dense cluster in each step. *Global k-means* [19] on the other hand is a deterministic algorithm (although costly in computation time) that find a global (not local) optimum for the centers of clusters. Finally, an interesting derivative, *spherical k-means* has been proposed for clustering very large datasets [10].

3.3 Evaluation Criteria

It is important to have a measure for detecting which algorithm or which representation is improving the clustering results. Many of the following measurements assume that a reference collection is present and that a manual clustering can be obtained. For sake of clarity, the manual groups will be called *classes*, while the automatically found will be called *clusters*. The manual classification is denoted by $\mathcal{C}$ and their classes by $\mathcal{C}_i$, while the automatic one is denoted by $\mathcal{P}$ and their clusters by $\mathcal{P}_j$. Also, it will be assumed that n patterns are going to be clustered. The whole idea of evaluation methods is to determine how similar are the clusters to the classes.

3.3.1 Rand Index

The Rand index is computed after examining all pairs of patterns in the dataset passed the clustering. If both patterns are in the same group in the manual classification as well as in the clustering, then a *hit* is counted. If both patterns are in different groups in the manual classification and in the clustering, again a *hit* is counted. Otherwise, no hit is processed. Let's denote by $h(x_i, x_j)$ the function that determine whether is a hit between patterns x_i and x_j. The rand index is just the ratio between the number of hits and the number of pairs:

$$RI = \frac{2 \sum_{i=1}^{n} \sum_{j=i+1}^{n} h(x_i, x_j)}{n(n-1)}$$

3.3.2 Mutual Information

This is an information theory measure that compares the overall degree of agreement between the classification and the clustering with a preference for clusters with high purity (those more homogeneous according to the classification). The higher the value of this index, the better the clustering.

$$MI = \sum_{i=1}^{k} \sum_{j=1}^{k} \frac{|\mathcal{C}_i \cap \mathcal{P}_j| \log(n|\mathcal{C}_i \cap \mathcal{P}_j|)}{nk^2 |\mathcal{C}_i||\mathcal{P}_j|}$$

3.3.3 F-Measure

This measure combines the ideas of *recall* and *precision* from the Information Retrieval literature [27]. The precision and recall of cluster j with respect to class i are defined as:

$$P = Precision(i, j) = \frac{N_{i,j}}{N_j}$$

$$R = Recall(i, j) = \frac{N_{i,j}}{N_i}$$

where $N_{i,j}$ is the number of members of class i in cluster j, N_j is the number of members of cluster j and N_i is the number of members of class i.

Finally, the F-measure of class i with respect to cluster j is:

$$F(i, j) = \frac{2PR}{P + R}$$

Then, for each class it is selected the cluster with the highest F-measure to be the cluster that represents that class and its F-measure becomes the F-measure of the class. The overall F-measure for the clustering result $\mathcal{P}$ is the weighted average of the F-measure for each class:

$$F_{\mathcal{P}} = \frac{\sum_{i=1}^{k} |\mathcal{C}_i| F_i}{n}$$

3.3.4 Entropy

This measure provides a good way to determine how good partition has been without dealing with nested clusters, analyzing one level in the hierarchical clustering. The output determine how homogeneous a cluster is. The higher the entropy, the lower the homogeneity of cluster. The entropy of a cluster that only contains one object is zero.

To compute the entropy, it is needed to calculate the probability $p_{i,j}$ which is the odds that a member of cluster j belongs to class i. After that, the standard formula is applied $E_j = -\sum_{i=1}^{k} p_{i,j} \log(p_{i,j})$ and the sum is taken over all classes. The total entropy for a clustering is calculated as the sum of entropies of each cluster weighted by the size of that cluster:

$$E_{\mathcal{P}} = \sum_{j=1}^{k} \left(\frac{|\mathcal{P}_j|}{n} + E_j \right)$$

4 Symbolic Representations

This section deals with the definition of a new representation for web documents. First, symbolic data will be presented and then the proper model will be explained. Finally, an interesting property of symbolic data will be offered.

4.1 Symbolic Objects

Traditionally, real-valued vectors have been used to model web documents. If n documents are evaluated by m variables, then a $n \times m$ matrix will hold all the relationships between them. However, the real world is too complex to be described in this relatively simple tabular model [5]. In order to deal with more complex cases we use symbolic data. In this context, types are not confined to be real values, but can be selected from a huge list: sets, intervals, histograms, trees, graphs, functions, fuzzy data, etc.

A *symbolic object* is a vector where each entry has a symbolic data type from the ones described above. Symbolic objects can better at representing the *variability* associated with a real life object.

Each symbolic object is implemented by a symbolic data array. This structure contains all the sensed information for some real life object. The general model admits even different types for every variable. Nevertheless, the same data type can be used for all variables as this type better encapsulates all the

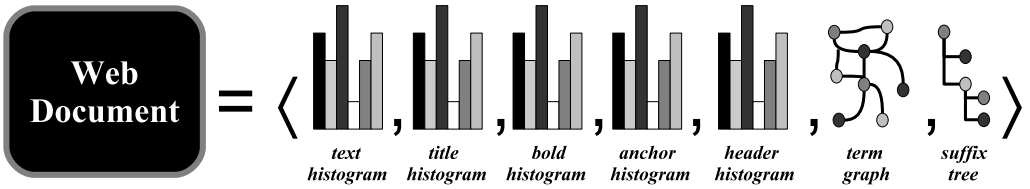

Figure 2: A symbolic representation for a web document.

variability of the object. This model can be applied to web document representation by considering the HTML tags.

Symbolic objects are better at representing concepts rather than individuals. That means, a concept is made by grouping characteristics of diverse patterns that form a single concept. For example, a web site can be thought as a concept that mixes the content of its web pages. Web documents can be conceived as a concept that represent the content of the several HTML code sections that form it.

4.2 Representing a Web Document

The main idea of this paper is to provide a novel framework for representing web documents and web document collections by means of symbolic objects. Such representation must provide an insight on which variable from the complex symbolic representation is the most contributive to cluster the original collection. As described above, symbolic data has a great potential as they can model the variability associated with some concept. Instead of using real-valued objects, symbolic arrays can replace classical vectors with much representation power.

As discussed in section 2, some models have been proposed to overcome the limitations of real-valued vectors. These frameworks can be joined in a single representation. Symbolic objects can aggregate scalars, histograms, graphs, trees and many data types more.

One first option is to use histograms to count words according to which HTML tag they appear into. Given that the original vectorial model is unable to separate terms in different tags (other than multiply an apparition by some constant factor given the HTML tag [8]), source information is lost as it is impossible to distinguish when a term appear several times in one tag or many others in diverse tags. This is true unless extra information is conserved about the source of terms. For example, in [13] several vectors of terms are maintained. However, using histograms is an equivalent approach and terms can be still be separated and a formula for aggregate them is also provided.

A *histogram* is a data type that forms a distribution for given categories. A probability distribution can be modeled by means of this type. A histogram contains p categories where its value can be stored. More formally, each document d in the collection $\mathcal{D}$ is represented by the symbolic object x_d in m histogram dimensions $\{x_{d1}, x_{d2}, ..., x_{dm}\}$. Each variable x_{di} is a normalized histogram $\{x_{di1}, x_{di2}, ..., x_{dip}\}$ with p *categories* or *modalities*.

Nevertheless, the basic model can be extended to include more data types and so more information about the document. Several data types can be added to the initial description. Figure 2 presents the basic representation for a web document using histograms and symbolic data arrays. In this case, only 5 tags have been considering: text, title, bold, anchor and header. Extending the model to include more tags is straightforward. By considering the term graph and the suffix tree, figure 2 presents the final representation for a web document.

4.3 Distance Measures

There must be a formula to compute how distant is one symbolic variable from other. For measuring distances between two histograms h_x and h_y, the extended Jaccard index [28] has been adapted:

$$\delta_{JAC}(h_x, h_y) = 1 - \frac{h_x \cdot h_y}{||h_x||^2 + ||h_y||^2 - h_x \cdot h_y}$$

where $||h_x||$ is the magnitude of histogram h_x if conceived as a vector.

For graph data, the distance proposed by [25] is based on cosine measure. Let g_x and g_y be two graph representations:

$$\delta_{GRA}(g_x, g_y) = 1 - \frac{|mcs(g_x, g_y)|}{max(|g_x|, |g_y|)}$$

Here, the *mcs* function stands for maximum common subgraph and $|x|$ is the size of the graph (i.e., the number of nodes and edges in the graph).

A distance measure for suffix trees, based on the one proposed by [14] is the following:

$$\delta_{TRE}(t_x, t_y) = 1 - \frac{\sqrt{\sum_{i=1}^{r} [l_i g(l_i)(f_{x,i} w_{x,i} + f_{y,i} w_{y,i})]^2}}{\sum_{j=1}^{|d_x|} |s_{x,j}| w_{x,j} + \sum_{k=1}^{|d_y|} |s_{y,j}| w_{y,j}}$$

where r is the number of matching phrases between trees t_x and t_y. Length of each matching phrase is denoted by l_i and function g measures how much of the original phrase was matched:

$$g(l_i) = \left(\frac{l_i}{max(|s_{x,i}|, |s_{y,i}|)} \right)^{\gamma}$$

and γ is a parameter for balancing the function. As [14] this parameter is equal to 1.2. Finally, $f_{x,i}$ is the frequency of phrase i in document d_x and $w_{x,i}$ is the weight of this phrase according to the HTML tag where it appears. The weights typically follow three levels: LOW (plain text), MEDIUM (header, bold and anchor tags) and HIGH (title tag). The values for the weights can be set to 1,2 and 3, respectively.

Finally, all distances from variables must be aggregated into the following formula:

$$\delta(d_x, d_y) = \sum_{i=1}^{m} w_i * \delta_i(d_x[i], d_y[i])$$

where d_x and d_y are two symbolic representations for web documents, $d_x[i]$ is the i-th symbolic variable that forms d_x and δ_i is the respective distance measure.

4.4 Measuring Contribution

One important question arises when each dimension is analyzed for its contribution to partition. For example, given a document collection $\mathcal{D}$ and a symbolic representation that comprises several data types: What is more important for the clustering results? Will it be the title terms histogram? Will it be the suffix tree?

The answer to this questions remains inside the properties of document collections. Some collections can be more prone to be analyzed by its phrase structure, others by their bold tag terms, and so on.

In their paper [30], Verde et al make an analysis of what is more important to clustering results. Their formulas permit to obtain which dimension is more relevant for clustering a given collection.

By using the *k-means* algorithm, if we need to cluster a data collection $\mathcal{D}$, then we will obtain a partition $\mathcal{P} = \{\mathcal{P}_1, \mathcal{P}_2, ..., \mathcal{P}_k\}$, whose gravity centers will be g_1,g_2,...,g_k, respectively. Consider the following definition as the variability of cluster $\mathcal{P}_i$ with respect to x belonging to the space of description:

$$S(\mathcal{P}_i, x) = \sum_{j=1}^{m} S_j(\mathcal{P}_i, x) = \sum_{j=1}^{m} \sum_{x_s \in \mathcal{P}_i} \delta(x_s[j], g_i[j])$$

Then, the contribution of variable j to the partition $\mathcal{P}$ allows to evaluate the role of this variable to the building of such partition and is defined:

$$K_j(\mathcal{P}) = \frac{\sum_{i=1}^{k} S_j(\mathcal{P}_i, g_i[j])}{\Delta(\mathcal{P}, g)}$$

where function Δ is the criterion for the quality measure and is computed:

$$\Delta(\mathcal{P}, g) = \sum_{i=1}^{k} S(\mathcal{P}_i, g_i)$$

5 Experiments

All datasets will be described at the beginning of this section. In the second part, results for different representations will be provided. The experiments were run several times and average data are shown.

5.1 Web Document Collections

Four different datasets were used for determining the contribution of different HTML features. Table 1 present a summary of datasets properties. In all cases, there is a manual classification for comparing the resulting clustering.

The first one is the *Webdata* dataset [14] and contains 314 documents from 10 categories. This web pages were taken from a university web site in Canada. The contents span from home pages to documents about sport activities. The second one is a subset of the ODP web directory [1]. A total of 1495 web pages were crawled from 5 categories: arts, business, health, science and sport. The third and fourth are datasets taken from the WebKB project. The third one is known as *WebKB* and consists of 4 classes and 1915 documents. The last dataset is a subset of the *20Newsgroup*. It consists of 2000 documents from 20 categories and correspond to messages into a newsgroup.

Dataset name	**Number of files**	**Number of classes**	**File size average (Kb)**	**Dictionary size**
Webdata	314	10	13.12	11734
ODP	1495	5	22.20	43639
WebKB	1915	4	5.15	27480
20Newsgroup	2000	20	2.60	35459

Table 1: Reference collections features.

5.2 Results

Two strategies were employed for obtaining the resulting contributions. In all cases a symbolic representation (as in section 4) was used with the following properties. All five sections (text, title, bold, anchor and header) were represented with a histogram of 200 categories. Then, a term graph was build with the top 50 terms, which implies the graph had 50 nodes. Finally, the suffix tree was constructed using the first 30 sentences for each analyzed section.

The first method is the typical *one feature per time* methodology, which is ideal for non symbolic data. In this scenario, the dataset was clustered using only one feature every time. The experiment was repeated 20 times and average values were computed. Figure 3 shows the results for the 4 datasets. All features were analyzed in every case, but certain features doesn't provide a convergent clustering (according to *k-means* algorithm in section 3). For example, in Webdata and ODP datasets, the feature *header* doesn't provide a convergent clustering. Such features were eliminated from the figure.

According to evaluation measures, in figure 3, tha Webdata dataset is better clustered if plain text is used. The second place stands for the graph representation and the worst performance was for the suffix tree. In the ODP dataset, the text and the title obtained the two first places, while the last was the suffix tree. The WebKB dataset showed that the text and header terms are very valuable for improving the clustering. Again, the suffix tree was relegated to the last place. In the 20Newsgroup dataset only 3 features were measured, because documents don't contain any HTML tag. The ranking of features was text, graph and tree, in that order.

As we mentioned before, one important discovery is to determine how relevant is some feature to the given clustering results. Figure 4 shows the results for contribution of every HTML features in the different datasets. The same symbolic representation was used and the experiments were repeated 20 times each.

Figure 4 demonstrated that in Webdata dataset text terms (those that appear in none special tag) are the most predominant factor in clustering, followed by title and anchor terms. The header terms are the least relevant for the clustering results. The graph obtained almost 10% of contribution, while suffix tree has very low participation.

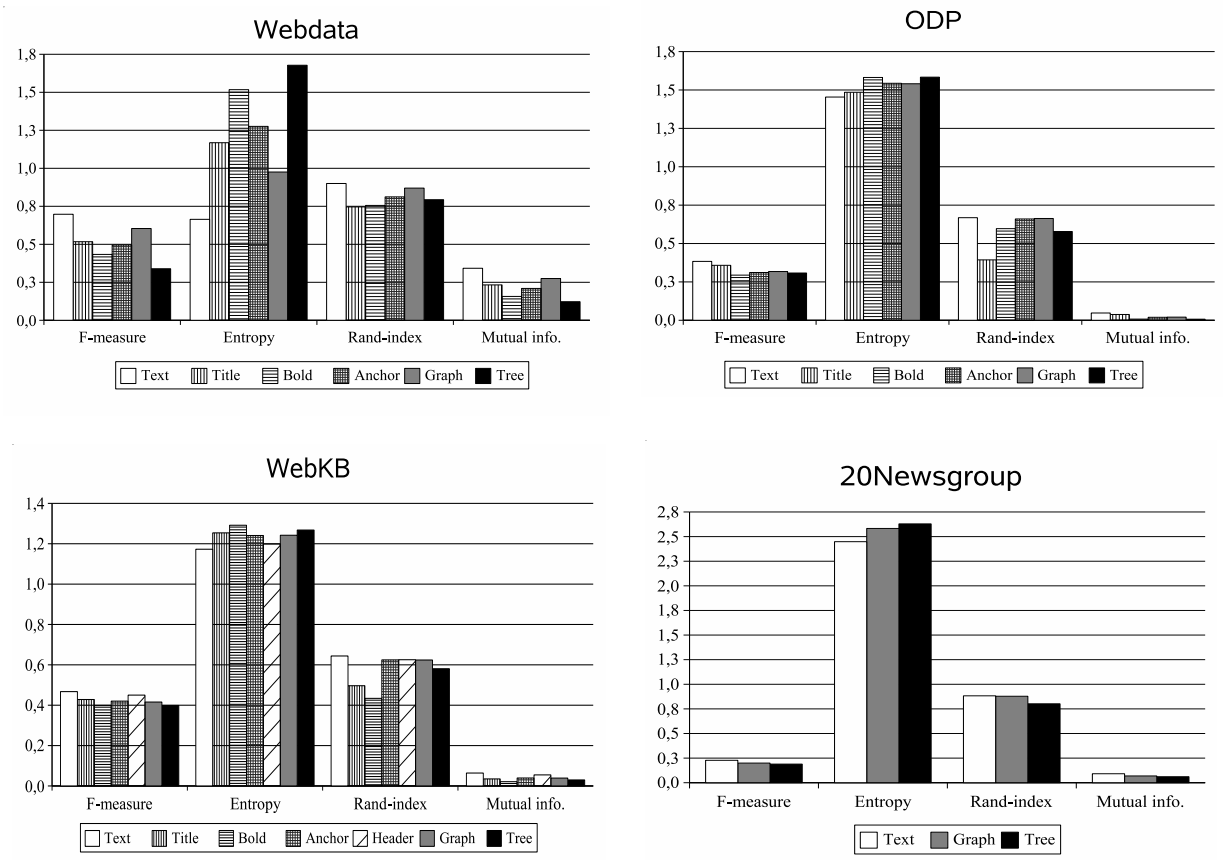

Figure 3: Evaluation of separated HTML features for different datasets.

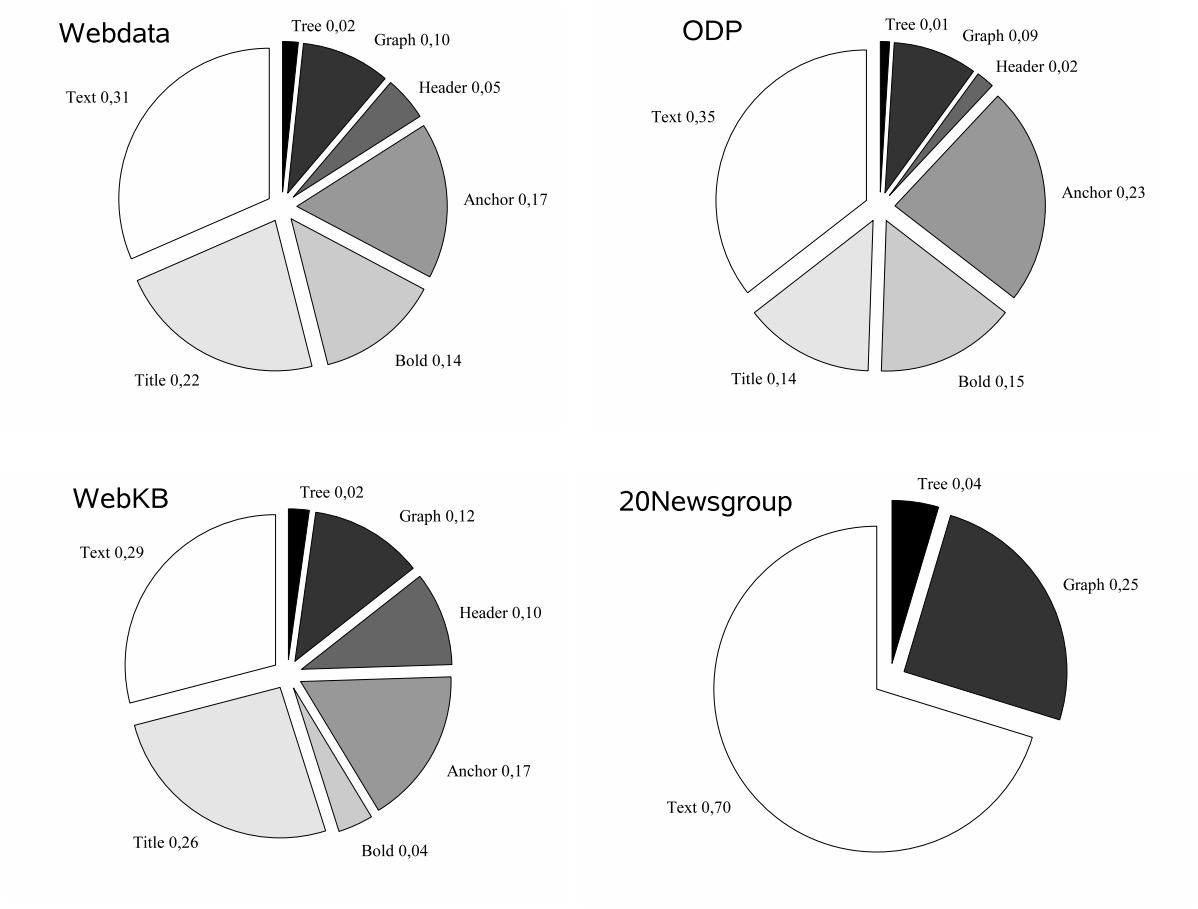

Figure 4: Contribution of HTML features for different datasets.

On the other hand, ODP dataset shows a slightly different story about contribution in figure 4. Text terms are again the most important factor, but anchor terms get the second place, followed by bold and title terms. Header terms are the least relevant for the resulting partition. The contribution of graphs is below 10%, while suffix tree contribution continues to be poor.

In the case of WebKB dataset, figure 4 offers a little variation according to the ranking of contribution. The first place is for text term, followed by title terms. The least contributing terms are those in bold tags. The graph and suffix tree have similar contribution as in the other cases.

One extreme case appears when considering newsgroup dataset, given that tag information is poor or nonexistent. Figure 4 presents a case where information appears only in the plain text content of the document. One more time, text is the most important factor, followed by graph and the tree is in the last position.

6 Conclusions and Future Work

Symbolic objects are a new approach that permits to include more information about a web document. It is a flexible representation, were data is not restricted to be real-valued. Instead, many data types can be used: intervals, sets, histograms, graphs, trees, you name it.

The main contribution of this paper is to present a new representation model for web documents and how it can help to determine which feature is more important in the clustering task. The results showed that text terms in the body of the document is the most contributing factor, followed by title and anchor terms. The suffix tree presented a poor contribution, while the order term graph offered a little help in that regard.

Nevertheless, there is a lot of work to be done in this area. It would be interesting to explore new data

types: sets, intervals, layouts, and some others, and to determine if that data type contribute in a significant way to cluster document collections. These structures can address different problems when included into a symbolic representation. Besides, there is a tendency in including structural information. This is obtained after analyzing the hyperlink relationships among web documents. There is a first proposal [7] on how to integrate these two dimensions: content and structure. Symbolic objects could include such data to improve the results.

Finally, an important feature about symbolic objects is their propensity to visualize information. As complex information repositories, symbolic representation offers new possibilities to visualize relationships. It would be ideal to develop techniques for exploit the information from symbolic object to obtain graphical impressions of web document collections.

Acknowledgements

This work is part of the *Klá* research project, which is under development at the Computing Research Center (located at the Costa Rica Institute of Technology).

References

[1] Odp: Open directory project. http://dmoz.org. Visited on July 15th, 2007.

[2] Yahoo! search directory. http://dir.yahoo.com. Visited on July 15th, 2007.

[3] Baeza-Yates, R., and Ribeiro-Neto, B. *Modern Information Retrieval*. Addison Wesley - ACM Press, 1999.

[4] Baldi, P., Frasconi, P., and Smyth, P. *Modeling the Internet and the Web: Probabilistic Methods and Algorithms*. John Wiley & Sons, 2003.

[5] Bock, H.-H., and Diday, E. *Analysis of Symbolic Data*. Springer-Verlag, 2000.

[6] Boley, D., Gini, M., Gross, R., Han, E.-H., Hastings, K., Karypis, G., Kumar, V., Mobasher, B., and Moore, J. Partitioning-based clustering for web document categorization. *Decision Support Systems* (1999).

[7] Calado, P., Cristo, M., Moura, E., Ziviani, N., Ribeiro-Neto, B., and Gonalves, M. A. Combining link-based and content-based methods for web document classification. *Conference on Information and Knowledge Management* (2003).

[8] Chakrabarti, S. *Mining the Web*. Morgan Kaufmann Publishers, 2003.

[9] Crabtree, D., Gao, X., and Andreae, P. Improving web clustering by cluster selection. *International Conference on Web Intelligence* (2005).

[10] Dhillon, I. S., Fan, J., and Guan, Y. Efficient clustering of very large document collections. *Data Mining for Scientific and Engineering Applications* (2001).

[11] Diday, E., and Simon, J. Cluster analysis. *Digital Pattern* (1976).

[12] Duda, R., and Hart, P. *Pattern Classification and Scene Analysis*. New York: Wiley, 1973.

[13] Fathi, M., Adly, N., and Nagi, M. Web documents classification using text, anchor, title and metadata information. *The international conference on Computer Science, Software Engineering, Information Technology, e-Business and Applications* (2004).

[14] Hammouda, K. M., and Kamel, M. S. Efficient phrase-based document indexing for web document clustering. *IEEE Transactions on Knowledge and Data Engineering* (2004).

[15] Haruechaiyasak, C., Shyu, M.-L., Chen, S.-C., and Li, X. Web document classification based on fuzzy association. *Proceedings of the 26th International Computer Software and Applications Conference on Prolonging Software Life: Development and Redevelopment* (2002).

[16] Hu, W.-C., Chang, K.-H., and Ritter, G. X. Web document classification using modified decision trees. *ACM Southeast Regional Conference* (1999).

[17] Kobayashi, M., and Takeda, K. Information retrieval on the web. *ACM Computing Surveys* (2000).

[18] Kwon, O.-W., and Lee, J.-H. Web page classification based on k-nearest neighbor approach. *International Workshop on Information Retrieval with Asia Languages* (2000).

[19] Likas, A., Vlassis, N., and Verbeek, J. J. The global k-means algorithm. *Pattern Recognition 36*, 1 (2003).

[20] Markov, A., Last, M., and Kandel, A. Model-based classification of web documents represented by graphs. *WebKDD: Workshop on Web Mining and Web Usage Analysis* (2006).

[21] Meneses, E., and Rodríguez-Rojas, O. Using symbolic objects to cluster web documents. *15th World Wide Web Conference* (2006).

[22] Porter, M. F. An algorithm for suffix stripping. *Program, 14(3) pp 130-137* (1980).

[23] Salton, G., and Lesk, M. The smart automatic document retrieval system : An illustration. *Communications of the ACM* (1965).

[24] Salton, G., Wong, A., and Yang, C. A vector space model for automatic indexing. *Communications of the ACM* (1975).

[25] Schenker, A., Last, M., Bunke, H., and Kandel, A. Classification of web documents using a graph model. *Seventh International Conference on Document Analysis and Recognition* (2003).

[26] Schenker, A., Last, M., Bunke, H., and Kandel, A. A comparison of two novel algorithms for clustering web documents. *2nd IWWDA* (2003).

[27] Steinbach, M., Karypis, G., and Kumar, V. A comparison of document clustering techniques. *University of Minnesota* (2000).

[28] Strehl, A., Ghosh, J., and Mooney, R. Impact of similarity measures on web-page clustering. *AAAI-2000: Workshop of Artificial Intelligence for Web Search* (2000).

[29] Vakali, A., Pokorny, J., and Dalamagas, T. An overview of web data clustering practices. *EDBT 2004Workshops* (2004).

[30] Verde, R., Lechevallier, Y., and Chavent, M. Symbolic clustering interpretation and visualization. *Journal of Symbolic Data Analysis 1*, 1 (2003).

[31] Wang, Y., Hodges, J., and Tang, B. Classification of web documents using a naive bayes method. *Proceedings of the 15th IEEE International Conference on Tools with Artificial Intelligence* (2003).

[32] Yu, C., Lam, K., and Salton, G. Term weighting in information retrieval using the term precision model. *Journal of the ACM* (1982).

[33] Yu, H., and Han, J. Pebl: Positive example based learning for web page classification using svm. *SIGKDD* (2002).

[34] Zamir, O., and Etzioni, O. Web document clustering: A feasibility demonstration. *SIGIR* (1998).

[35] Zamir, O., and Etzioni, O. Grouper: A dynamic clustering interface to web search results. *Computer Networks* (1999).