Automatic Logo and Trademark Extraction from Large Corporate Web Sites

Baratis Evdoxios
Dissertation Thesis

Technical University of Crete (TUC)
Department of Electronics and Computer Engineering

Chania 2005

Table of Contents

Abstract

As the size and diversity of the World Wide Web (WWW) grows rapidly, Web sites become bigger and more complicated in content and structure and it is becoming more and more difficult to skim over their contents. This work is directed towards Web site summarization by image content focusing on the extraction of logo and trademarks from large corporate Web sites. This task is complementary to text summarization methods but, as opposed to methods that are based on text, the proposed method is based on image feature extraction from images and machine learning for distinguishing logo and trademarks from images of other categories (e.g., landscapes, faces). Because the same logo or trademark may appear many times in various forms within the same Web site, unique logo and trademark images are extracted first. These images are then ranked by importance. The most important Logos and Trademarks are finally selected to form the image summary of a Web site. The evaluation of the method demonstrated very promising performance.

1. Introduction

1.1. Motivation

As the size and diversity of the World Wide Web grows rapidly leading to information overload it is becoming more and more difficult for a user to skim over a web site and to get an idea of its contents. A solution to this problem is web site summarization [1]. Our objective is summarization by image content. The combination of text and image summarization could lead to more complete web site summaries and effective web browsing and retrieval by extending web site text results with images.

1.2. Problem being solved: web site image summarization

A site contains images of various types: logos, trademarks, portraits, landscapes etc. Each image type has different characteristics. Our objective is to extract Logos and Trademarks by finding those characteristics that discriminate them from the other types of images. The goal is to find the most important Logos and Trademarks of a web site.

1.3. Methodology: Main Idea

The process of finding the most important Logos and Trademarks of a web site is divided into three steps.

Training to learn how to extract Logos and Trademarks images: This step is based on image feature extraction. These features describe the main characteristics of Logos and Trademarks. Machine learning is used to discriminate between Logos and Trademarks and images of other categories such as person images, outdoor images, images of products etc.

Clustering of similar images: There are cases where the same logo/trademark is used (displayed, pointed to by pages) more than once in a Web site. The same image may also appear with different size, with the same or different color, or even as grey scale image. Variances in the spatial properties of images corresponding to the same logo are also common. Once all images have been extracted from a web page and the logo/trademarks have been detected (step 1), identical or similar images are grouped together into clusters. This step is also based on feature extraction and machine learning as the above.

Image ranking: The final step includes the selection of the most important – characteristic images from a Web site. This stage accepts the results of the previous stage and ranks images by importance. An image of each group is selected to represent the set of images with in the same group. The top **k** (k is the number of clusters) images are selected as the most characteristic logos/trademarks of the web site.

1.4. Structure of Thesis

Logo and Trademark detection is discussed in *Section 2*. It describes the characteristics of Logos and Trademarks and introduces the features, which take advantage of these characteristics. Machine learning by decision trees as well as methods for training a decision tree for detecting Logo and Trademark images are also discussed.

Section 3 describes the step for Logo and Trademark similarity. First it introduces the similarity criteria (e.g. histograms intersections and invariant moments) and then describes the basic theory of these features, decision trees, image distance functions and finally the different methods for image clustering are also discussed.

Section 4 refers to the ranking Logo and Trademark. It introduces the main idea for image ranking, as well as the methods for judging image importance and selecting the most important images from large web sites.

Section 5 presents experimental results. It demonstrates the step-by-step operation of the method and includes experiments based on different web sites.

Section 6 summarizes the method and discusses issues for future work.

2. Logo and Trademark Detection

A web site contains images of different types (e.g., landscapes, person images). Our objective is to discriminate Logos and Trademarks from the other images. This discrimination is based on image features.

2.1. Image Content Descriptions

In general a logo or a trademark is a small graphic image, with no too many intensity levels and colors. Besides, logo and trademarks exhibit a lower spatial distribution of intensities and colors than images of other categories. Our goal is to extract and use image features that will take advantage of these peculiarities. First, we introduce the row (i.e., intensity and frequency) histograms and then a set of features that are defined on histograms.

2.1.1. Row Histograms

An "Intensity **Histogram"**, represents the distribution of image intensities within the image itself. It is a plot showing the number of pixels for each intensity value [2]. In this work, all images are reduced (and so their respective intensity histograms) to 256 discrete intensity levels (256 histogram bins). Also, because the same logo or trademark image may appear as color or grey scale image within the same Web site, color information is not useful in content representations. In the following, all images are converted to grey scale. Color images are converted to grey scale according to the formula

$$I = 0.299R + 0587G + 0.114B$$ [3]

Image 2.2 is a logo while image 2.1 is a non-logo. Figures 2.1 and 2.2 illustrate the two intensity histograms for these images. The histograms are computed on their respective grey scale images.

Image 2.1: non-logo image

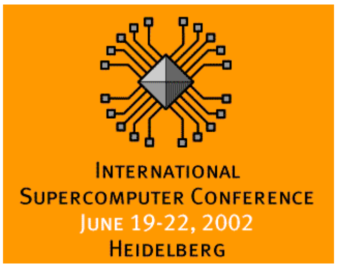


Image 2.2: logo image

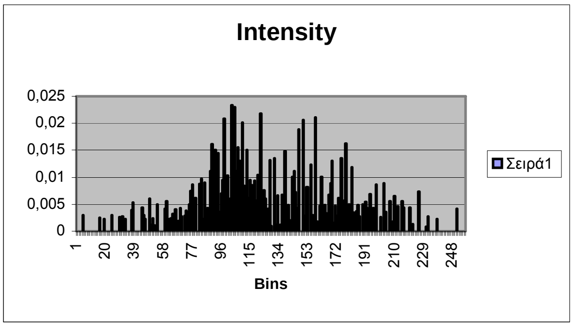


Figure 2.1: Intensity histogram of image 2.1.

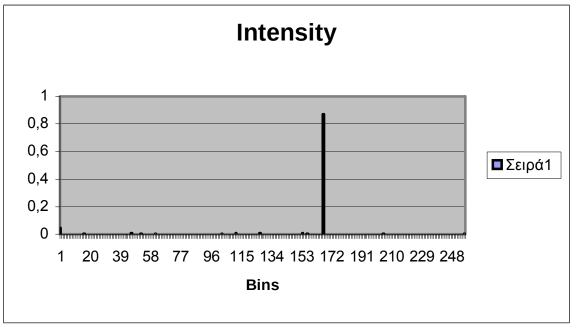


Figure 2.2: Intensity histogram of image 2.2. Image 2.2 uses fewer colors. Also these colors are less distributed than image 2.1.

The intensity histogram of image 2.1 has more occupied bins than that of the logo image. Notice also that the logo images exhibit a lower spatial distribution of intensity levels than non-logo images, with the majority of pixels concentrated at bin 165. It should be mentioned that histograms are normalized: each histogram bin is divided by the sum of image pixels (so that the value at each histogram bin also represents the probability of occurrence of the corresponding intensity level in the image). Therefore, the summation of all histograms bins equals 1.

In the following, two additional histograms defined on image frequency information are also introduced in this work for both describing image content and for discriminating between logo and non-logo images. For frequency domain analysis we used the Fourier transformation [4]. An efficient implementation of this transform by Cooley and Tukey, [4] is the fast Fourier transform (FFT). FFT requires that image horizontal and vertical sizes to be a power of 2 which is not always true for random images. We have chosen to extend images up, to the list integer which is greater than the image dimensions, by padding pixels: Pixels outside image borders at position x, y take values x mod M and y mod N where M, N are the desired horizontal and vertical image size in pixels (which are power of 2). Alternatively, the size can be reduced to the lower power of 2. This results in faster analysis but portions of image (and consequently image content information) are lost. The advantage of padding is that the padded image retains the content of the original image.

Image 2.3 illustrates the radial frequency spectrum of image 2.2 with rings of radius r. The radial frequency spectrum measures the frequency power spectrum within each ring between radius r and radius r + dr. Two methods were examined for dr definition. The first requires constant area increment of each circle, while the second requires constant radius increment. Both split spectrum into 256 circles. The increment dr for these two methods is defined as:

Method 1: Constant area increment

1st circle radius: $r_1 = \sqrt{\frac{\pi R^2}{\pi 256}} = \frac{R}{16}$

Where R the last's (largest) circle radius.

Each circle radius: $r_i = \frac{R}{16}\sqrt{i}$

Where i the enumerator of each circle (1-256).

dr definition: $dr = r_i - r_{i-1}$

Method 2: Constant radius increment

1st circle radius: $r_1 = \frac{R}{256}$

Where R the last's (largest) circle radius.

Each circle radius: $r_i = \frac{R}{256} i$

Where i the enumerator of each circle (1-256).

dr definition: $dr = r_i - r_{i-1}$

The method we finally used is the second. The reason is that the differences are mainly at low and mid frequencies (figures 2.3 and 2.4) and the first circles of the *constant area increment* method are too large to capture these differences.

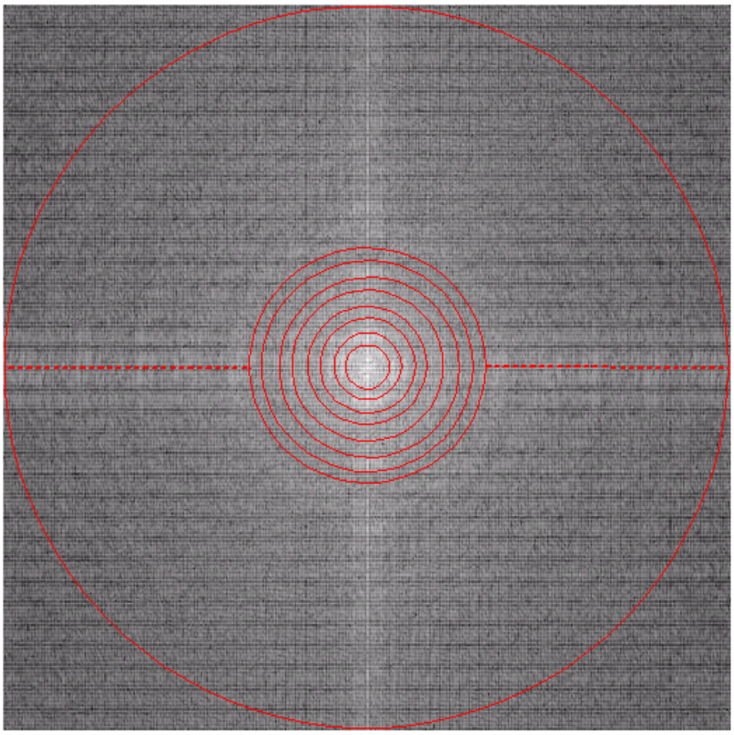


Image 2.3: Radial Frequency Spectrum of image 2.2

The first of the two frequency histograms is the **Radial Frequency Histogram**. It displays the power spectrum from low frequencies at the left of the plot (corresponding to smaller rings closer to the center of the frequency spectrum) to high frequencies at the right (corresponding bigger outer rings). Figures 2.3 and 2.4 illustrate the two radial frequency histograms for images 2.1 and 2.2 respectively.

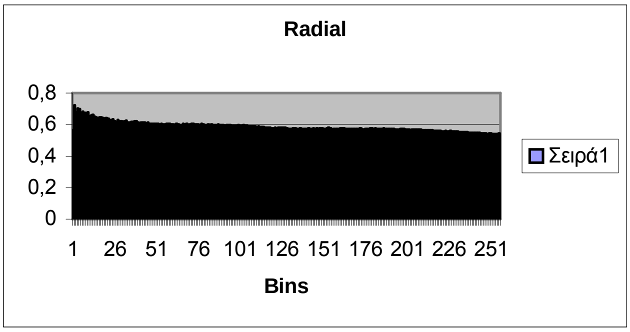


Figure 2.3: Radial frequency histogram of image 2.1.

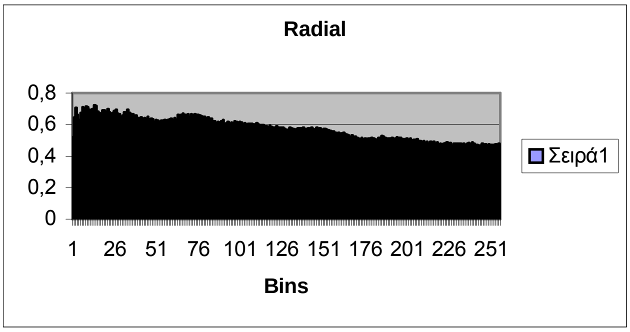


Figure 2.4: Radial frequency histogram of image 2.2.

The difference of these two histograms is mainly at low and mid frequencies. The radial frequency histogram of image 2.1 has more pixels of low frequencies (1-10 bins) while the radial frequency histogram of logo image has more pixels of mid frequencies (40-80 bins). Logo (graphic images) exhibit more intent (sharp) color changes pushing the power spectrum towards higher frequencies. “Natural” images (landscapes or person images) usually exhibit more temperate chromatic changes pushing the power spectrum towards lower frequencies. Histogram is normalized. Each histogram bin represents a ring and is divided by the number of pixels in the ring. Therefore it represents ring mean value and varies between 0 and 1.

Image 2.4 illustrates the angle frequency spectrum of image 2.2 with regions taken every θ angle. The angle frequency spectrum measures the frequency power spectrum between angle θ and angle $\theta + d\,\theta$. The frequency spectrum is splitted into 256 regions defined every $d\,\theta$ intervals.

dθ definition

1st angle: $\theta_1 = \frac{180^o}{256}$

step θ : $\theta_i = \frac{180^o}{256} i$

$$1 \le i \le 256$$

$$d\theta = \theta_i - \theta_{i-1}$$

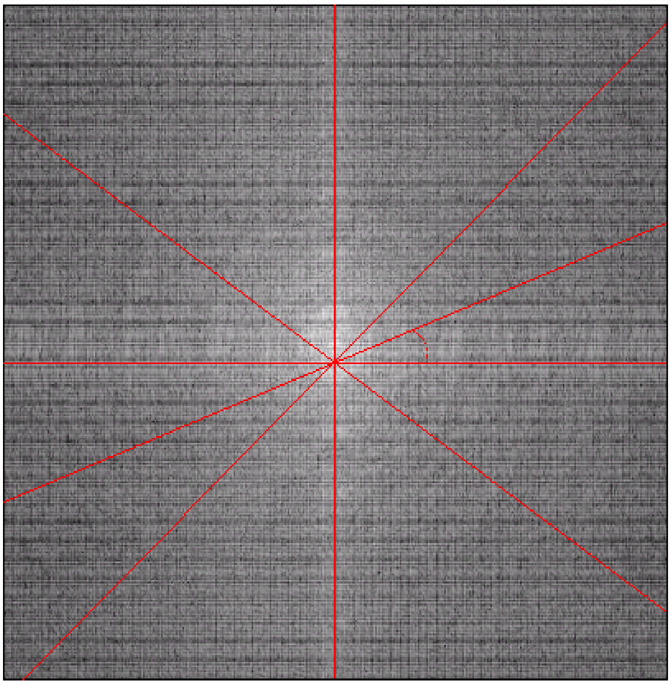


Image 2.4: Angle Frequency Spectrum of image 2.2

A second type of histogram defined over frequency information is the **Angle Frequency Histogram**. It displays power spectrum as a function of azimuthal angle from 0^o at the middle of the plot to $+180^o$ at the right and -180^o at the left. Figures 2.5 and 2.6 display the two angle frequency histograms for images 2.1 and 2.2 respectively.

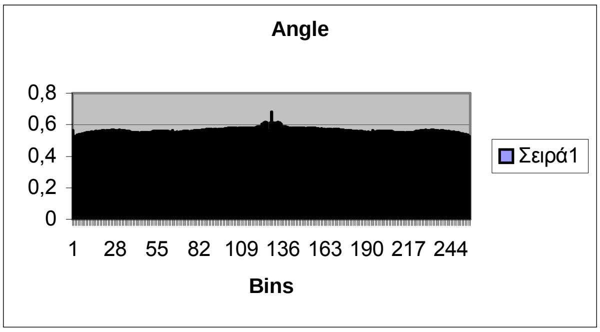


Figure 2.5: Angle frequency histogram of image 2.1.

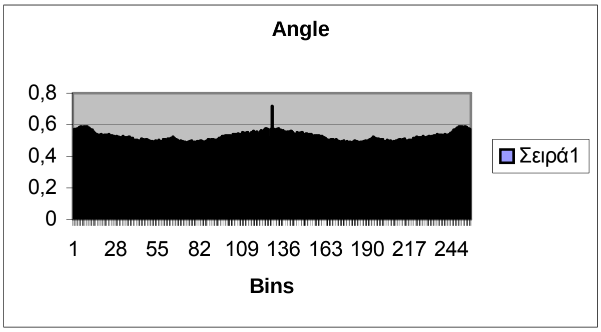


Figure 2.6: Angle frequency histogram of image 2.2.

The angle frequency histogram of image 2.1 exhibits little variation while the angle frequency histogram of the logo image has three picks at -180^0, 0^0 and 180^0 indicating the existence of an object. Histogram is normalized. Each histogram bin represents a region and is divided by the number of pixels in the region. Therefore it represents portion's mean value and varies between 0 and 1.

A fourth histogram can also be defined as the union of the three histograms, leading to a histogram with $3 \times 256 = 768$ bins.

2.1.2. Histogram Features

Histograms features are properties measured on histograms. In the following, 7 such features are defined on histograms namely: **mean value**, **threshold**, **standard deviation**, **third order central moment**, **fourth order central moment**, **energy and entropy**. The same features are computed on intensity, angle and radial histogram making up a vector of 21 features. An additional (22nd) feature, namely number of **occupied bins** is computed only on intensity histogram. The 23rd feature is **image size**.

Mean value: For intensity histogram it demonstrates the luminance of an image and ranges between 0 and 255. Besides intensity histograms of logo-trademark images tend to have greater mean values than the other types of images because of their white background.

Threshold: Thresholding is a binarization method [5]. The **Otsu Threshold**, also called discriminant method, is a method for optimal thresholding. It returns an integer value between 0 and 255, separating the image pixels into to classes. Like mean value, Otsu Threshold demonstrates the luminance of an image and is greater for logo-trademark images.

Standard deviation characterizes a distribution's "variability" around its central value [6] and is computed as

$$\text{Standard Deviation} = \sqrt{\sum_i (i-m)^2 \times prob[i]}$$

where the summation is taken over all the pixel intensities (256 values), i is the pixel intensity, m is the mean value and prob[i] is the probability of a pixel with intensity i.

Third order central moment, or "skewness" [6], characterizes the degree of asymmetry of a distribution around its mean. It is a nondimensional quantity that characterizes the shape of the distribution. A positive value of skewness indicates a distribution with an asymmetric tail extending out rightward while a negative value signifies a distribution with an asymmetric tail extending out leftward**.** The third order central moment is computed as

$$\text{Skewness} = \sum_i \left(\frac{(i-m)}{\sigma} \right)^3 \times prob[i]$$

where σ is the standard deviation of the distribution.

Fourth order central moment, or **"**kurtosis" [6]. Similarity to skewness, kyrtosis is also a nondimensional quantity. It measures the relative (to normal distribution), peakedness or flatness of a distribution. A distribution with negative kurtosis is called platykurtic, while a distribution with positive kurtosis is termed leptokurtic. Kyrtosis is computated as

$$\text{Kyrtosis} = \left(\sum_i \left(\frac{(i-m)}{\sigma} \right)^4 \times prob[i] \right) - 3$$

Energy, or angular second moment [7], measures the homogeneity of an image. The more homogeneous an image is the higher the value of the energy is. A logo image is more homogeneous than other images (large areas with the same pixel intensity), leading to higher values of energy. Energy is computed as

$$\text{energy} = \sum_i (prob[i])^2$$

Entropy [2]: Measures the average bits per pixel. For an 8-bit image it takes values between 0 and 8. The larger the range of pixel intensities and their distribution, the higher the value of entropy (approaching 8). Small entropy indicates few intensity levels and the presence in the image of regions in which there is little or no variation in pixels values (graphic images). The entropy of the intensity histogram for a logo/trademark image is smaller than other images, ranging between 0.5 and 2.5, while typical values for non-logo images are between 4.5 and 7. The entropy is computed as

$$\text{entropy} = \sum_i \left(prob[i] \times \log_2 (prob[i]) \right)$$

As mentioned above, **occupied bins** [2] is a feature measured only on intensity histograms. It ranges between 1 and 256 and it indicates the number of intensity values used in each image. An image with 250 occupied bins covers almost the full range of pixel intensities, while an image with 10 occupied bins uses only a few of them. Logo and trademark images are graphic images with few intensity levels, which implying few occupied bins (fewer than the non-logo ones). The use of this feature on the other two histograms types is meaningless since they cover the full range of pixel intensities.

Image size measures as file size (in bytes): Logo-Trademarks tend to be smaller than the other types of images.

Table 2.1 presents two image representation vectors, one for a logo image and one for a non-logo.

	Features		
Intensity Histogram	Mean	184.466	67.8259
	Standard Deviation	88.2025	40.537
	3rd Order Central Moment	-0.723273	0.879701
	4th Order Central Moment	-1.21193	0.215777
	Otsu Threshold	169	73
	Occupied Bins	154	135
	Entropy	4.24811	6.71443
	Energy	0.231521	0.011296
Radial Frequency Histogram	Mean	124.317	139.594
	Standard Deviation	65.5305	82.9229
	3rd Order Central Moment	0.731467	-0.566765
	4th Order Central Moment	-0.751995	-1.411
	Otsu Threshold	143	122
	Entropy	6.68266	8.97264
	Energy	0.00366401	0.00523903
Angle Frequency Histogram	Mean	126.426	140.3169
	Standard Deviation	67.3482	70.7094
	3rd Order Central Moment	1.53676	1.59622
	4th Order Central Moment	-0.318597	-0.20073
	Otsu Threshold	186	187
	Entropy	4.57153	4.28171
	Energy	0.00281608	0.00230898
	Filesize	3730	308278

Table 2.1: Image representation vectors.

A 23-dimensional vector represents an image with each dimension corresponding to one of the above-defined features measured on each one of the 3 types of histograms. The 23 features are: 8 features are derived from the intensity histogram, 14 features are derived from the two frequency histograms (7+7) and 1 feature is the image size.

An analysis of the features computed to several logo and non-logo images reveals that, generally, intensity histogram's mean value and threshold take higher values for logos-trademarks than in other types of images. Besides entropy and 4th order central moment are smaller, energy is higher and occupied bins are fewer (the example of table 2.1 is an exception). Radial frequency histogram's Otsu threshold and 4th order central moment are higher for logo images. Angle frequency histogram's Otsu threshold is, usually, smaller.

2.2. Machine Learning for logo and trademark detection

2.2.1. Decision Trees

Decision Trees [8] are the natural result of a "divide-and-conquer" approach to the problem of learning from a set of independent instances. Each node in a decision tree involves testing a particular attribute, while leaf nodes give a classification to all instances that reach the leaf. Attributes represent the columns while instances the lines of the data set.

The construction of a decision tree is a recursive process. At each step, it involves the selection of an attribute to place at a node (starting at the root node) along with spitting the example set into subsets based on the values that the members of the example set take for the selected attribute. Then the same process is repeated recursively for each branch until all the instances at a node have the same classification or until all attributes have been tested. The decision of how to determine which attribute to split is based on the measure of "*purity*" of each node, called "**information**" [9] and represents the expected amount of information that would be needed to specify whether and new instance should be classified into some class (ex. "yes" or "no"), given that the example reached that node. Before a split, the purity of the new node is computed. Then the information gain is computed and the one with the higher gain is chosen to split on. The computation of node information or purity is based on entropy:

$$\text{entropy}\left(p_1, p_2, \ldots, p_n \right) = - p_1 \log p_1 - p_2 \log p_2 \ldots - p_n \log p_n$$

For a hypothetic three class situation with instances n_1, n_2, n_3 and a total number of instances N then the info is computed as:

$$\operatorname{info}([n_1, n_2, n_3]) = \operatorname{entropy}(n_1 / N, n_2 / N, n_3 / N)$$

One major problem of decision trees is that they are usually over-fitted to the training data and do not generalize well to independent test sets [10]. By "over-fitting", we mean that the decision tree follows the training data too slavishly (figure 2.7 (a)). In this case, the training set contains artifacts of the actual values used to create the decision tree, which are not genuine features of the "real" data set. The solution to this problem is pruning. Figure 2.7 (a) illustrates an un-pruned, overfitted tree and figure 2.7 (b) the same after pruning. The majority of the instances of the left sub-tree of attribute 1 belong to *Class 1*. Thus a single leaf of Class 1 has replaced the sub-tree. The left tree performs better on training data set, while the right one performs better on independent data sets.

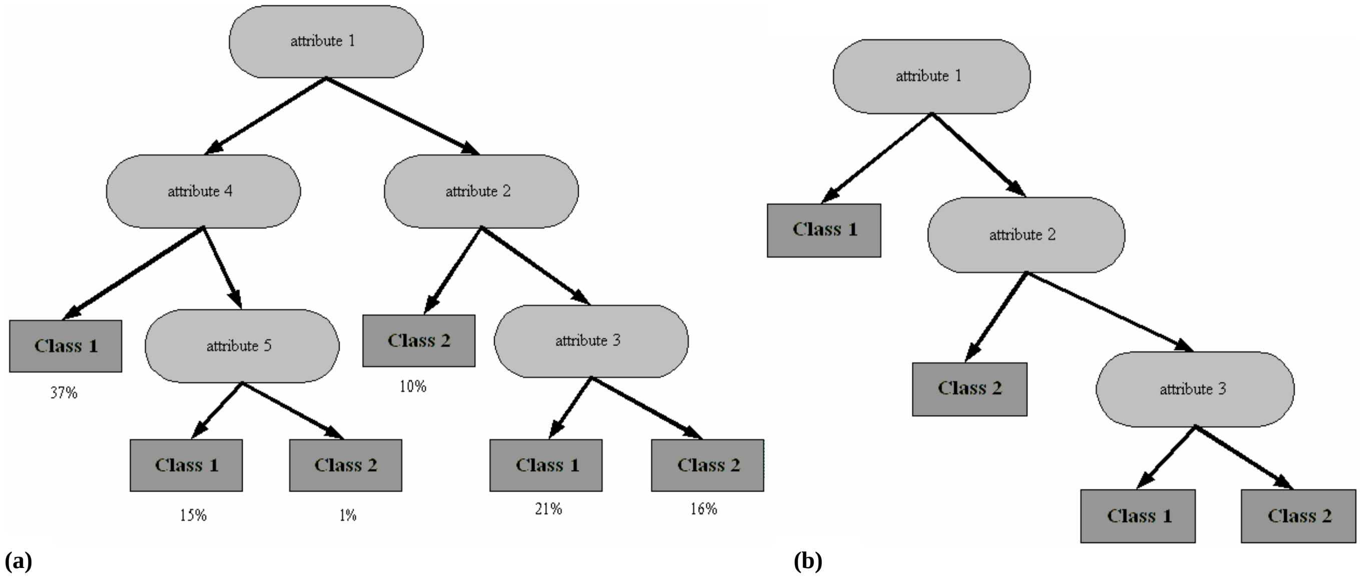


Figure 2.7: Unpruned and pruned decision trees

There are two pruning strategies: *post-pruning (or backward pruning)* and *pre-pruning (or forward pruning).* Pre-pruning involves trying to decide when to stop developing sub-trees during the tree-building process while post-pruning involves processing the tree after its construction. Post-pruning works better and it is the method that C4.5 (the decision tree program we used) utilizes [10]. Two different operations have been considered for post-pruning:

sub-tree replacement and *sub-tree raising*. Subtree replacement selects some sub-trees and replaces them by single leaves (figure 2.7) whereas sub-tree raising raises an entire sub-tree to replace the above one and then reclassifies the examples at the new nodes (figure 2.8). Both operations lead to decreased accuracy on the training set but to increased accuracy on independent test sets.

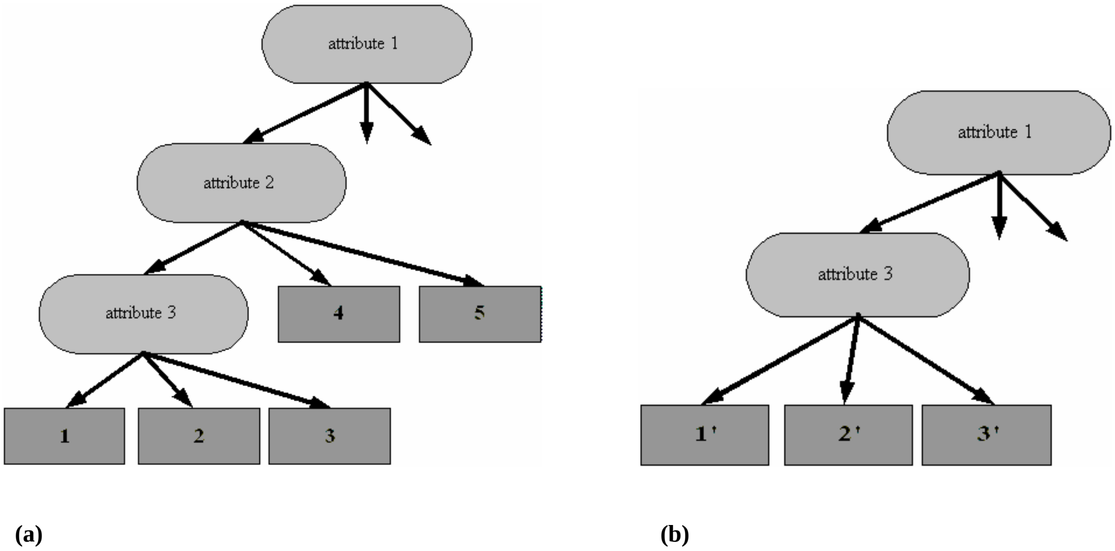


Figure 2.8: Example of sub-tree raising

On figure 2.8 the entire sub-tree from attribute 3 downward has been raised to replace the attribute 2 sub-tree. It should be mentioned that the daughters of attributes 2 and 3 are not necessarily leafs, as they can be entire sub-trees. After the raising operation, is necessary to reclassify the examples at the nodes 4 and 5 into the new sub-tree headed by attribute 3. This is why the new daughters of node 3 are marked as: 1',2',3'.

At each node the learning scheme decides whether it should perform one of the above operations or leave the sub-tree unpruned. To make this decision, it is necessary to estimate the expected error rate at a particular node given an independent test set. One-way of coming up with this error estimate is the standard verification technique: this technique suggests, holding back some of the data given for training and use it as an independent test set. This method is called reduced error pruning and it suffers from the decrement of data for tree training. The other method, that C4.5 utilizes, is to make some error estimates based on the training data itself. It is a heuristic based on statistical reasoning and works well in practice. The idea is to imagine that the majority class, of the set of instances that reach each node, is chosen to represent that node [10]. That gives a number of errors E, out of the total instances N. Imagine that the true probability of error at a node is q, that the N instances are generated by Bernoulli process with parameter q, of which E turn out to be errors and that the observed error rate is $f = E/N$. Given a confidence c (described below), we find confidence limits z such that:

$$\Pr\left[\frac{f-q}{\sqrt{q(1-q)/N}} > z\right] = c$$

This leads to an upper confidence limit for q. We use that upper confidence limit as a pessimistic estimate for the error rate e at the node:

$$e = \frac{f + \frac{z^2}{2N} + z\sqrt{\frac{f}{N} - \frac{f^2}{N} + \frac{z^2}{4N^2}}}{1 + \frac{z^2}{N}}$$

Note that z is the number of standard deviations corresponding to the confidence c, which for c = 25% is z = 0.69.

C4.5 is a decision tree program that uses a two way (binary) split for numeric attributes. It includes a number of options and parameters such as “confidence value” (for controlling the degree of pruning), or “test method” (for selecting the test method). Tuning a decision tree is a time-consuming process and needs to be done carefully. Sections 2.2.2 and 2.2.3 describe the process of tuning the two different decision trees as well as the different types of tests.

2.2.2. Method 1: Training based on Row Histograms

The first training method for trademark and logo extraction is based on row histograms (section 2.1.1.). The training data set consists of 1180 instances (675 logo-trademarks and 505 images of other types). Each instance (image) is described by a 768-dimesional vector, which is obtained by concatenating the three raw histograms of the image (3 histograms times 256 values each = 768 values or features).

For each image the decision tree gives an estimate as to whether it is logo or trademark image or not. We experimented with several trainings methods each one corresponding to different confidence parameters controlling the rate of pruning. Low values of confidence lead to drastic pruning, while higher values lead to a milder pruning. Besides, each tree was tested with many different test methods, such as “cross validation”, “split on training set” and “independent test set”. The characteristics of each method are discussed below. Table 2.2 illustrates the results of this experiment.

Row Histograms				
Tree	**Confidence**	**Test Method**	**Size(leaves/nodes)**	**Success %**
Pruned	0.25	Stratified Cross-Validation	44/87	88.661
Pruned	0.25	Split on data (66%)	44/87	86.8159
Pruned	0.25	Test set (100)	44/87	85.9259
Pruned	0.1	Stratified Cross-Validation	30/59	89.9153
Pruned	0.1	Split on data (66%)	30/59	87.8109
Pruned	0.1	Test set (100)	30/59	85.0000
Unpruned	-	Stratified Cross-Validation	44/87	88.661
Unpruned	-	Split on data (66%)	44/87	86.8159
Unpruned	-	Test set (100)	44/87	85.9259

Table 2.2: Decision Trees Results for Row Histograms.

The first column indicates whether a tree is pruned or not while the second one indicates the degree of pruning. Confidence value 0.25 (Weka’s default value) causes a less drastic pruning than confidence value 0.1. The next column corresponds to the test methods. Stratified cross-validation is a standard method for testing decision trees [11]. A fixed number of folds (partitions of the data) are chosen (in our case 10). Then the data is split into ten (approximately equal) partitions, and each one is used for testing while the remainder $(9/10)$ is used for training. The whole procedure is repeated ten times so that every instance has been used once for testing (*ten-fold stratified cross-validation*). At the end the mean value of the ten trainings is returned. The term stratified indicates that each class is properly represented in both training and test sets. Split on data simply uses the 1/3 of the data for testing and the rest for training. At this point it should be mentioned that both stratified cross-validation and split on data use the *whole* data set for the production of the decision tree. Accordingly the test results are related with trees that come from smaller data sets. The last test method is the Test set. It uses an independent data set of 100 instances that were not included on the training set. The fourth column indicates the size of every tree (leaves/nodes) and the last one the success percentage.

The variation of tree pruning from confidence 0.1 (drastic pruning) to unpruned, creates minor effect to final percentage. Besides trees with confidence value 0.25 and unpruned ones are the same (44/87). In this work the selected confidence value is 0.25.

2.2.3. Method 2: Training based on Histogram Features

The second training method for logo-trademark extraction is based on features computed on the three types of histograms (section 2.1.2. The training data set has again 1180 instances (675 logo-trademarks and 505 different types of images) but only 24 features. These features are 8 from the intensity histogram, 7 + 7 from the radial and angle frequency histograms, the file size of each image (1) and the class they belong to (1). The same process with section 2.2.2 was followed. We tried different confidence values and different test methods. Table 2.3 illustrates the results of this experiment.

Features				
Tree	**Confidence**	**Test Method**	**Size(leaves/nodes)**	**Success %**
Pruned	0.25	Stratified Cross-Validation	22/43	92.9661
Pruned	0.25	Split on data (66%)	22/43	92.5373
Pruned	0.25	Test set (100)	22/43	90.7143
Pruned	0.1	Stratified Cross-Validation	20/39	93.0508
Pruned	0.1	Split on data (66%)	20/39	92.2886
Pruned	0.1	Test set (100)	20/39	90.7143
Unpruned	-	Stratified Cross-Validation	23/45	92.7119
Unpruned	-	Split on data (66%)	23/45	92.5373

Unpruned	-	Test set (100)	23/45	90.7143

Table 2.3: Decision Trees Results for Row Histograms.

Unpruned tree is the largest of the three trees. The 0.25 confidence value tree comes second and the 0.1 one third. However the final percentage varies a little. Again, we selected the tree with confidence value 0.25.

2.2.4. SVD

A problem with the above methods (especially the second one) is that many of the attributes are linearly dependent to each other. This may be a drawback of the machine learning and the construction of the decision tree. The idea is to decompose the input training matrices. SVD *(singular value decomposition)* is the method we used for this purpose [12]. This method is expected to give better quality (uncorrelated) dataset. SVD is based on the following theorem of linear algebra: “An $M \times N$ matrix A whose rows M is greater than or equal to its number of columns N, can be rewritten as the product of an $M \times N$ column-orthogonal matrix U, an $N \times N$ diagonal matrix W with positive or zero elements, and the transpose of an $N \times N$ orthogonal matrix V.”

$$\begin{pmatrix} & & \\ & A & \\ & & \end{pmatrix} = \begin{pmatrix} & & \\ & U & \\ & & \end{pmatrix} \cdot \begin{pmatrix} w_0 & & & \\ & w_1 & & \\ & & \cdots & \\ & & & w_{N-1} \end{pmatrix} \cdot \begin{pmatrix} & & \\ & V^T & \\ & & \end{pmatrix}$$

Matrix A is the initial one and matrix U is the new input training matrix, which represents the improved information. The elements of U are nondimensional and have no natural meaning. They are just numbers representing the initial data set. Applying this matrix to the two methods described above gives the following results:

Row Histograms (SVD)				
Tree	**Confidence**	**Test Method**	**Size(leaves/nodes)**	**Success %**
Pruned	0.25	Stratified Cross-Validation	64/127	74.9463
Pruned	0.25	Split on data (66%)	64/127	75.6219
Pruned	0.25	Test set (100)	64/127	73.213
Pruned	0.1	Stratified Cross-Validation	58/115	75.5496
Pruned	0.1	Split on data (66%)	58/115	76.1194
Pruned	0.1	Test set (100)	58/115	73.5231
Unpruned	-	Stratified Cross-Validation	67/133	74.9153
Unpruned	-	Split on data (66%)	67/133	75.6219
Unpruned	-	Test set (100)	67/133	72.819

Table 2.3: Decision Trees Results for Row Histograms (SVD).

Again, there is a minor variation between the different types of trees. However, the final decision trees are much larger than the ones of method 1 (section 2.2.2) and the results are worse.

Features (SVD)				
Tree	**Confidence**	**Test Method**	**Size(leaves/nodes)**	**Success %**
Pruned	0.25	Stratified Cross-Validation	34/67	91.9492
Pruned	0.25	Split on data (66%)	34/67	91.791
Pruned	0.25	Test set (100)	34/67	89.6123
Pruned	0.1	Stratified Cross-Validation	22/43	91.0169
Pruned	0.1	Split on data (66%)	22/43	90.796
Pruned	0.1	Test set (100)	22/43	89.7245
Unpruned	-	Stratified Cross-Validation	34/67	90.5085
Unpruned	-	Split on data (66%)	34/67	91.5423
Unpruned	-	Test set (100)	34/67	89.2074

Table 2.4: Decision Trees Results for Row Histograms (SVD).

The trees of feature training method with SVD are smaller than trees of histogram training method. Also they perform better. Section 2.2.5 illustrates a comparison of all methods.

2.2.5. Comparison

Before we compare the four methods (Row Histograms and Features with or without SVD), it should be mentioned that for all these methods the variations between the different test modes are minor, so hereafter we will use the *ten-fold stratified cross-validation*.

A first note concerns the size of the decision trees. SVD trees are larger than the other types and Row Histograms trees are larger than Features trees. Of course strongly pruned trees (confidence value 0.1) are the smallest ones, while unpruned trees are the largest. The size of the tree is an indication of how over-fitted to the training data the decision tree is. This must be the reason for the reduced performance of the SVD methods (especially those applied on Row Histograms).

This result suggests that SVD training should be repeated with smaller dimensional features obtained by selecting only higher order (corresponding to the most significant) SVD features and ignoring lower order (less significant features). Lower order feature correspond to less discriminating features that could potentially bias the results of the classification overall. However the experimentation did not caught this expectation.

As mentioned on section 2.2.1, over-fitted decision trees fail to generalize well to test sets as they are trapped on the artifacts of the training data set that do not apply to the test sets. Table 2.5 summarizes the results. Figure 2.7 illustrates the performance of the four methods tested.

Comparison				
Tree	**Confidence**	**Training Method**	**Size(leaves/nodes)**	**Success %**
Pruned	0.25	Row Histograms	44/87	88.661
Pruned	0.25	Features	22/43	92.9661
Pruned	0.25	Row Histograms (SVD)	64/127	74.9463
Pruned	0.25	Features (SVD)	34/67	91.9492
Pruned	0.1	Row Histograms	30/59	89.9153
Pruned	0.1	Features	20/39	93.0508
Pruned	0.1	Row Histograms (SVD)	58/115	75.5496
Pruned	0.1	Features (SVD)	22/43	91.0169
unpruned	-	Row Histograms	44/87	88.661
unpruned	-	Features	23/45	92.7119
unpruned	-	Row Histograms (SVD)	67/133	74.9153
unpruned	-	Features (SVD)	34/67	90.5085

Table 2.5: Comparison of the four methods. Test method: cross-validation.

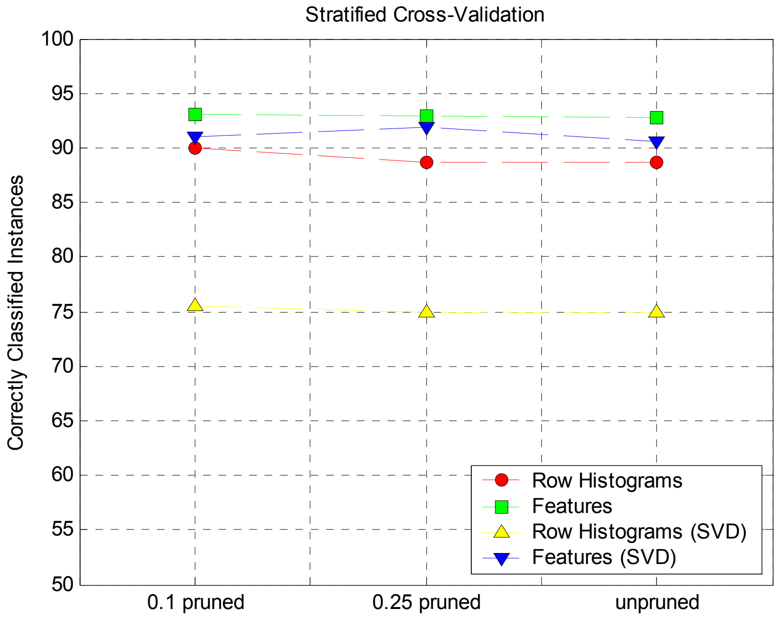


Figure 2.7: Performance of the four methods

Methods based on features computed on histograms seem to outperform methods based on raw histograms alone. Feature based methods seem to achieve up to 93% correctly classified instances, followed by Feature-based methods with SVD achieving up to 91,5% correctly classified instances. However, a further improvement of the SVD-based method is reasonably expected by selecting only the more significant features (after SVD) for training. Generally, methods based on features perform better than methods based on raw histograms. The reason is that the input training data is more elaborated than the simple histograms. Each feature describes a characteristic of an image that discriminates logos-trademarks from the rest images. In reverse, the information that comes from histograms is a low level description of the images' content and it is rather difficult to produce an effective decision tree. We finally selected the 0.25 pruned trees that suggest a mild pruning and perform better on independent test sets.

3. Logo and Trademark Similarity

The second step of the method is to detect logo and trademark images (from those detected in the previous step), which are characteristic of the content of the Web site. Because various instance of the same logo or trademark image may be repeated within the same Web site, the next step is to group all similar logo and trademark images into clusters. These images can be either identical or very similar. All similar images must be grouped together, and from each cluster, one image will be selected to represent the cluster in the summary.

3.1. Image Features

The approach depends on a method for computing image similarity. In turn, image similarity can be computed as a function similarity of features in the two images. Image similarity is computed as a function of differences between the 8 features discussed earlier (section 2.1.2), histogram intersection (section 3.1.1) and moment invariants (section 3.1.2). If two images are identical or similar enough, their features must have minor differences. Table 3.1 contains the difference of features for three images. The differences between the features of the first similar pair of images are significant smaller than those of the other two pairs, confirming the idea.

3.1.1. Histogram Features

The idea here is to compare the histograms themselves instead of their features. Histograms can be compared by an *intersection* operation [13]. Let A and B be two histograms containing N bins. The intersection of these two histograms is defined as:

$$\sum_{i=1}^{N} \min(A_i, B_i)$$

The above formula may be interpreted as enumerating the number of pixels, which are common to both histograms. Both histograms are normalized, so the summation takes values between 0 and 1. Table 3.2 shows the values of the three histograms intersections corresponding to the same three images as above. The similarity values computed on the intensity histograms of two actually similar images is much higher than the similarity corresponding to the other two pairs (i.e., 0.8 value indicates two almost identical histograms). The variations of the other two histogram intersections are not useful in detecting image similarity, since the majority of images exhibit almost a similar power spectrum.

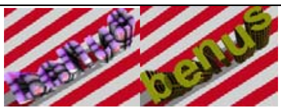
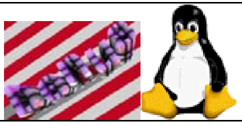
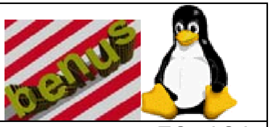
	Features			
Intensity Histogram	Mean	6.91135	66.2197	73.131
	Standard Deviation	0.467003	23.9984	23.5314
	3rd Order Central Moment	0.133557	1.34538	1.47894
	4th Order Central Moment	0.136172	1.51789	1.38172
	Otsu Threshold	5	40	45
	Occupied Bins	0	46	46
	Entropy	0.088028	2.54202	2.45399
	Energy	0.00114516	0.193126	0.191981
Radial Frequency **Histogram**	Mean	0.742126	2.57901	1.83688
	Standard Deviation	0.971741	9.69	8.71826
	3rd Order Central Moment	0.0185456	0.180778	0.162232
	4th Order Central Moment	0.0201076	0.146158	0.126051
	Otsu Threshold	0	7	7
	Entropy	0.0872612	0.700646	0.613385
	Energy	0.000146785	0.000777077	0.000630292
Angle Frequency **Histogram**	Mean	6.91135	66.2197	73.131
	Standard Deviation	0.219185	0.670441	0.889626
	3rd Order Central Moment	0.000536203	0.0135771	0.0141133
	4th Order Central Moment	0.00446296	0.0311601	0.0266972
	Otsu Threshold	0	0	0
	Entropy	0.0184846	0.0598555	0.0783401
	Energy	0.00281608	0.00230898	0.000119419

Table 3.1: Differences between features.

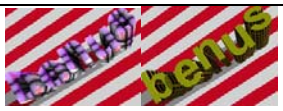
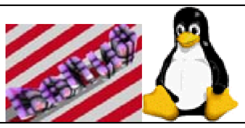
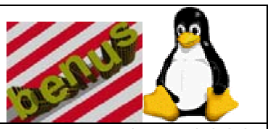
Histograms			
Intensity	0.800312	0.185041	0.150666
Radial Frequency	0.91415	0.92181	0.92304
Angle Frequency	0.589338	0.590993	0.589369

Table 3.2: Intersection of Histograms

3.1.2. Moments

The similarity between two images can also be computed with moment invariants [14]. Before applying moments, images are converted into binary by thresholding, (i.e., the Intensity Histogram Otsu threshold of section 2.1.2 is applied). Moments assume that non-zero pixel values represent regions. Invariant moments are independent on scaling, translation and rotation. Computing invariant moments involves computing normal moments and central moments. Calculation of normal moments uses the type:

$$m_{pq} = \sum_{i=-\infty}^{\infty} \sum_{j=-\infty}^{\infty} i^p j^q$$

where i, j are the pixel co-ordinates. Translation invariance can be achieved by using the central moments:

$$\mu_{pq} = \sum_{i=-\infty}^{\infty} \sum_{j=-\infty}^{\infty} (i - x_c)^p (j - y_c)^q$$

where x_c, y_c are the co-ordinates of the regions center of gravity (centroid), which can be obtained using the relationships:

$$x_c = \frac{m_{10}}{m_{00}} \ , \ y_c = \frac{m_{01}}{m_{00}}$$

where m_{00} represents the region area (binary case). Scaled invariance is achieved with the un-scaled central moments:

$$\theta_{pq} = \frac{\mu_{pq}}{(\mu_{00})^{\gamma}} \text{ where } \gamma = \frac{p+q}{2} + 1$$

rotation invariance is achieved with the seven invariant moments:

$$\phi_1 = \theta_{20} + \theta_{02}$$
$$\phi_2 = (\theta_{20} - \theta_{02})^2 + 4\theta_{11}{}^2$$
$$\phi_3 = (\theta_{30} - 3\theta_{12})^2 + (3\theta_{21} - \theta_{03})^2$$
$$\phi_4 = (\theta_{30} + \theta_{12})^2 + (\theta_{21} + \theta_{03})^2$$
$$\phi_5 = (\theta_{30} - 3\theta_{12})(\theta_{30} + \theta_{12})\left[(\theta_{30} + \theta_{12})^2 - 3(\theta_{21} + \theta_{03})^2\right] + (3\theta_{21} - \theta_{03})(\theta_{21} + \theta_{03})\left[3(\theta_{30} + \theta_{12})^2 - (\theta_{21} + \theta_{03})^2\right]$$
$$\phi_6 = (\theta_{20} - \theta_{02})\left[(\theta_{30} + \theta_{12})^2 - (\theta_{21} + \theta_{03})^2\right] + 4\theta_{11}(\theta_{30} + \theta_{12})(\theta_{21} + \theta_{03})$$
$$\phi_7 = (3\theta_{21} - \theta_{03})(\theta_{30} + \theta_{12})\left[(\theta_{30} + \theta_{12})^2 - 3(\theta_{21} + \theta_{03})^2\right] - (\theta_{30} - 3\theta_{12})(\theta_{12} + \theta_{03})\left[3(\theta_{30} + \theta_{12})^2 - (\theta_{21} + \theta_{03})^2\right]$$

The above seven invariant moments describe the images and are rotation-, translation-, and scale-invariant. Then Euclidian distance of invariant moments between pairs of images is computed. Similar images give small difference.

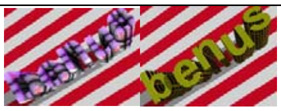
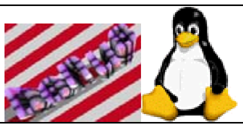
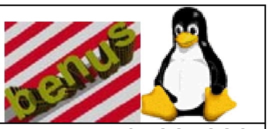
Invariant Moments			
Euclidean distance (0-1)	0.072852	0.162654	0.235338

Table 3.3: Euclidian distance on moment Invariants.

The Euclidean distance for the first pair of images is smaller than the other two pairs, as the first one consists of similar images.

3.2. Training

The purpose of this step is to train a decision tree for detecting pairs of similar images.

3.2.1. Decision Tree

The decision tree for detecting similar images is identical to the decision tree of section 2.2.1 for detecting logo and trademark images. The training data set has 1888 instances, 390 similar pairs and 1498 non-similar ones. Each image pair is represented by the differences computed over the set of features (i.e., a set of 8 differences corresponding to difference of the 8 image features, three histograms intersections and the Euclidian distance of the seven invariant moments), that is 26 features (differences) total. The decision tree accepts a pair of images (in fact differences of 26 features) as input and computes whether the two images are similar or not. The output tree is pruned with 0.1-confidence value and the stratified-cross validation gives *89.7648%*.

3.3. Image Distance Function

An interesting expansion is the definition of an image distance function. This image function is based on the summation of the, normalized, features of section 3.1. Features are normalized by dividing each one with each maximal value. Besides we use *1 - histogram intersection*, as histograms intersections are 1 when two images are identical. Image distance function is defined as:

$$D(A,B)=\sum_i w_i d(A_i,B_i)$$

Where A,B are two images, i is the number of features and $d(A_i,B_i)$ is the distance between two images for feature i. Terms w_i represent the relative importance of the features themselves. The last issue before defining an image distance function is the specification of weights. Weight specification utilizes the decision tree.

We proposed that weights are computed based on properties of a trained decision tree as follows

$$w_{f_i} = \sum_{node_j=f_i} \frac{Maxdepth+1-depth(f_i)}{\sum_j Maxdepth+1-depth(node_j)}$$

Where f_i is every feature, $node_j$ is each node of the decision tree and Maxdepth the maximum depth of the tree. The summation is taken over all nodes. This formula suggests that the higher a feature is and the more frequently it appears, the higher its weight will be. The final type contains 16 features, as some of the features of section 3.1 do not appear on the final tree. Features not appearing in the decision tree are not taken into account in the computation of distance. Table 3.4 illustrates three pairs of images and their distance according to the formula above. Notice, that the distance of the two first images is smaller than the other pairs.

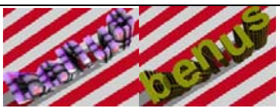
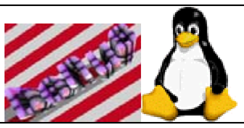
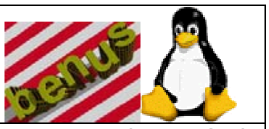
Image Distance			
Distance (0-1)	0.178618	0.444577	0.455349

Table 3.4: Image distances

3.4. Image Clustering

The goal of this is to group all the similar images together into clusters. For this purpose we introduce two methods. The first one uses the image distance function (defined above), while the second one utilizes the decision tree itself.

3.4.1. Method 1: Histogram Threshold

This method is based on the image distance function, and its purpose, firstly, is to decide whether two images are similar and then to group all the similar images together into clusters. The idea is that the distribution of distances of general images is Gaussian. Similarity is regarded as an exception: only a few pairs are expected to be similar and have high values of similarity (equivalently low values of distance). Based on this observation, we define two images as similar if they have distance lower than $\mu - \tau\sigma$, where t is a user defined threshold (in this work τ=1.5) and μ, σ are the mean and standard deviation of the distribution of distances between the training set of images. Figure 3.1 illustrates the method:

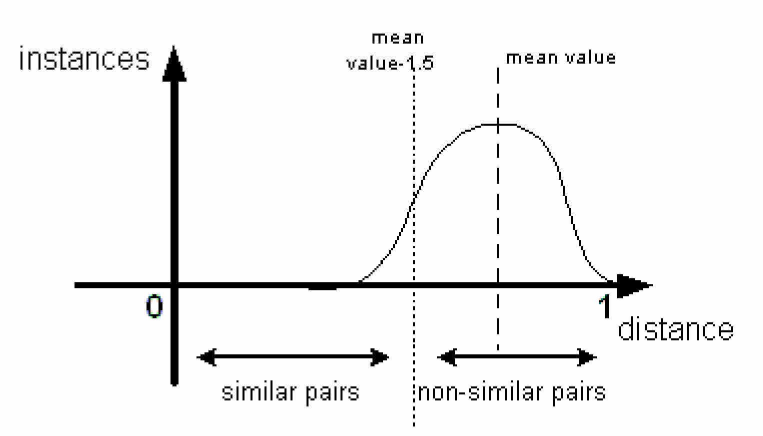


Table 3.1: Histogram Threshold Method

When training the method is as follows:

```
Input: features of section 3.1 for pairs of images
Output: threshold to detecting similar images

1.  for each pair
    1.1. compute image distance
2. create the histogram of distances (1000 bins)
3. compute mean value and standard deviation of histogram
4. compute threshold for similar pairs as $T=\mu-1.5\sigma$
5. output $\mu,\sigma,T$
```

The next step is to define an algorithm for clustering similar images. The method is as follows:

```
Input: features of section 3.1 for pairs of images
Output: clusters of similar images

1.  for each pair of images
    1.1. compute their distance
2. find pairs of images with distance less than T
3. for each pair (with distance less than T)
    3.1 compare with all the other similar pairs.
    3.2 if they have at least one image in common merge and
    continue with 3.1
    3.3 if it has been compared with all the pairs, keep the cluster,
    select the next pair and continue with 3.1. At the end all the pairs
    with similar images are grouped together.
4. output the clusters of similar images
```

3.4.2. Method 2: Graph Clustering

This method uses the decision tree for detecting the similar images. A more sophisticated method for image clustering is based on the idea of "cliques". A set of images forms a clique, if every image in the set is similar to every other image within the same set. Typically, a clique finding algorithm works on a graph: Each image corresponds to a node of the graph; any two nodes (images) are connected by an edge if the two nodes are similar. A clique is a fully connected component of the graph. An algorithm for finding all cliques on the above graph is applied [15].

This method worked really better eliminating the problem of *all in one cluster*. It includes two steps. The input of the first step of the algorithm is set of instances. Each instance corresponds to a pair of images, and includes 26 features (a set of 8 differences corresponding to difference of the 8 image features, three histograms intersections and the Euclidian distance of the seven invariant moments – section 3.2.1). The input is formulated as an *arff* file[*]. The output of the algorithm is the prediction of the algorithm for each pair of images (similar/not-similar). When *training* the algorithm is as follows:

```
Input: 15 features for each image (mean value, threshold, standard
deviation, third order central moment, fourth order central moment,
energy, entropy, occupied bins and the 7 invariant moments).
Output: - (trains decision tree)

1. for each pair of images (each instance)
    1.1. compute Euclidian distance of invariant moments, distance
    features (mean value, threshold, standard deviation, third order
    central moment, fourth order central moment, energy, entropy,
    occupied bins) and histograms intersections
    1.2. assign the human defined class (similar, not-similar)
2. form the above features to arff file.
3. pass the arff file through the decision tree for training.
```

[*] The Arff file is a standard way of representing datasets that consist of independent, unordered instances and does not involve relationships between instances [16]

When *using* the decision tree, the algorithm is as follows:

```
Input: 15 features for each image (mean value, threshold, standard
deviation, third order central moment, fourth order central moment,
energy, entropy, occupied bins and the 7 invariant moments).
Output: prediction for each pair of images (similar/not-similar)

1. for each pair of images (each instance)
    1.1. compute Euclidian distance of invariant moments, distance
    features (mean value, threshold, standard deviation, third order
    central moment, fourth order central moment, energy, entropy,
    occupied bins) and histograms intersections
2. form the above features to arff file.
3. pass the arff file through the decision tree.
4. find the similar pairs of images.
```

The figure below summarizes the second step of the method, the clustering algorithm:

```
Input: the prediction of the previous step for each pair of images
Output: clusters of similar images

1. find cliques on the set of similar pairs of images
2. each clique represents a cluster. If an image belongs
to two or more clusters keep the largest one.
3. output clusters
```

3.5. Experiments on image clustering

Given the initial set of images of table 3.5 both methods worked perfect, recognizing the similar images and grouping them to clusters. However for larger sets of images the method with the cliques outperforms the first one, which tends to create very large clusters of non-similar images. On the other hand, the second method (based in clique finding) tend to break large clusters of similar images into two or three sub-clusters of common images. An obvious improvement would be to relax the requirement of perfect cliques and allow for not fully connect components in cliques (e.g., consider that a clique is formed by nodes connected to all but one other nodes).

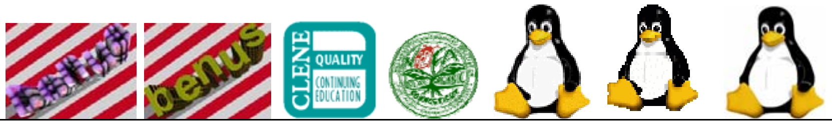
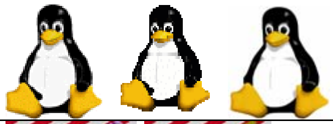
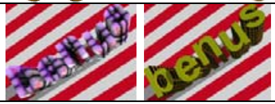
Initial Test-set	
Group 1	
Group 2	
Group 3	
Group 4	

Table 3.5: Image cluster

4. Logo and Trademark Ranking

The purpose of this step is to find the most important-characteristic logos and trademarks of a web-site (only the most important logos-trademarks should be displayed).

4.1. Methods based on image depth and back-links for finding the most important image from a Web site

The methods for logo and trademark ranking are based on the importance of each image. The first method ranks each image individually and picks one image from each cluster while the second one initial ranks the clusters (rather than the images) and then picks the most characteristic image from each cluster. Sections 4.1.1 and 4.1.2 describe further the two methods for image ranking.

4.1.1. Method 1

The main idea of this method is that the higher an image at the web-site hierarchy, the more important it is. Also, the more the links to that image are (pages pointing to the image or to the page containing the image) and the higher the probability of being logo-trademark is, the more important it is. The following formula combines the above ideas:

$$importance = depth * \left(\frac{backLinks}{allLinks} \right) * probability \, ,$$

where $backLinks$ is the number of links to the image, $allLinks$ is the total number of $backlinks$ to all images in the cluster and $probability$ is the logo-trademark probability of the image. Finally, depth is defined as the minimum number of links from the root page that need to be visited in order to access the image. Depth is computed as:

$$depth = \frac{Maxdepth + 1 - linkdepth}{Maxdepth}$$

Where *Maxdepth* is the maximum depth of a web-site (the depth of the inner-most image) and *linkdepth* is the actual depth of image (in number of links required to access the image starting from the root page). Depth varies between 0-1. An image that is included at the root page of a web-site has the maximum depth 1. Notice that, importance takes values between 0 and 1. This method first ranks the images of a cluster and then picks the most important image from each cluster. This image represents the cluster as a whole. The algorithm below summarized this process:

```
Input: the clusters of similar images
Output: the most characteristic-important images

1. for each image in the web-site
    1.1. compute its importance by

         $importance = depth * \left( \frac{backLinks}{allLinks} \right) * probability$

2. sort images by importance
3. pick the most important image from each cluster
4. output the most important images of all clusters
```

4.1.2. Method 2

The main idea of this method is the same with method 1. The difference is that this method *first ranks the clusters by importance and then it selects the most characteristic image form each cluster*. At the end the most important image is the one that belongs to the most important cluster. The importance of a cluster is computed by adding the importance of the images it contains. The algorithm for this method is as follows:

```
Input: the clusters of images
Output: the most characteristic-important images of a Web site

1. for each image
    1.1. compute its importance as

        $impor\tan ce = depth*\left(\frac{backLinks}{allLinks}\right)* probability$

2. sort images by importance
3. for each cluster
    3.1 compute importance by adding the importance of its
    images
4. sort clusters by importance
5. for each cluster (starting with the most important)
    5.1 pick the most important image from each cluster
6. output the most important images
```

5. Evaluation of the Method

This section presents experimental results. It includes experiments on different Web sites and it demonstrates the step-by-step operation of the method (logo & trademark extraction, similarity, ranking). It ends up with the most important-characteristic images of the web sites.

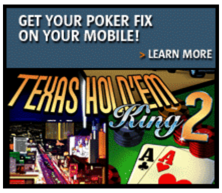
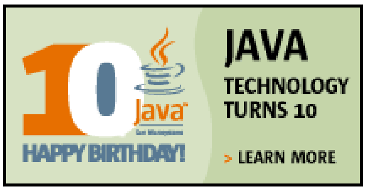
The following sections include images from two Web sites: www.java.com and www.suse.com.

5.1. 1st Web Site: www.java.com

The following experiment uses images from java's web site.

5.1.1 Logo and Trademark Extraction

Table 5.1 contains images derived from the web-site. Notice that it contains only 21 images. The program does not actually download images from the web site. What it does to communicate with a database that contains portions of web sites.

1		2	Verify Installation	3		4	JEWEL QUEST THOUSANDS OF INSTANT WINNERS I-play Mobile gaming
5	 GET YOUR POKER FIX ON YOUR MOBILE! > LEARN MORE TEXAS HOLD'EM King 2	6	WARNER MOBILE MUSIC STREAMING JUKEBOX THE LATEST FROM FAT JOE > CHECK IT OUT	7	THEME OF THE MONTH MULTI-PLAYER GAMES & SOFTWARE BEJEWELED multiplayer PUZZLE PIRATES Soccer > LEARN MORE	8	 10 Java HAPPY BIRTHDAY! JAVA TECHNOLOGY TURNS 10 > LEARN MORE
9	for your Mobile Phone > LEARN MORE	10		11		12	Java Everywhere

13	 	14	 	15	 Welcome to java.com. Brought to you by Sun Microsystems.	16	
17	 	18	 Java Sun Microsystems	19	 Java Sun Microsystems	20	
21							

Table 5.1: Images from web-site.

For the next step the histogram features process, described at section 2 was used. The features of section 2.1.2 were derived for each of the above images. These features formed a data-set for the decision tree, which selected the following images as logos-trademarks:

1	 10 Java JAVA TECHNOLOGY TURNS 10 HAPPY BIRTHDAY! > LEARN MORE	2	 V-girl™ - your virtual girlfriend for your Mobile Phone > LEARN MORE	3	 	4	
5	 Java Everywhere	6	 	7	 	8	 Welcome to java.com. Brought to you by Sun Microsystems.
9	 Java Sun Microsystems	10	 Java Sun Microsystems				

Table 5.2: Predicted logos-trademarks.

Eleven (11) images were excluded from the initial set of table 5.1. These images are (using table 5.1 numerations): 1, 2, 3, 4, 5, 6, 7, 16, 17, 20 and 21. Except image 17 none of the others are logo-trademarks, therefore the tree decided well and excluded them. However it included into the logos-trademarks set, simple graphic images such as (using table 5.2 numeration): 2.

Sometimes it also includes buttons. Although the training data set contained similar images as negative examples (non-logo images), it is very difficult for the tree to exclude them, as their properties are alike to logo images: they have no too many intensity levels and colors and exhibit low spatial distribution of intensities and colors. However, we can exclude button images either manually or by filtering out images with file-names referring to buttons (e.g., “button.gif”) or by excluding images with very small file size (a method we utilized) or by using a gazetteer (catalog) of the most common button images: Every time a new button image is found it is added in the catalog, and every time such an image is found in the Web site it excluded from the input of the decision tree (images can be compared pixel by pixel).

5.1.2 Logo and Trademark Similarity

The process described at section 3 was used. The features of section 3.1 were extracted for all the pairs of images and a data-set for the decision tree was formed. The tree returned the predicted similar pairs of images. The cliques’ algorithm (section 3.4.2) was used to group similar pairs into clusters. Table 5.3 illustrates the four (4) groups.

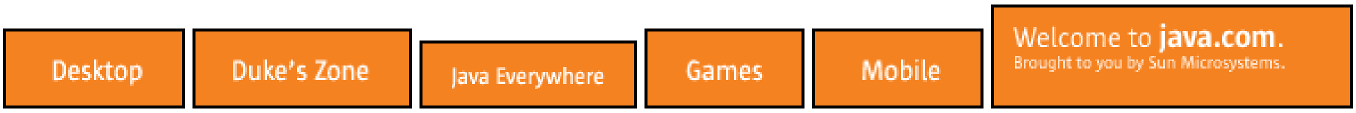
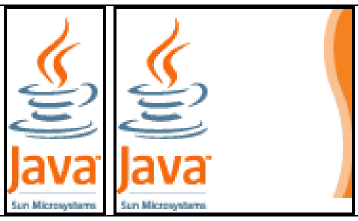
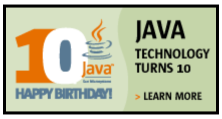
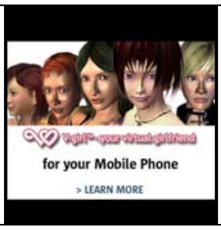
Group 1	 Desktop Duke's Zone Java Everywhere Games Mobile Welcome to java.com. Brought to you by Sun Microsystems.
Group 2	
Group 3	 10 Java HAPPY BIRTHDAY! JAVA TECHNOLOGY TURNS 10 > LEARN MORE
Group 4	

Table 5.3: Clusters of images.

The algorithm grouped the 10 images into 4 clusters. When clustering, there are two improper situations: a cluster to contain images that are not similar or similar images to be on different clusters. The first leads to *information loss*: only one image from each cluster is selected at the final step. On the other hand the second leads to *information repetition*: the same (or similar) image will be selected more than once at the final step.

For this web site the program worked perfect and none of the above situations happened. However there are cases that the program does not cluster images so well. Section 5.2 contains such an example.

5.1.3 Logo and Trademark Ranking

The final step is Logo and Trademark ranking. The method we used is that of section 4.1.1. All the images are sorted by importance and the most characteristic-important from each cluster is selected. Table 5.4 illustrates the 4 more characteristics images, ordered by importance:

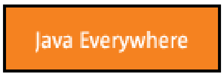
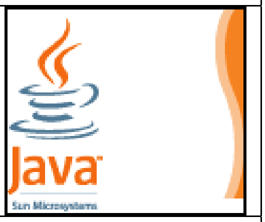
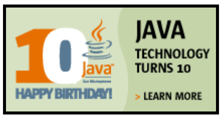
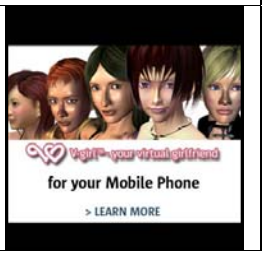
Image 1	
Image 2	
Image 3	 10 Java HAPPY BIRTHDAY! JAVA TECHNOLOGY TURNS 10 > LEARN MORE
Image 4	 V-girl™ - your virtual girlfriend for your Mobile Phone > LEARN MORE

Table 5.4: Final outcome.

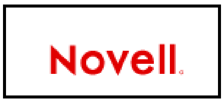
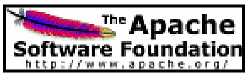
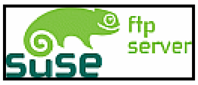
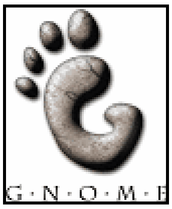
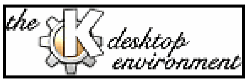
The image importance is based on its backlinks, its position at the web-site tree and its probability to be a logo-trademark.

5.2. 2nd Web Site: www.suse.com

The following experiment uses images from suse web site.

5.2.1 Logo and Trademark Extraction

Table 5.5 contains images from www.suse.com. Again notice that it contains only 62 images.

1	□	2	▯	3		4	
5		6		7		8	
9		10		11		12	››
13	■	14	□	15	■	16	››
17	››	18	››	19	››	20	▯
21	□	22	□	23	▯	24	□
25	□	26		27	■	28	
29	■	30	💾	31		32	
33		34		35		36	

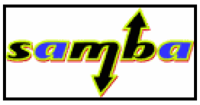
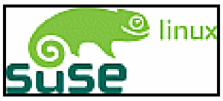
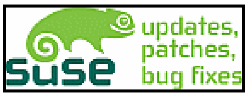
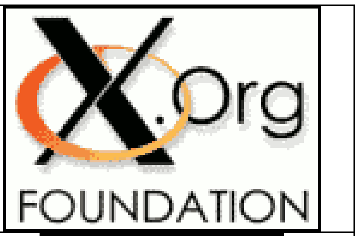
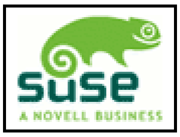
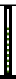
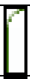
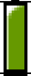
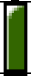
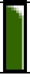
37		38		39		40	
41		42		43		44	
45		46		47		48	
49		50		51		52	
53		54		55		56	
57		58		59		60	
61		62					

Table 5.5: Images from web-site.

Again for the next step the histogram features process, described at section 2 was used. The features of section 2.1.2 were derived for each of the above images. These features formed a data-set for the decision tree, which predicted the following images as logos-trademarks:

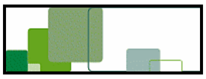
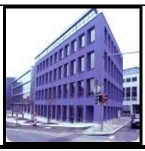
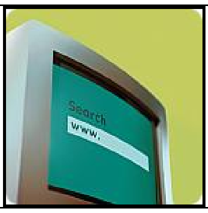
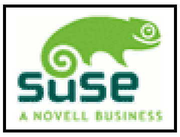
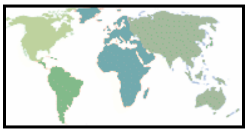
1		2		3		4	
5	The Apache Software Foundation http://www.apache.org/	6	SuSE ftp server	7		8	the desktop environment
9		10		11	SuSE updates, patches, bug fixes	12	X.Org FOUNDATION
13		14		15			

Table 5.6: Predicted logos-trademarks.

Forty four (47) images were excluded from the initial set of table 5.5. None of these were logos-trademarks, therefore the tree decided well and excluded them. However it included into the logos-trademarks set, two photos (3, 13) and simple graphic images (1, 2, 15). These graphic images could be logos (especially 1 and 15) and it is very difficult for the tree to exclude them. Both photos (3, 13) have no too many intensity levels and colors (especially 13) and exhibit low spatial distribution of intensities.

5.2.2 Logo and Trademark Similarity

The process described at section 3 was used. Features of section 3.1 were extracted for all the pairs of images and a data-set for the decision tree was formed. The tree returned the predicted similar pairs of images. The cliques' algorithm (section 3.4.2) was used to group similar pairs into clusters. Table 5.7 illustrates the ten (10) groups.

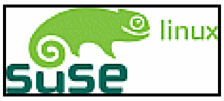
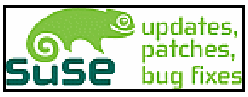
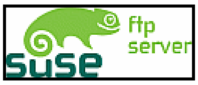
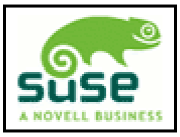
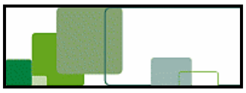
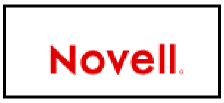
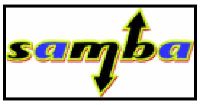
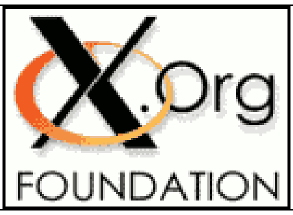
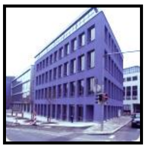
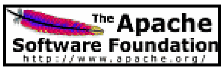
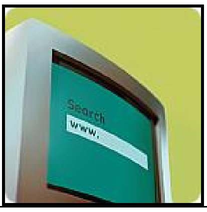
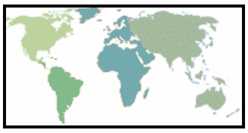
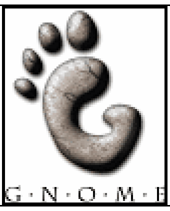
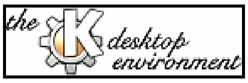
Group 1	    
Group 2	
Group 3	
Group 4	
Group 5	
Group 6	
Group 7	
Group 8	
Group 9	
Group 10	

Table 5.7: Clusters of images.

The algorithm grouped the 15 images into 10 clusters. In general the algorithm performed well. However two of the clusters are improper: The first and the second group contain images that belong to different clusters. The first contains five images four of which are similar. The second contains two, not similar images. This will lead to information loss: only one image from each cluster will be selected at the final step. We should also mention that similar images were grouped together, so information repetition will not happen at the next step.

5.2.3 Logo and Trademark Ranking

The final step is Logo and Trademark ranking. The method we used is that of section 4.1.1. All the images are sorted by importance and the most characteristic-important from each cluster is selected. Table 5.8 illustrates the 10 more characteristics images, ordered by importance.

Image 1		Image 6	
Image 2		Image 7	the desktop environment
Image 3		Image 8	
Image 4		Image 9	X.Org FOUNDATION
Image 5	The Apache Software Foundation http://www.apache.org/	Image 10	

Table 5.8: Final outcome.

Table 5.8 contains no characteristic logo of suse web site. The first reason is that the program includes separate steps, each of which uses the previous' step output as input: a mistake at the first step may continue and even grow up at the final steps. Image 1 (using table 5.6 enumerator) was mistakenly considered as logo at first step (although it could be). At the second step it was improperly grouped with the four most characteristic images of the web site and at the third step it was selected as the most characteristic image of the cluster.

The other reason is that the program communicates with a database that contains parts of web sites. This has two effects. The first is that images which could be characteristic of the web site are not included at the initial set. The program is bounded at a small portion of web site's images. It also affects the images importance computation (image backlinks, image depth) as pages are missing from the database.

6. Conclusion – Future work

A Web site summarization method focusing on image content is presented and discussed. We chose the problem of logo and trademark images as a case study for the evaluation of the proposed methodology. The problem of logo and trademark extraction (or Web site summarization by Logo and trademark extraction), is of significant commercial interest (e.g., ImageLock www.imagelock.com provides services on unauthorized uses of logos and trademarks) and this technology can benefit from the proposed approach. Extending the proposed methodology to handle any other image type is straightforward (i.e., the algorithms for logo and trademark selection, description and matching can be replaced by algorithms for the new image type).

The method works into steps, first by extracting images with high probability of being logos or trademarks, by clustering similar images together and by ranking (by importance) the images in each cluster. The most important image from each cluster is included in the summary.

Logo and trademark extraction and clustering are based on feature extraction and on machine learning by decision trees. Various features for describing the content of images of this type has been tested including low level intensity features as well as features defined at the low to intermediate level such as features computed on intensity and frequency histograms and moments invariants. Image importance scores are finally computed to images belonging to the same clusters for the purpose of selecting the most characteristic image from each cluster and finally, the most characteristic logo and trademark images of a Web site a whole. A prototype web summarization system for logo and trademarks is also implemented as part of this work.-

The experimental results demonstrated that the method handles logo and trademark images successfully in most cases and manages to extract the most characteristic images of this type from even from large corporate Web sites.

Future work includes experimentation with larger training data sets and image types for improving the performance learning for logo and trademark detection and image clustering. More elaborate features for image content analysis may also be utilized, (e.g., by applying the same analysis on all parts of the same image which are produced by fitting a grid on the image). Further improvements might also be achieved in detecting clusters of similar images by relaxing the requirement of perfect cliques and by allowing for not fully connect components in cliques. Finally, future work also includes combination of text summarization tools with the proposed method for the purpose of developing a fully automated text-image Web site summarization system.

References

1. Euripides G.M. Petrakis, E.V., Evangelos Milios, *Weighted Link Analysis for Logo and Trademark Image Retrieval on the Web. .*
2. J.Sammon, M.S.L.O.G.M., *2. Global Image Analysis*, in *Practical Algorithms For Image Analysis. Description, Examples, and Code*, C.U. Press, Editor. 2000, The Press Syndicate Of The University Of Cambridge: Cambridge. p. 21-37.
3. Borko Furht, S.W.S., HongJiang Zhang, *4.3 Image Concepts And Structures*, in *Video And Image Processing In Multimidia Systems*, B. Furht, Editor. 1995, Kluwer Academic Publicers. p. 98-99.
4. J.Sammon, M.S.L.O.G.M., *7. Frequency Domain Analysis*, in *Practical Algorithms For Image Analysis. Description, Examples, and Code*, C.U. Press, Editor. 2000, The Press Syndicate Of The University Of Cambridge: Cambridge. p. 246-264.
5. J.Sammon, M.S.L.O.G.M., *3. Gray -Scale Image Analysis*, in *Practical Algorithms For Image Analysis. Description, Examples, and Code*, C.U. Press, Editor. 2000, The Press Syndicate Of The University Of Cambridge: Cambridge. p. 110-117.
6. Flannery, W.H.P.W.T.V.S.A.T.B.P., *14. Statistical Description Of Data*, in *Numerical Recipes in C++ : The Art of Scientific Computing*. 2003. p. 616 - 617.
7. Milan Sonka, V.H., Roger Boyle, *14. Texture*, in *Image Processing, Analysis, and Machine Vision*, K. McGee, Editor. 1999, PWS. p. 646-653.
8. Ian H. Witten, E.F., *3.2 Decision Trees*, in *Data Mining : Practical Machine Learning Tools and Techniques with Java Implementations*, D.D. Cerra, Editor. 2000, Academic Press. p. 58-63.
9. Ian H. Witten, E.F., *4.3 Devide and conquer: Constructing decision trees*, in *Data Mining : Practical Machine Learning Tools and Techniques with Java Implementations*, D.D. Cerra, Editor. 2000, Academic Press. p. 89-97.
10. Ian H. Witten, E.F., *6.1 Decision Tress*, in *Data Mining : Practical Machine Learning Tools and Techniques with Java Implementations*, D.D. Cerra, Editor. 2000, Academic Press. p. 159-170.
11. Ian H. Witten, E.F., *5.3 Cross-validation*, in *Data Mining : Practical Machine Learning Tools and Techniques with Java Implementations*, D.D. Cerra, Editor. 2000, Academic Press. p. 125-127.
12. Flannery, W.H.P.W.T.V.S.A.T.B.P., *2.6 Singular Value Decomposition*, in *Numerical Recipes in C++ : The Art of Scientific Computing*. 2003. p. 62-73.
13. Borko Furht, S.W.S., HongJiang Zhang, *11.2 Image Features For Content - Based Retrieval*, in *Video And Image Processing In Multimidia Systems*, B. Furht, Editor. 1995, Kluwer Academic Publicers. p. 230-232.
14. Milan Sonka, V.H., Roger Boyle, *6.3 Region-based shape representation and description*, in *Image Processing, Analysis, and Machine Vision*, K. McGee, Editor. 1999, PWS. p. 259-262.

15. Milan Sonka, V.H., Roger Boyle, *7.5 Recognition as graph matching*, in *Image Processing, Analysis, and Machine Vision*, K. McGee, Editor. 1999, PWS. p. 323-328.

16. Ian H. Witten, E.F., *2.4 Preparing the input*, in *Data Mining : Practical Machine Learning Tools and Techniques with Java Implementations*, D.D. Cerra, Editor. 2000, Academic Press. p. 49-50.