Detecting image purpose in World-Wide Web documents

Seungyup Paek

Center for Telecommunications Research
Columbia University
New York, N.Y. 10027-6699

John R. Smith

IBM T.J. Watson Research Center
30 Saw Mill River Road
Hawthorne, NY 10532

ABSTRACT

The number of World-Wide Web (WWW) documents available to users of the Internet is growing at an incredible rate. Therefore, it is becoming increasingly important to develop systems that aid users in searching, filtering, and retrieving information from the Internet. Currently, only a few prototype systems catalog and index *images* in Web documents. To greatly improve the cataloging and indexing of images on the Web, we have developed a prototype rule-based system that detects the *content* images in Web documents. Content images are images that are associated with the main content of Web documents, as opposed to a multitude of other images that exist in Web documents for different purposes, such as decorative, advertisement and logo images. We present a system that uses decision tree learning for automated rule induction for the content image detection system. The system uses visual features, text-related features and the document context of images *in concert* for fast and effective content image detection in Web documents. We have evaluated the system by collecting more than 1200 images from 4 different Web sites and we have achieved an overall classification accuracy of 84%.

Keywords: image, image classification, image catalog, image indexing, search engine, Web documents, WWW.

1. INTRODUCTION

The number of World-Wide Web (WWW) documents available to users of the Internet is growing at an incredible rate. The Web is becoming the primary mechanism for disseminating a diversity of information, which includes news, technical publications, business and entertainment information, scientific data, and personal communications. In light of the exponentially increasing amount of information accessible to users on the Internet, it is becoming extremely useful to develop systems that search, filter, and retrieve information from the Internet. Web search engines, which in turn rely on systems that catalog and index Web documents are one example of such systems.

Although the Web consists entirely of digital information, automating the understanding, cataloging and indexing of Web content remains a great challenge. Part of the challenge comes from the fact that Web documents typically consist of information of multiple modalities i.e., text, images, audio and video. Recently, images have been shown to form up to 70% of Internet traffic.[5] However, although many text-based systems exist for cataloging and indexing Web documents by textual content (Alta Vista, Excite), only a few prototype systems catalog and index the images in Web documents[8].[3] Therefore, a large portion of Web content is beyond the reach of today's Web search engines.

In Web documents, images are used for a variety of purposes such as navigation (image maps), decoration (bullets, backgrounds), advertisements and text-related *content* images. Table 1 describes some examples of images with different purposes that are found in Web documents. To improve the automated understanding, cataloging and indexing of images on the Web, we have developed a prototype system based on learning through user-interaction that detects *content* images in Web documents. Content images refer to images that are associated with the main content of Web documents. The system uses the visual features, text-related features and the document context of

Advertisement images
Content images (e.g. images associated with a body of text)
Decorative images (e.g. buttons, balls, rules, masthead)
Informational images (e.g. under construction, warnings, what's new)
Logo (e.g. "IBM" corporate logo)
Navigation images (e.g. arrows, back to home, image maps)

Table 1. Examples of different image purposes in Web documents

images *in concert* to achieve content image detection in Web documents. The content image detection system can be used to improve the cataloging and indexing of images on the Web by filtering out the content images from the multitude of other images that are found in a typical Web document, such as decorative graphics and advertisement images. It is important to note that although the system is specifically developed to detect content images in Web documents, the same framework can equally be used to detect images of other purposes in Web documents. The system consists of three main components (Figure 1). Each of these components is discussed in greater depth in sections 2 and 3.

- Web retrieval, parsing and extraction: The system automatically performs Web retrieval, parsing and extraction. The system establishes a network connection with a Web server and then retrieves a Web document and its associated images from the Web server. After the document is retrieved, it is parsed and various information associated with each image is extracted. The various information extracted from a Web document for each image are used to form a set of *image objects* (Table 2). The retrieval parsing and extraction provides the mapping between a Web document and a set of image objects. The image objects are the inputs to the content image detection system.
- Automatic rule-induction: The heart of the system is a classification system based on decision tree learning for automatic rule induction. For decision tree learning, large numbers of images are collected from different Web sites using a Web spider that automatically traverses the Web and collects information. The images are manually classified as content images or non-content images (Table 1). The collected and classified images are input into a decision tree learning algorithm which automatically finds a set of rules for the detection of content images.
- Rule-based detection: Content images in Web documents are detected by using the rules found in steps 1 and 2. The rules use the visual features, document context and text-related features associated with the image in concert to detect content images.

For the content image detection system, we have chosen a decision tree based approach since it is efficient, and therefore can deal with large training data sets. In addition, the final classifier produced is symbolic and can therefore be interpreted by a Web site 'domain expert'. This is in contrast to a neural network approach or a pattern recognition based approach. In these approaches, the result of training cannot be directly interpreted.[1]

The decision tree learning algorithm automatically finds a set of rules for content image detection. Since the data input into the learning algorithm is inherently noisy, a *rule pruning* algorithm was also used to avoid overfitting the data. Rule pruning leads to a set of rules with a smaller number of boolean tests, with improved accuracy in classification.

In section 4, we show the performance of the system in automatically detecting content images in different Web sites. We conclude in section 5 and briefly discuss future research to extend and build on this work.

2. WEB DOCUMENT RETRIEVAL AND PARSING

The content image detection system automatically performs Web retrieval, parsing and extraction. The input to the detection system is a Universal Resource Locator (*URL*) for a specific Web document. The system establishes a *TCP/IP* network connection with the Web server specified by the *URL* and then performs a Hypertext Transfer Protocol (*HTTP*) retrieval of the Web document. The file that is retrieved contains Hypertext Markup Language

(*HTML*) code for the Web document. The retrieval process is explained in detail in Ref. 11. After the *HTML* code is retrieved, the code is parsed to extract information from the *HTML* code.

The *HTML* code typically contains a set of image *URLs* . These represent the images within a Web document. The image *URLs* are extracted from the *HTML* code, and another series of *HTTP* retrievals are performed for each of the image *URLs* in order to retrieve all the associated images of a given Web document. The images are retrieved so that they can be processed to extract various visual features. To inline or embed an image in a Web document, the following *HTML* code is included in the Web document:

```
<img src=URL alt=[alt text]>
```

The `URL` gives the relative or absolute address of the image. The optional `alt` tag specifies the text that subsitutes for the inlined image when the image is not displayed. In the *WebSEEk* system[8] it was shown that useful information can be obtained from the *URLs* and from the *HTML ALT* tags of images in Web documents. In our system, we parse *URLs* and *HTML ALT* tags of images to extract a set of *labels* associated with each of the images. The extracted labels are filtered and any words that occur in a library of *stop-words* (e.g. 'a', 'in', 'the', 'or') are removed. Furthermore, we check whether the labels extracted for each image (excluding stop-words) also occur in the text portions of the Web document. In Ref. 6, work has been done in which image captions are automatically found from text that surrounds images in Web documents. In our system, we only use image keywords extracted from the *URL* and *ALT* tag.

By parsing the *HTML* code of the Web document, it is also possible to extract information concerning the document context of each of the images. Such information can be obtained either directly from *HTML* tags, or indirectly by analyzing the *HTML* code.

The various information extracted from a Web document are used to form a set of *image objects* . The module for the retrieval parsing and extraction provides the mapping between a Web document and a set of image objects. The image objects are the inputs to the content image detection system. The information contained in an image object is shown in Table 2. The *label occurrences* in Table 2 refer to whether certain keywords occurred in the set of labels that were extracted for each image. For example, if any of the words {bullet, button, rule, line} occurred in the set of image labels, the *decorative label occurrence* (attribute id=17) was set to TRUE. The body text label occurrence was set to TRUE if any of the words extracted for an image (excluding stop-words) were found in the text portions of the Web document e.g. paragraphs, headers. For attributes with a continous attribute range, the attribute values were uniformly quantized into four discrete values for decision tree learning. The visual features were used in Ref. 9 to successfully detect the type of an image {color photo, complex graphic, simple graphic, gray photo, gray graphic, b/w photo, b/w graphic}.

3. CONTENT IMAGE DETECTION SYSTEM

The heart of the content image detection system is a rule-based classification system that is based on decision tree learning for automatic rule induction and rule-pruning for improved classification accuracy with noisy data.

For decision tree learning, large numbers of images were automatically collected from different Web sites using a Web spider that traverses the Web. For each image, all the information described in Table 2 were collected. All the images were manually classified as content images or non-content images (Table 1). Table 5 shows the number of images and data that were collected from different sites and used for training and testing the decision tree learning algorithm.

The collected images are input into a decision tree learning algorithm which automatically finds a set of rules for content image detection. Since the data input into the learning algorithm is inherently noisy, a *rule pruning* algorithm is used to avoid overfitting the data. This is explained in detail below. Rule pruning leads to a set of rules with a smaller number of boolean tests, with improved accuracy in classification. The set of rules are used in the detection system to determine whether an image is a content image.

3.1. Decision tree learning for automatic rule induction

In this section, we summarize the decision tree learning algorithm for automatic rule induction. The algorithm uses a heuristic based on information theory to find a small tree. The basic idea is to test the most important attributes first, in order to find the correct classification with a small number of tests. A detailed explanation is given in Ref. 7 and Ref. 4. A decision tree is learned for content image detection. Figure 2 shows the automatically created

Attribute id	Attribute class	Attribute description	Attribute range
0	Context	Image format {gif, jpg}	{0, 1}
1	Context	*HTML* image type	{0, 1, 2, 3}
2	Context	(image number)/(total number of images)	0.0-1.0
3	Visual	No. of colors	{0, 1,..Max. colors}
4	Visual	% Black	0.0-100.0
5	Visual	% Gray	0.0-100.0
6	Visual	% White	0.0-100.0
7	Visual	No. of grays	{0, 1,..Max. grays}
8	Visual	No. of hues	{0, 1,..Max. hues}
9	Visual	No. of saturation	{0, 1,..Max. sat.}
10	Visual	% Quarter saturation	0.0-100.0
11	Visual	% Half saturation	0.0-100.0
12	Visual	% Fully saturation	0.0-100.0
13	Visual	Image width	{1, 2,..Max. image width}
14	Visual	Image height	{1, 2,..Max. image height}
15	Text	Advertisement label occurrence	{0, 1}
16	Text	Body label occurrence	{0, 1}
17	Text	Decorative label occurrence	{0, 1}
18	Text	Informational label occurrence	{0, 1}
19	Text	Logo label occurrence	{0, 1}
20	Text	Navigation label occurrence	{0, 1}

Table 2. Image object

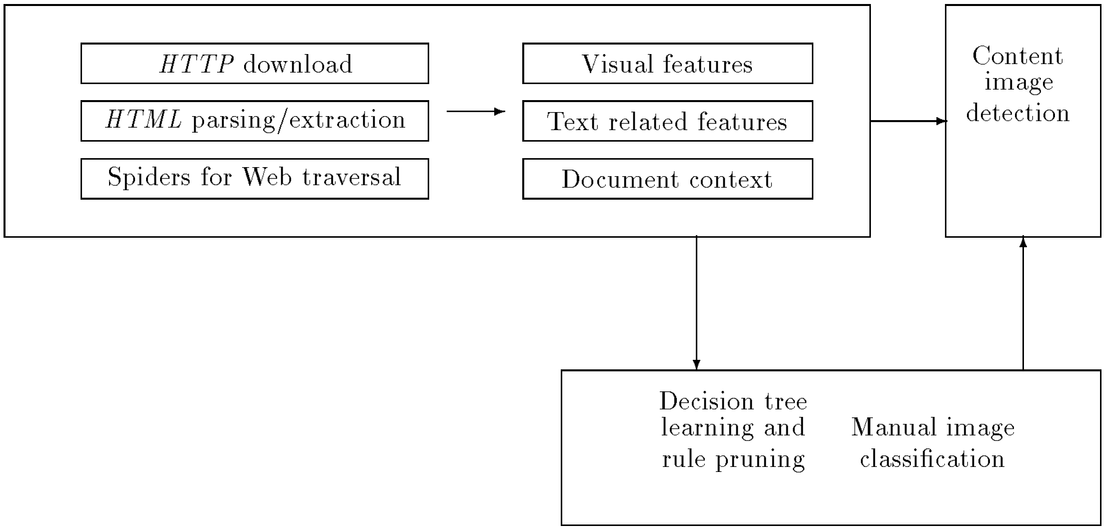

Figure 1. Content image detection system

decision tree for the *content* images in the CNN web site that was learned using training data. The numbers in the nodes correspond to a test on one of the attributes given in Table 2. The different edges leading out of each node correspond to different values that the attribute can take (the left-most node corresponds to the first value, and the right-most node corresponds to the last value). Note that different nodes have different numbers of edges that go out of them, depending on the attribute test that is associated with each node (e.g. nodes with attribute tests with boolean values have two edges that go out of them).

The procedure to find the decision tree is as follows:

- Given a training set of N examples, we divide the examples into positive and negative examples. Each example is an image object with all the attributes and attribute values described in Table 2. Each example has an associated manual classification which indicates whether the image is a content image or a non-content image. For all the examples, positive examples are those for which the images are content images. Negative examples are examples with images that are non-content images e.g. navigation, decorative, logo etc.
- For all the attributes, we decide which attribute to use as the first test in the tree. The importance of an attribute is measured by the *information gain* of an attribute. The information gain is defined using information theory and is described in detail in Ref. 7 and Ref. 4. In brief, the information gain of an attribute is expressed as the reduction in the information that is required to classify a set of examples when the value of an attribute for all the examples is known. Each attribute test divides the examples depending on the value the test attribute has in each example.
- After the first attribute divides the examples, each outcome is a new decision tree learning problem in itself, with fewer examples and one less attribute. This process is repeated recursively to form the decision tree.

The decision tree learning algorithm is very simple to use in the sense that we simply give the positive and negative examples and let the system do the rest. The system automatically picks the attribute ordering with the objective of minimizing the size of the decision tree.

In summary, the basic idea of the decision tree learning algorithm is to test the most important attributes first. The goal is to find the correct classification with a small number of tests, which means that all paths in the tree will be short and the tree as a whole will be small.

3.2. Rule pruning

The decision tree learning algorithm described above attempts to grow the tree deeply enough to classify the training examples. Although this is a reasonable strategy, it can lead to difficulties when there is noise in the data, or when the number of training examples is too small. In such cases, the above simple algorithm can produce trees that *overfit* the training examples.

A certain hypothesis *overfits* the training examples if some other hypothesis that fits the training examples less well actually performs better over the entire distribution of instances, incuding the testing data. To illustrate this, consider the effect of adding the positive training example shown in equation (1), incorrectly labeled as a negative example, to an otherwise correct tree with a path shown in equation (2).

$$Example :< Attribute0 = TRUE, Attribute1 = FALSE > Classification = FALSE \tag{1}$$

$$Path :< Attribute0 = TRUE, Attribute1 = FALSE > Classification = TRUE \tag{2}$$

The addition of this incorrect example will now lead to a more complex tree. The new tree will fit the set of training examples, however, we would expect the simpler tree to perform better over a set of testing data. In order to solve this problem, we used a *rule post-pruning* algorithm that prunes a learned decision tree to avoid overfitting the data. The details of the rule post-pruning algorithm are given in Ref. 7 and Ref. 4. We summarize the rule post-pruning algorithm below:

- Create a decision tree using the decision tree learning algorithm.

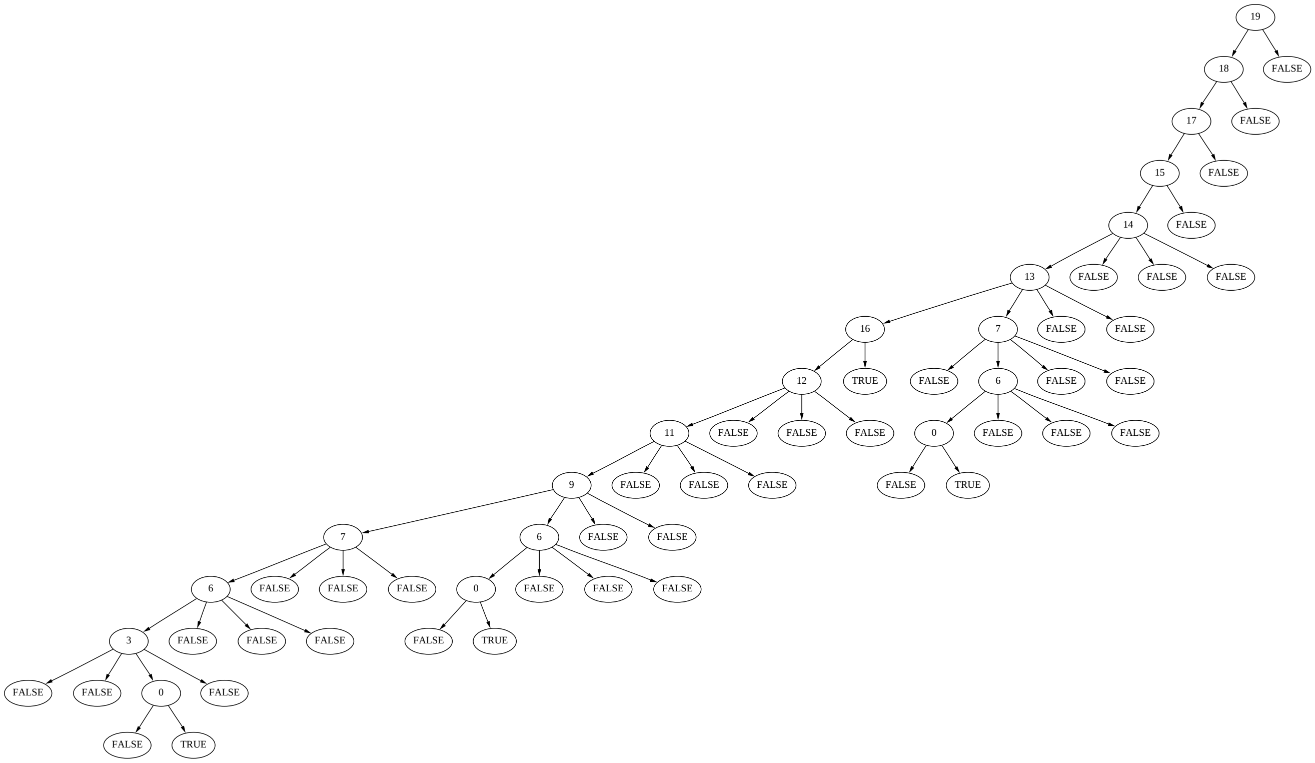

Figure 2. Automatically generated decision tree for CNN Web site

Rule no.	Attribute values	Goal
0	a19=0 a18=0 a16=0 a15=0 a14=0 a13=0 a12=0 a11=0 a9=0 a7=0 a6=0 a3=0	g=0
1	a19=0 a18=0 a16=0 a15=0 a14=0 a13=0 a12=0 a11=0 a9=0 a7=0 a6=0 a3=1	g=0
2	a19=0 a18=0 a16=0 a15=0 a14=0 a13=0 a12=0 a11=0 a9=0 a7=0 a6=0 a3=2	g=0
3	a19=0 a18=0 a16=0 a15=0 a14=0 a13=0 a12=0 a11=0 a9=0 a7=0 a6=0 a3=3	g=1
.	.	.
.	.	.
55	a19=0 a18=0 a16=0 a15=0 a14=0 a13=1 a17=0 a7=0 a5=0 a3=2 a0=0	g=1

Table 3. Unpruned rules for CNN Web site

- Convert the learned tree into a set of rules by creating one rule for each path from the root node to a leaf node.
- Prune each rule by removing any pre-conditions that result in improving the estimated accuracy of the rule.
- Sort the pruned rules by their estimated accuracy, and consider them in this sequence when classifying subsequent examples.

For the automatically created decision tree using training data from the CNN Web site, the first few sorted, un-pruned rules and sorted, pruned rules are shown in Table 3, Table 4.

4. EVALUATION

Table 6 shows the results for the evaluation of the content image detection system. For evaluation, images were automatically collected from different Web sites using a Web spider that traverses the Web. For each image, all the information described in Table 2 were collected. All the images were manually classified as content images or

Rule no.	Attribute values	Goal
0	a9=0 a6=0 a3=1	g=0
1	a13=0 a9=0 a7=0 a6=0 a3=2	g=0
2	a9=0 a7=0 a3=3	g=1
.	.	.
.	.	.
55	a18=0 a16=0 a15=3	g=1

Table 4. Pruned rules using rule post-pruning for CNN Web site

Web site URL	Key	Total no. of images	Content images
`http://www.cnn.com`	`CNN`	312	94
`http://www.music.sony.com`	`SONY`	214	48
`http://www.apple.com`	`APPLE`	583	76
`http://www.calif.gov`	`CALIF`	134	14
Total	4	1243	232

Table 5. Decision tree training and testing data

non-content images (Table 1). Table 5 shows the number of images and data that were collected from different sites and and used for testing the content image detection system.

The negative detection accuracy was excellent. The detection system detected the non-content images with an average detection accuracy of 90%. That is, if an image is a non-content image, it is highly unlikely to be misclassified as a content image. The positive detection accuracy was much lower. The detection system detected the content images with an average detection accuracy of 48%. The detection system detects approximately half of the content images. The overall detection accuracy is 84%.

The detection system can be significantly improved by using more extensive and sophisticated visual features and text-related features, using automatic caption localization and extracting more document context information. Furthermore, using more extensive training data can improve the detection system performance.

The prototype detection system is very useful in filtering out the content images from the multitude of non-content images found in typical Web documents.

5. CONCLUSIONS AND FUTURE RESEARCH

In Web documents, images are used for a variety of purposes such as navigation (image maps), decoration (bullets, backgrounds), advertisements and text-related *content* images. To improve the automated understanding, cataloging and indexing of images on the Web, we have developed a prototype system based on learning through user-interaction that detects *content* images in Web documents. The system uses the visual features, text-related features and the document context of images *in concert* to achieve content image detection in Web documents. The content image detection system can be used to improve the cataloging and indexing of images on the Web by filtering out the content images from the multitude of other images that are found in a typical Web document, such as decorative

Web site	`CNN`	`SONY`	`APPLE`	`CALIF`
Overall detection accuracy before/after pruning	82%/86 %	63%/67%	87%/88%	91%/93%
Positive detection accuracy before/after pruning	81%/74 %	42%/42%	45%/34%	43%/43%
Negative detection accuracy before/after pruning	82%/91 %	69%/74%	94%/96%	97%/98%
No. of rules	56	53	55	21
No. of conditions before/after pruning	550/180	380/141	628/198	156/58

Table 6. Detection performance

graphics and advertisement images. Although the system is specifically developed to detect content images in Web documents, the same framework can equally be used to detect images of other purposes in Web documents.

The heart of the system is a classification system based on decision tree learning for automatic rule induction. For decision tree learning, large numbers of images are collected from different Web sites and manually classified as content images or non-content images. The collected and classified images are input into a decision tree learning algorithm which automatically finds a set of rules for the detection of content images.

For the content image detection system, we have chosen a decision tree based approach since it is efficient, and therefore can deal with large training data sets. The final classifier produced is symbolic and can therefore be interpreted by a Web site 'domain expert'.

In this paper, the primary motivation given for the content image detection system was to improve the performance of systems that catalog and index images in Web documents, which are used to facilitate image search engines for the Web. However, another exciting area in which this system can be applied is in systems that transcode Internet content for heterogenous client devices.[10] For future research we are seeking to apply the content image detection system to boost the performance of transcoding systems for the Internet.

REFERENCES

1. U.M. Fayyad, S.G. Djorgovski, and Nicholas Weir. *From Digitized Images to Online Catalogs. Data Mining a Sky Survey.* AI Magazine, American Association for Artificial Intelligence (AAAI), Summer 1996.
2. A. Fox and E. A. Brewer. *Reducing WWW latency and bandwidth requirements by real-time distillation.* Proceedings of the International World Wide Web Conference. Paris, France. May 1996.
3. C. Frankel, M. Swain and V. Athitsos. *WebSeer: An Image Search Engine for the World Wide Web.* University of Chicago Technical Report TR-96-14. July, 1996.
4. T. M. Mitchell. *Machine Learning* McGraw Hill. 1997.
5. A. Ortega, F. Carignano, S. Ayer, M. Vetterli. *Soft Caching: Web Cache Management Techniques for Images.* IEEE Signal Processing Society 1997 Workshop on Multimedia Signal Processing. Princeton, New Jersey, USA. June, 1997.
6. N.C. Rowe, B. Frew. *Automatic Caption Localization for Photographs on World Wide Web Pages.* To appear in Information Processing and Management. 1998.
7. S. J. Russell and P. Norvig. *Artificial Intelligence: A Modern Approach.* Prentice Hall. 1995.
8. J. R. Smith and S.-F. Chang. *Visually searching the Web for content.* IEEE Multimedia. July - September, 1997, Vol. 4, No. 3, pp. 12 – 20.
9. J. R. Smith and S.-F. Chang. *Multi-stage Classification of Images from Features and Related Text.* Proc. Fourth DELOS workshop, Pisa, Italy, August, 1997.
10. J. R. Smith, R. Mohan and C.-S. Li. *Transcoding Internet Content for Heterogenous Client Devices.* To appear in IEEE Inter. Conf. on Circuits and Systems (ISCAS-98), Special session on Next Generation Internet, June, 1998.
11. N.J. Yeager, R.E. McGrath. *WWW Server Technology.* Morgan Kaufmann Publishers. Inc. 1996.