Distinguishing Photographs and Graphics on the World Wide Web

Vassilis Athitsos, Michael J. Swain and Charles Frankel
Department of Computer Science
The University of Chicago
Chicago, Illinois 60637
{ vassilis, swain, frankel }@cs.uchicago.edu

Abstract

When we search for images in multimedia documents, we often have in mind specific image types that we are interested in; examples are photographs, graphics, maps, cartoons, portraits of people, and so on. This paper describes an automated system that classifies Web images as photographs or graphics, based on their content. The system first submits the images into some tests, which look at the image content, and then feeds the results of those tests into a classifier. The classifier is built using learning techniques, which take advantage of the vast amount of training data that is available on the Web. Text associated with an image can be used to further improve the accuracy of the classification. The system is used as a part of WebSeer, an image search engine for the Web.

1 Introduction

Collections of multimedia documents can contain a vast amount of textual and visual information. However, the bigger the size of such collections grows, the harder it gets to locate specific information in them. We can use indexing programs, which go through a collection and classify documents and multimedia components, such as images or videos, based on the information they can extract from them. The problem is that current computer programs can extract much less information from those components than humans can. For images in particular, it is trivial for a human to look at a picture of a dog, locate the dog, and index the image under the word "dog". It is beyond current computer vision technology to make a program that does the same thing.

Images embedded in multimedia documents have text associated with them. That text often contains words that describe the content of the images. Indexers make use of that fact and index images based on the text around them. However, when we look for images, we usually have more in mind than just some keywords; we want a specific type of images that are associated with those words. For example, we may want to find photographs of dogs, portraits of the president, maps of Europe, help buttons, or inflation charts. Programs that can classify images as photographs, portraits, maps, buttons, charts, or several other types, make it considerably easier for people to specify and get back the kind of images they are interested in.

The photo detector we use in WebSeer classifies images as photographs or computer-generated graphics. It is an example of a program that can extract a small but valuable piece of information from an image. The detector relies primarily on the image content to do the classification. Preliminary results show that the document that contains the image is also a useful source of information.

2 Related Work

Up to now there have been very few efforts to automate the classification of images as photographs and graphics. The WebSeek search engine [2] performs that classification based on information obtained from the color histograms of the images. The system described in [3] uses some information from the image content, as well as information from the image context, that is the HTML document in which the image is embedded. The image content clues that are used are the squareness of the image, the number of colors, the fraction of impure colors (colors that are not pure white, black, grey, red, green or blue), the neighbor variation (fraction of horizontally-adjacent pixels of the same color) and the color dispersion (fractional distance of the mean color in the sorted color histogram). In addition, the filename portion of the image URL is tested for the occurrence of words that are usually associated with only one of the two image types. The existence or not of such words is an additional feature that is considered in the classification.

One problem with [2] and [3] is that it is hard to evaluate their accuracy. In [2] the authors claim a recall rate of 0.914 for Web photographs and 0.923 for Web graphics. However, they don't specify exactly what they consider photographs and graphics. We see later in the paper that we can define those image types in different ways, and our definitions have a direct impact on the error rate. The authors of [3] also don't specify exactly what they consider photographs and graphics. Consequently, we refrain from

comparing our error rates to the error rates attained in those systems, since we may have used different definitions for the two image types.

3 An Overview of the System

The photo detector submits images to several different tests. Those tests originate from a few statistical observations about the differences between computer-generated graphics and photographs that appear on the Web. In general, photographs and graphics differ in shape, size, the colors they use, and the pattern of color transitions from pixel to pixel.

Based on those observations, we have created several image metrics. The metrics are functions from images to real numbers. For those numbers we use the term "metric scores". A simple metric is the number of colors in an image. Our goal is to design metrics in which graphics tend to score in different ranges than photographs. This way, we can use the metric scores to decide if an image is a photograph or a graphic.

The scores we obtain from individual metrics are rarely definitive. In order to achieve high accuracy rates, we have to combine scores from several metrics when we make the final decision. The system uses learning to create decision trees, which specify how to classify an image based on its metric scores. The trees are constructed in an automated way, using the metric scores of large sets of images, which we randomly choose and download from the Web, and which we pre-classify by hand as photographs or graphics.

4 The Basic Assumptions

This section talks about the assumptions underlying the design of our system: What are photographs and graphics, and how they differ from each other.

4.1 What are Photographs and Graphics

We use the word "graphics" for computer generated images. For most images a human has no trouble deciding if they are photographs or graphics. Our goal for the system is obviously to classify those images the same way a human would. However, some times it is not clear whether an image should be considered a photograph or a graphic, and some times none of the two categories is applicable:

- Mixed images. A significant fraction of Web images have both a photograph and a computer-generated part. Examples are photographs with a frame around them, photographs with text overlaid on them, and images that are half photographs and half graphics.

- Hand drawings. Hand drawings are clearly not computer-generated graphics. However, even when the images are actually photographs of drawings, we don't consider them to belong to the "photograph" type.

Figure 1: An example of a photograph.

Figure 2: An example of a graphic.

The system is not designed to handle such cases in a consistent way, and images falling into those categories were not used for training or testing.

4.2 Differences between Photographs and Graphics

By looking at many photographs and graphics, one can easily notice certain basic differences between them, that are easy to describe in quantitative terms. These are the differences we used as a starting point in the design of our metrics:

- Color transitions from pixel to pixel follow different patterns in photographs and graphics. Photographs depict objects of the real world, and regions of constant color are not common in the real world, because objects tend to have texture. (Figure 1). In addition, photographs of objects always contain a certain amount of noise, that causes even nearby pixels to have different RGB values. On the other hand, graphics tend to have regions of constant color. Figure 2 is a typical example. The image has only 8 different colors, and most of the pixels have the same color as their neighbors.

On the other hand, edges in graphics tend to be much sharper. Typically an edge occurs between a region of constant color and another region of constant color, and the transition takes place over one pixel. In photographs, boundaries between objects are often blurred because the camera is not focused precisely on them. In addition, many color transitions do not correspond to boundaries between objects, but to light variations and shading. Such transitions are much smoother.

- Certain colors are much more likely to appear in graphics than in photographs. For example, graphics often have large regions covered with highly saturated colors. Those colors are much less frequent in photographs.

- Graphics have fewer colors than photographs. This is related to the fact that they tend to have large one-color regions. On the Web in particular, people often prefer to use graphics with a small number of colors, because they compress better.

- Graphics tend to have different shapes than photographs. They are often narrow, much longer in one dimension than in the other. Photographs tend to be more square. In addition, graphics frequently come in small sizes, which are very rare for photographs.

5 Image Metrics

To implement a photo detector, we need precise tests, which we can apply to an image and get back results that give us information about the type of the image. Based on the general observations we have described in the previous section, we have implemented several metrics, which map images to real numbers. Photographs and graphics tend to score in different ranges in those metrics. Because of that, the metric scores are evidence that we can use to differentiate between those two types.

In the following discussion, we assume that an image is represented by three two-dimensional arrays, each array corresponding to the red (R), green (G) and blue (B) color band of the image respectively. The entries of those arrays are integers from 0 to 255. The color vector of a pixel p is defined to be (r, g, b), where r, g and b are respectively the red, green and blue component of the color of the pixel.

The metrics we use are the following:

- The number of colors. The score of the image in this metric is the number of distinct colors that appear in it.

- The prevalent color metric. We find the most frequently occuring color in the image. The score of the image is the fraction of pixels that have that color.

- The farthest neighbor metric: For two pixels p and p', with color vectors (r, g, b) and (r', g', b') respectively, we define their color distance d as $d = |r - r'| + |g - g'| + |b - b'|$. Since color values range from 0 to 255, d ranges from 0 to 765. Each pixel p_1 (except for the outer pixels) has neighbors up, down, left and right. A neighbor p_2 of p_1 is considered to be the farthest neighbor of p_1 if the color distance between p_1 and p_2 is not smaller than the color distance between p_1 and any other of its neighbors. We define the transition value of p_1 to be the distance between p_1 and its farthest neighbor.

  In tne farthest neighbor metric, we have to specify a parameter P between 0 and 765. The score of the image is the fraction of pixels that have a transition value greater than or equal to P.

  We use a second version of the same metric to accentuate the difference in scores between graphics and photographs for high values of P. In the second version, the score of an image is the fraction f_1 of pixels with transition value greater than or equal to P, divided by the fraction f_2 of pixels with transition value greater than 0. Graphics have even higher scores with respect to photographs than they do in the first version, because f_2 tends to be larger for photographs.

- The saturation metric. For a pixel p, with color vector (r, g, b), let m be the maximum and n be the minimum among r, g and b. We define the saturation level of p to be $|m - n|$.

  We specify a parameter P. The score of the image is the fraction of pixels with saturation levels greater than or equal to P. For high values of P we expect graphics to score higher than photographs, since saturated colors occur more frequently in graphics.

- The color histogram metric. We create an average color histogram for graphics, and one for photographs. The score of the image depends on its correlation with the two histograms.

  A color histogram is a three dimensional table of size $16 \times 16 \times 16$. Each color (r, g, b) corresponds to the bin indexed by $(\lfloor \frac{r}{16} \rfloor, \lfloor \frac{g}{16} \rfloor, \lfloor \frac{b}{16} \rfloor)$ in the table (where $\lfloor x \rfloor$ is the floor of x). The color histogram of an image initially contains at each bin the fraction of pixels in that image whose colors correspond to that bin. Then it gets normalized, so that its length (as a vector) is equal to 1.

  The correlation $C(A, B)$ between two normalized histograms A and B is defined as $C(A, B) = \sum_{i=0}^{15} \sum_{j=0}^{15} \sum_{k=0}^{15} (A_{i,j,k} B_{i,j,k})$, where $A_{i,j,k}$ and

$B_{i,j,k}$ are respectively the bins in A and B indexed by (i, j, k).

We create a graphics color histogram H_g by picking hundreds or thousands of graphics, taking the average of their color histograms and normalizing it. We similarly create a photographs color histogram H_p using a large set of photographs.

Suppose that an image I has a color histogram H_i. Let $a = C(H_i, H_g)$ and $b = C(H_i, H_p)$. The score of the image in the color histogram metric is defined as $s = \frac{b}{a+b}$. Clearly, as $C(H_i, H_p)$ increases, s goes up, and as $C(H_i, H_g)$ increases, s goes down. Therefore, we expect photographs to score higher in this metric.

- The farthest neighbor histogram metric. The farthest neighbor histogram of an image is a one-dimensional histogram with 766 bins (as many as the possible transition values for a pixel, as defined in the farthest neighbor metric). The i-th bin (starting with 0) contains the fraction of pixels with transition value equal to i. We create average histograms F_g and F_p for graphics and photographs respectively, in the same way as in the color histogram metric. We define the correlation $D(A, B)$ between histograms A and B as $D(A, B) = \sum_{i=0}^{765} A_i B_i$, where A_i and B_i are respectively the i-th bins of A and B.

  Let F_i be the farthest neighbor histogram of the image, $a = D(F_i, F_g)$ and $b = D(F_i, F_p)$. Then, the score s of the image in this metric is defined as $s = \frac{b}{a+b}$. As in the color histogram metric, we expect photographs to score higher than graphics.

- The dimension ratio metric: Let w be the width of the image in pixels, h be the height, m be the greatest of w and h and l be the smallest of w and h. The score of an image is $\frac{m}{l}$. Graphics very often score above 2, whereas photographs rarely do so.

- The smallest dimension metric: The score of an image is the length of its smallest dimension in pixels. It is much more common for graphics to score below 30 in this metric than it is for photographs.

Metric	T	E_g	E_p	E
Color histogram	0.46	11.9	9.4	10.6
Farthest neighbor histogram	0.35	12.5	9.0	10.7
Farthest neighbor version 2 (264)	0.17	13.0	16.6	13.7
Prevalent color	0.26	13.8	13.9	13.9
Farthest neighbor version 1 (1)	0.16	14.9	15.2	15.1
Saturation (63)	0.67	32.0	6.7	19.3
Number of colors	200	13.0	34.6	23.8
Smallest dimension	72	33.4	14.8	24.1
Dimension ratio	1.63	47.1	12.1	30.0

Table 1: Individual metrics

Table 1 gives some indicative results for each metric. The training and the testing set we used to obtain these results consisted each of about 600 graphics and 600 photographs. The two sets were disjoint. The columns of the table have the following meanings.

- Metric is the name of the metric. If the metric uses a parameter, we give that in parentheses.
- T is the threshold we used. If more graphics than photographs score below T in the training set, images from the testing set that score below T are classified as graphics, and the rest as photographs. The reverse happens if more photographs than graphics score below T in the training set.
- E_g is the error rate for graphics in the testing set (percentage of graphics classified as photographs).
- E_p is the error rate for photographs in the testing set.
- E is the error rate overall (average of E_g and E_p). The threshold T was picked in each case so that it would minimize the error rate in the training set.

6 Combining the Metric Scores

The individual metric scores that we get are not definitive. To make the final decision, we need a decision making module that will make the final classification based on those scores. We currently use multiple decision trees for that task. Our decision tree design is based on Yali Amit's work with decision trees [1], with minor modifications, in order to adjust it to our domain.

6.1 Classification with Multiple Decision Trees

Each decision tree is a binary tree. Each non-leaf node n has a test field, which contains a metric M_n, a parameter P_n to be used with M_n (if applicable), and a threshold T_n. Each leaf node contains a real number, between 0 and 1, which is a probability estimate that the image is a photograph. To classify an image using a tree, we perform the following recursive procedure:

1. If the root r is not a leaf node, let S_r be the score of the image under the metric M_r and parameter P_r. If $S_r < T_r$, we classify the image with the subtree headed by the left child of the root. Otherwise, we use the subtree headed by the right child of the root.

2. If the root is a leaf node, we return as result the number that is stored in that node.

To classify an image using a set of trees, we find the mean A of the results that we get from all trees in the set. If A is less than a given threshold K, the image is considered a graphic, and otherwise it is considered a photograph.

6.2 Constructing the Decision Trees

To construct a decision tree, we have to specify a training set of images S, and a set of tests D. A test is either a metric or a metric together with a parameter, for those metrics that require us to specify a parameter. Images in S have been hand-classified as photographs or graphics. The following is a recursive description of how a decision tree gets constructed.

We start at the root. If the images in S are all photographs or all graphics, we stop. Otherwise, we pick the optimal test for the root, with respect to our training set. We use the same criteria as [1] to determine what the optimal test in a given node is. [4] explains the intuition behind the notion of "information gain" that we and [1] use to evaluate the informational value of a given test at a given node. If the information gain from all tests is zero, we stop. Otherwise, we recursively construct the left and right subtree under the root. For the left subtree we use as training set all images in S whose score under metric m and parameter p is less than t. We use the rest of the images as a training set for the right subtree.

6.3 Preparation of training and testing sets

The Web is a vast source of training data. The crawler we use for WebSeer can currently locate and download about 1 million images a day, together with the HTML pages that refer to them. We can hand-classify images as photographs or graphics at a rate of 2,500 images an hour. It only takes a couple of days to classify tens of thousands of images for our training and testing sets.

Web images appear in the GIF and JPEG format. We get much better results by using different decision trees to classify images in each format. Images in the two formats have important differences, that make them score differently in our metrics. For example, JPEG images have thousands of colors regardless of whether they are photographs or graphics, because of the way JPEG compression works. So, we maintain different training and testing sets for the two formats.

To create the decision trees, we used as a training set 1025 GIF graphics, 362 GIF photographs, 270 JPEG graphics and 643 JPEG photographs. To construct the average color histograms and the average farthest neighbor histograms we used about as many images, which were not included in the training sets for the decision trees. Now that we have tens of thousands of hand-classified images at our disposal, we plan to create new trees, with much larger training sets, to test how the accuracy of the system relates to the amount of training data.

After we create tens of different trees, we manually put them together into several sets, which we test in order to pick the set among them that gives the highest accuracy rate. We are looking into ways to automate that procedure, by specifying some heuristics to prune the space of all possible combinations of trees, and make sure that a reasonably good set is chosen.

6.4 Reasons for using multiple decision trees

For every image, we get thousands of scores by using different tests (combinations of metrics and parameters). Decision trees can use the tests that yield the most information, and ignore other tests that are highly correlated to the ones already used. In addition, decision trees allow us to examine them and understand exactly what image features they use, and why they fail when they fail. This is an advantage over neural networks, where it is much harder to examine the state.

Multiple decision trees offer several advantages over single decision trees:

- We have so many possible tests that, given the size of our training sets, we cannot use all the information we get in a single decision tree.

- We can add additional metrics without having to increase the size of the training set, or alter the training and classification algorithms.

- Multiple decision trees offer increased accuracy over single decision trees, even if all trees are built based on the same metrics (as long as the metrics are used in different order in the different trees, and with different parameters). Single trees are less accurate in borderline cases than groups of trees, where misclassifications in individual trees are cancelled out by correct decisions in other trees.

7 Results

As we mentioned earlier, after we get the average of the results of all decision trees for an image, we compare it with a threshold K and consider the image to be a graphic if the average is less than K. The choice of K affects directly the accuracy of the system for images of each type. As we increase K, we get a higher error rate for graphics and a lower error rate for photographs. The error rate for images of a given type (photographs or graphics) is defined as the percentage of images of that type that the system classifies incorrectly.

K	GIF graphics	GIF photographs
0.37	8.2	3.5
0.38	6.0	3.9
0.40	5.0	6.9
0.42	4.2	7.4
0.44	3.6	12.1
0.50	2.4	17.8

Table 2. Error rates for GIF images.

K	JPEG graphics	JPEG photographs
0.40	20.0	3.4
0.44	16.4	4.4
0.47	11.8	6.4
0.50	9.3	8.7
0.55	6.1	15.3
0.59	5.0	17.6

Table 3. Error rates for JPEG images.

We tested the system on random images we downloaded from the Web and classified by hand. The test images consisted of 7245 GIF graphics, 454 GIF photographs, 2638 JPEG graphics and 1279 JPEG photographs. None of those images was used in constructing the average color and farthest neighbor histograms, in constructing the trees, or in experimenting with sets of trees to decide which set we should use.

Tables 2 and 3 gives error rates, as percent values for different choices of K, for the two formats.

In WebSeer, we set K to 0.5 both for GIF and JPEG images. GIF graphics are by far the most common image type on the Web. They occur about 15-20 times as often as GIF photographs, and about 6 times as often as JPEG images of any kind. Our choice of K allows GIF graphics to be classified correctly at a rate of 97.6%. If we had allowed a 5% error rate for GIF graphics, about 40% of all images classified as photographs by our system would be GIF graphics.

We measured the rate at which the photo detector can classify images. The measurements were made on an UltraSPARC-1, running at 167MHz, using the same images that were used for the results in tables 2 and 3. The images were read from disk. The speed of the system was 2.6 images per second.

8 Current and Future Work

Current work focuses on two areas: improving the accuracy of the system, and extending it to mixed images.

8.1 Decision making

As mentioned in previous sections, we now have at our disposal tens of thousands of hand-classified images, which we can use for training and testing. We plan to use those images to construct new decision trees, so that we can check whether larger training sets can improve the accuracy of the system.

A fact that our current decision-making module ignores is that, for some metrics, certain ranges of scores indicate with very high probability that an image is a graphic. For instance, in the color histogram metric, 310 out of 618 GIF graphics score below 0.2, and only 1 in 454 photographs does so. Currently, images that score below 0.2 in that metric get classified correctly by the trees that use that metric, but may get classified incorrectly by the rest of the trees, and by the system as a whole. It may be advantageous to force the system to classify all such images as graphics.

8.2 Image Context Metrics

Image context is the information that we have about an image that does not come from the image content. Images in multi-media documents have a rich context around them, which can be useful for our system.

The context of an image occuring on the Web is the HTML page in which the image is embedded. The content of that page gives us several clues about the type of the image. If an HTML page has a link to an image but does not actually display it, that image is usually a photograph. Images with the USEMAP and ISMAP attributes are usually graphics. The URL of an image can give statistically useful information; words like “logo”, “banner”, “bar”, appear much more frequently in URLs of graphics. Images that are grouped together in a page are usually of the same type. Finally, the text around an image can be useful, even if we just scan it for the appearance of specific words, like “photograph”.

Preliminary results show that we can use such information to increase the accuracy of our photo detector. We plan to implement new metrics, which will rely on the image context. We can use such metrics together with the already existing ones in decision trees.

8.3 Mixed Images

Mixed images are images that contain a photograph, but are not pure photographs. A mixed image can be a collage of photographs, a photograph with text on top, a photograph with a computer-generated frame or background around it, or any other combination of photographic and computer-generated material.

A medium term goal is to extend the system to perform some segmentation, so that it can identify photographic parts in mixed images. It seems that segmenting an image into photographs and computer-generated parts can be easier than the generic image segmentation problem. For at least some cases, like photographs with one-color backgrounds, it is pretty easy to segment out the computer-generated part, and gibve the rest to the system to classify.

9 Conclusions

Our photo detector is an efficient and highly accurate system, that can be used to classify images as photographs or graphics based on the image content. It is currently being used in WebSeer, an image search engine for the Web, to help index millions of images in ways meaningful to people who search for images on the Web. Its design is based on some simple statistical observations about the image content of graphics and photographs. Its implementation makes heavy use of the vast amount of training data that is available on the Web. The availability of training data allows us to use many statistical observations in lieu of a more definitive model of the differences between photographs and graphics. Initial results suggest that using the image context as well as the image content would further improve the accuracy of the system.

We believe that similar approaches, that rely on statistical observations, combine image content with image context, and make use of the availability of huge amounts of training data from the Web, can be useful in extracting additional information from an image. Examples of additional image types that we hope to detect using such methods are maps, charts, cartoons, and astronomical pictures. Detecting such types would be very useful in indexing images in extensive collections of multimedia documents, like the Web, in a meaningful way.

References

[1] Y. Amit and D. Geman (1987). “Randomized Inquiries About Shape: an Application to Handwritten Digit Recognition”

[2] John R. Smith and Shih-Fu Chang (1996). “Searching for Images and Videos on the World Wide Web”. Technical Report # 459-96-25. Center for Telecommunications Research, Columbia University.

[3] Neil C. Rowe and Brian Frew (1997). “Automatic Caption Localization for Photographs on World Wide Web Pages”. Department of Computer Science, Naval Postgraduate School

[4] Tom Mitchell (1997). “Machine Learning”. McGraw Hill