

Honours Year Project Report

Synthetic Image Categorization

By

Wang Fei

Department of Computer Science

School of Computing

National University of Singapore

2005/2006

Honours Year Project Report

Synthetic Image Categorization

By

Wang Fei

Department of Computer Science

School of Computing

National University of Singapore

2005/2006

Project No: H079200

Advisor: Dr. Min-Yen Kan

Deliverables:

Report: 1 volume

Program: 1 CD

Abstract

We introduce NPIC, an image classification system that focuses on synthetic (e.g., non-photographic) images. We use class-specific keywords in an image search engine to create a noisily labeled training corpus of images for each class. NPIC then extracts both content-based image retrieval (CBIR) features and metadata-based textual features for each image for machine learning. We evaluate this approach on three different granularities: 1) natural vs. synthetic; 2) map vs. figure vs. icon vs. cartoon vs. artwork; 3) and further subclasses of the map and figure classes. The NPIC framework achieves solid performance (99%, 97% and 85% in cross validation, respectively). We find that visual features provide a significant boost in performance, and that textual and visual features vary in usefulness at the different levels of granularities of classification.

Subject descriptors:

- I.2.6 Learning
- I.4.10 Image Representation
- I.5.2 Design Methodology

Keywords:

Image categorization, textual features, visual features, feature extraction, machine learning

Implementation software and hardware:

Perl, Java, MS-Windows, LINUX, BoosTexter, Intel OpenCV, ImageMagick, Perl XPM library

Acknowledgement

I would like to thank Dr. Min-Yen Kan, for his generous listening, patient guidance, and invaluable advice and help in pointing out the various tools I could utilize in this project.

Table of Contents

Title	i
Abstract	ii
Acknowledgement	iii
1. Introduction	1
2. Background	3
2.1 Related Works	3
2.2 Our previous work — NPIC system developed during UROP	4
2.3 Difficulties involved in image classification	5
3. System definition	8
3.1 An ontology of images	8
3.2 Level 1 classification — natural vs. synthetic images	10
3.3 Level 2 classification	11
3.4 Level 3 classification — maps	14
3.5 Level 3 classification — figures	15
3.6 NPIC at a glance	17
3.7 An overview of features used in NPIC	18
3.8 Hypothesis on the correctness of class labels	22
4. Methodology and system implementation	23
4.1 Automatic corpus collection using image search	24
4.2 Feature extraction	25
4.2.1 Extraction of textual features using “identify”	25
4.2.2 Extraction of color features using xpm module	26
4.2.3 Acquisition of other visual features using Intel OpenCV library	27
4.3 Machine learning and cross validation	29
5. Evaluation	
5.1 Reliability of Google image retrieval	30
5.2 Testing on Google and Wikipedia datasets	31
5.3 Feature performance	33
5.4 Scale testing on various corpus sizes	34
5.5 An error case	35
6. Conclusion and future works	37
Reference	38
Image References	40

1. Introduction

Images created entirely by digital means are growing in importance. Such synthetic images are an important means for recording and presenting visual information. The accurate classification of these images — such as icons, maps, figures and charts — is becoming increasingly important. With the advent of the web, images are being used not just to communicate content but also for decoration, formatting and alignment. An image classification system can improve image search and retrieval engines and can act an input filter for downstream web processing as well as image understanding systems.

We introduce NPIC, an image classification system that is specifically tuned for synthetic images. The implemented system uses semi-supervised machine learning to create a classifier. It does this by first using class-specific keywords to build a corpus of associated images via an image search engine. Textual features are extracted from the filename, comments and URLs of the images and content-based image retrieval (CBIR) features are also extracted. These features are strung together as a single feature vector and fed to a machine learner to learn a model. The resulting system is able to enhance the performance of text-only based image search, as the addition of visual features allows some spurious image matches to be correctly rejected.

A classifier needs ground truth labels to classify against. Existing image classification taxonomies are a good starting point. However, our dataset comes from the web, and in our opinion, there does not exist a suitable taxonomy of all content images that are found on the web. After sampling synthetic images culled from the web, we decided to create our own hierarchy for the classification of web images, loosely based on portions of the Getty Art and Architecture Thesaurus (Getty AAT).

NPIC achieves very good classification accuracy on all three granularities that we have trained the system on using a machine learner. A key point in the analysis of our study shows that although textual features are an immense help to synthetic image classification, their efficacy can be eclipsed by CBIR features at finer granularities.

After reviewing the related works on image classification in Section 2, we will describe our system, including the design for the image hierarchy, a detailed discussion of both the textual and visual features utilized in NPIC, and a hypothesis we make on the training dataset. We will continue in Section 4 by discussing our methodology: how we construct our training data set using the commodity image search engine, Google Image Search; how features are extracted from the images in the corpus; and, at last, how machine learning and cross validation are performed on the corpus. Finally, in Section 5, we will describe our experimental results.

2. Background

2.1 Related works

Image classification is a relatively young field of research, with many published systems being created after the year 2000. As of today, although many image categorization systems have been created, most classify against a very general classification scheme. A representative example is Lienhart and Hartman (2002), which implemented and evaluated a system that performs a two-stage classification of images: first, distinguishing photo-like images from non-photographic ones, followed by a second round in which actual photos are separated from artificial, photo-like images, and non-photographic images are differentiated into presentation slides, scientific posters and comics. The WebSeer system (Swain, Frankel and Athitsos, 1997) investigates how to classify images into three categories: photographs, portraits and computer-generated drawings. Both schemes are neither exclusive nor exhaustive; many images fall into multiple categories or none of the above. Work in image classification has also focused on specific synthetic image classes. Huang, Tan and Loew (2003) and Carberry, Elzer, Green, McCoy and Chester (2004) deal only with chart images. These works aim to classify and then extract the data and semantic meaning of several types of charts: such as bar, pie and line charts. Similar to our work, Hu and Bagga (2003) classifies web images found in news sites by their functionality: including classes for story images, advertisements, server host images, icons and logos.

Textual features. Quite a bit of research focuses on the textual features related to an image. Swain et al (1997) and Hu et al (2003) performed classification based on textual features such as the filename, alternate text, hyperlink and text surrounding the image. Both papers deal only with web-accessible images, so hyperlinks are always available to be used. Attempts have also been made to detect and recognize text embedded in images. Cao and Tan (2001) and Zhong, Karu and Jain (1995) use spatial variance and color segmentation techniques to separate text segments from graphics on an image. Optical character recognition (OCR) or similar techniques can often extract the text from regions of the image. Using this technique, Zhou and Lopresti (1997) detects text on images by examining connected components that satisfy certain criteria. Thus it is probably not surprising that Munson and

Tsymbalenko (2001) argues that textual features of images are far more useful in determining which images to return for a search query.

This, however, will not work in cases where an image to be classified does not come from the web. Reliance on textual features might degrade the system performance when an image is not identifiable by these features, yet is easily associated with a category by the image's visual features.

Visual features. Most systems use simple visual features such as the most prevalent color, width-to-height ratio, and image file type, among others. Using additional features from the image itself is the focus of Content Based Image Retrieval (CBIR). CBIR systems have progressively advanced, but practically all systems share a body of features based on the image's color histogram, texture, edge shape, and regions. From these low-level features, higher-level features that may have semantic meanings can be identified and built. For single images, region segmentation (Carson, Belongie, Greenspan and Malik, 2002 and Ma and Manjunath, 1997) or block segmentation (Smith and Chang, 1994) is usually done followed by spatial layout based matching of regions or statistical feature extraction (Wang, Chan and Wiederhold, 1999). Feature analysis of the same color, texture and line features can then be assessed for individual regions and matched.

While CBIR has undoubtedly improved much over recent years and has shown its mettle in retrieving similar images to a target, it remains a technology that seems to be omitted from standard image search. This is largely due to the fact that searchers would rather type in a textual description as a starting point. Automatic, content-based blind feedback on the top few ranked images also does not seem to work as performing text-based search followed by CBIR would be computationally expensive.

2.2 Our previous work -- NPIC system developed during UROP¹

In the paper we published in UROP 2004 on a similar topic (Wang and Kan, 2004), we discussed our previous work on non-photographic image categorization (and

¹ University Research Opportunity Program.

hence the name *NPIC*). The features we used in the UROP system were mainly text-based and very few visual features were then employed. In addition, the NPIC system developed in UROP only classifies a non-photographic image, i.e. a synthetic image, into five classes, namely *maps*, *figures*, *cartoons*, *artwork* and *icons*.

For the NPIC we developed for HYP¹, however, we proposed a much more sophisticated three-level classification scheme that handles any image in general, instead of being only able to classify the synthetic ones, as in UROP system. Besides, we acquired a larger image corpus to work with; size of training dataset is important to giving better predictions using machine learning. Tremendous effort has also been put in searching for more discriminative visual features. In this report, we will present these visual features that, to our knowledge, have never been used in past works; and we will also show how they help boost up the efficacy of classification in NPIC.

2.3 Difficulties involved in image classification

As listed in Section 2.1, there have been numerous attempts to develop an automatic image classification system using machine learning based on different sets of textual and visual features. However, very few achieved very high accuracy. Image classification is a difficult task due to a number of reasons discussed below.

First, for a complex image classification scheme, even if it is one that is both exhaustive and exclusive, two independent human annotators might not always assign the same class label to a given image, especially when it possesses characteristics shared by two or more classes. For example, **Figure 1** shows a screen capture in which a face photo appears. Some may consider this image as a synthetic image since it contains artificial graphics, i.e. the Operating System desktop on the background of the image. However, others may well argue that this is a photograph due to the presence of the face photo.

¹ Honours Year Project.



Fig. 1. An image that shows a person's face with a desktop background.

Second, extraction of advanced visual features from images cannot be done easily with an acceptable degree of accuracy. Many of the previously developed systems used OCR techniques to help perform classification by acquiring the semantic meaning of the embedded text. This, due to various limitations like font and text alignment, does not always give satisfactory results. Besides, acquisition of high-level visual features is also very costly. Systems like Google Image Search cannot afford the high computation complexity involved in the extraction of these features.

Third, different methods and features are good for different classes. For example, a Windows icon image is easily distinguished by its file extension *.ico* but a shape detector that recognizes circles and rectangles is more useful in telling a bar chart from a pie chart. Google Image Search, as discussed above, does not use image visual content to match queries. Instead, it relies solely on textual features such as image filename and URLs. Therefore, a search using the keyword “bar chart” returns two images that are not bar charts but contain the query keyword in their filenames, as shown in circles in **Figure 2**.

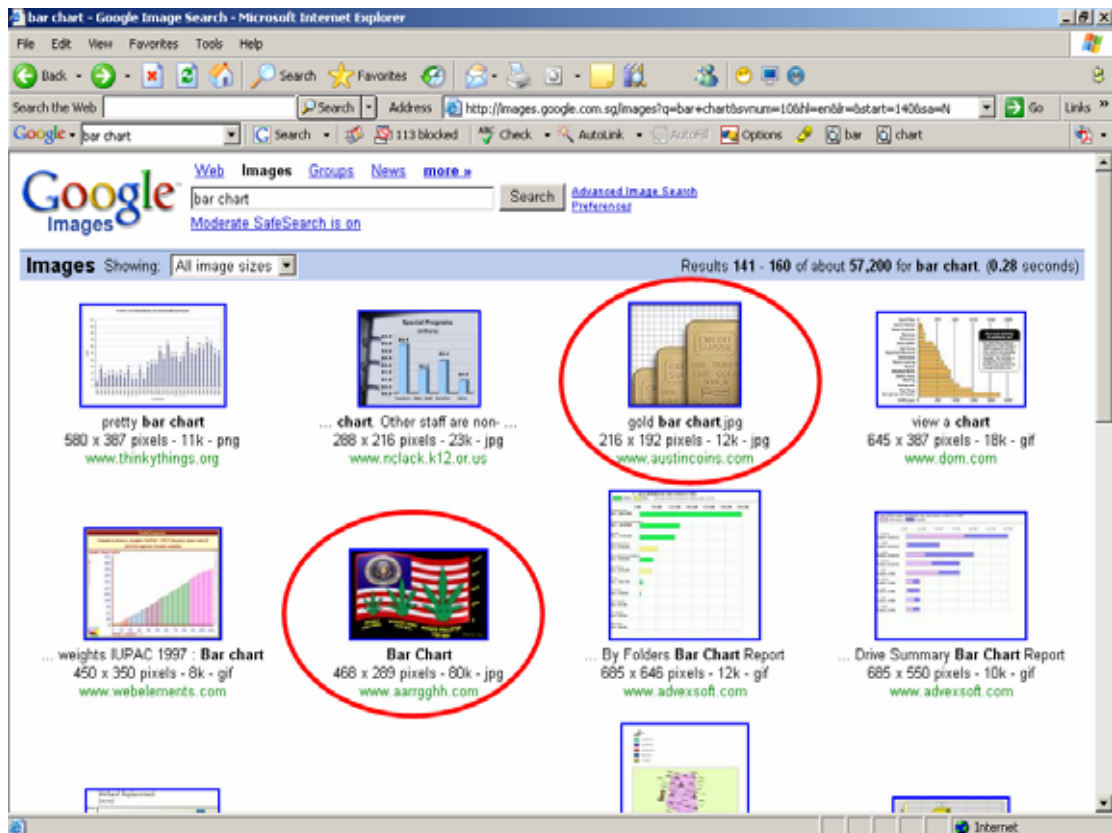


Fig. 2. Screen capture of Google Image Search query result on “*bar chart*”. Two images in red circles are incorrectly returned in the results page as they are not bar charts.

3. System definition

3.1 An ontology of images

NPIC system presents a three-level image classification shown in **Figure 3** below.

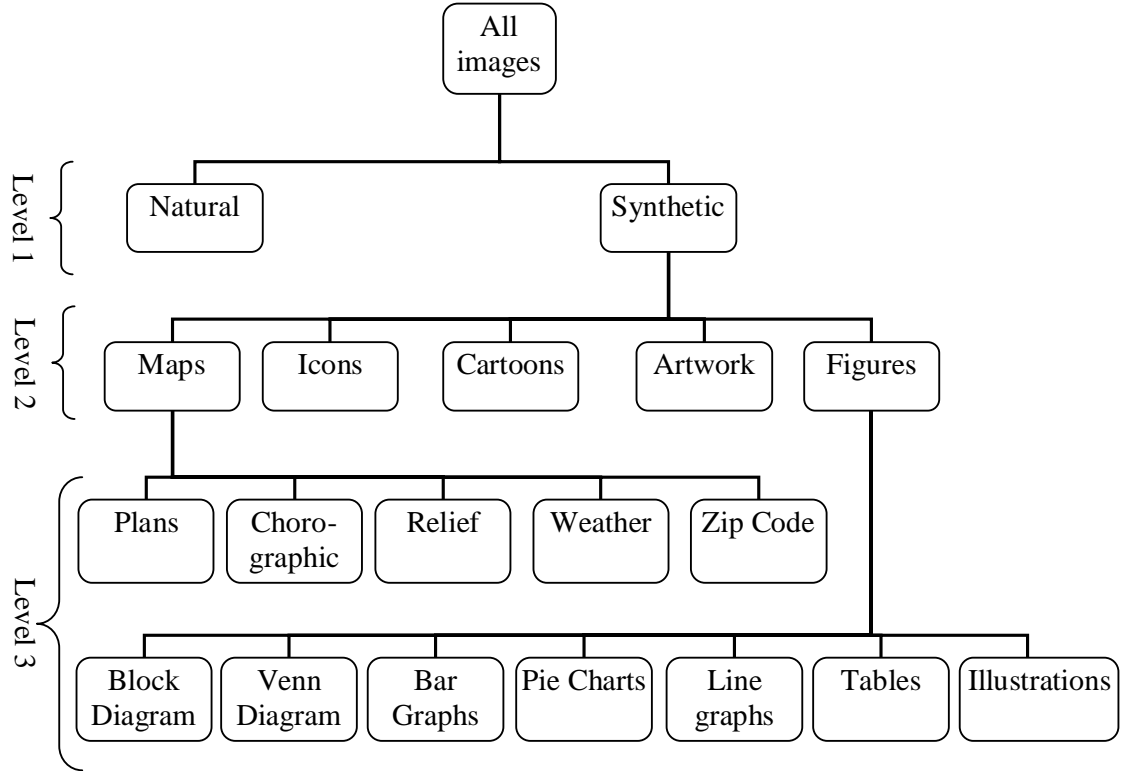


Fig. 3. NPIC's classification hierarchy

Our previous UROP system dealt with only synthetic images, i.e. Level 2 classification, so as a starting point, we will first discuss the taxonomy of synthetic images. To our knowledge, few classifications of synthetic images exist. In our exploration of related research, only Lienhart et al (2002)'s work has addressed synthetic images specifically. In their work, synthetic images found on the web are classified into four distinct categories of *photo-like images*, *presentation slides*, *scientific posters* and *comics*. Another possible way to classify synthetic images is to make use of the Getty AAT (Getty Institute, 2006). The Getty AAT is often used by museums and libraries to catalog their visual materials. It employs a faceted classification for objects, materials, activities, styles and periods (among others) and consists of over 133,000 generic terms.

A successful classification scheme must ensure that it classifies most items and that items clearly belong to distinct classes. For us, a successful classification needs to be simple enough such that an ordinary layman can understand and employ the classification scheme without needing specialist knowledge. Given these criteria, we feel neither Lienhart and Hartmann's classification (covers only certain types of web images) nor the Getty AAT scheme (too complex) is suitable.

Instead, our classification is based on what types of synthetic images a user encounters during her daily computing tasks. Our classification has five broad categories: *maps*, *figures*, *icons*, *cartoons* and *artwork*. We include *icons* not only because many images found on a computer are icons that are associated with programs or data files, but also due to the fact that small pictures such as navigation button icons, logos and symbols exist in web pages in vast quantities. *Artwork*, on the other hand, includes work drawings and pictures representing aesthetic images. *Figures* include all types of abstract data representations. In our empirical analysis, this classification covers a large portion of important functional image types that users encounter.

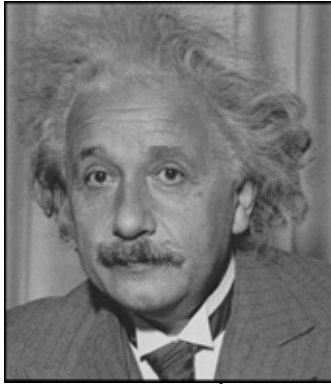
As most images do not come labeled as synthetic or natural, we must include and implement a super-ordinate classifier to distinguish between *natural* (photographs of real-life scenes and objects) and *synthetic* (digitally-made, computer-generated) images for NPIC to be useful. Also, the two classes of *maps* and *figures* can be refined as they are quite general. We use the Getty AAT to refine these two image types. The Getty AAT has classifications for maps based on its form, function, production method, or subject. Based on our analysis, we conflated these schemes to produce a single sub-ordinate classification of five categories: *plans*, *chorographic maps* (i.e., maps of large regions), *relief maps*, *weather maps* and *zip code maps*. Following the same editorial selection of the relevant Getty AAT categories, we construct a categorization of *figures* into seven categories: *block diagrams*, *venn diagrams*, *bar graphs*, *pie charts*, *line graphs*, *tables*, and *illustrations*.

We would like to emphasize that the hierarchy developed here constitutes a working attempt to compile a useable and useful classification for typical end users, and should

not be construed as a formal model for image classification. Other image classes or alternate organizations can be also considered; such alternative classification schemes would work equally well in the NPIC framework.

In the following few sections, we will give a detailed description of images in each class, together with some examples, aiming to present a clear picture of how the various different characteristics help distinguish classes from one another. Possible discriminative features are hypothesized for each class of images.

3.2 Level 1 classification — *natural* vs. *synthetic* images

Category	Description	Example images
Natural	Natural images refer to photographs of persons, objects or scenes, normally recorded by a camera. They normally contain many colors and irregularly shaped objects. It is very unlikely to find any considerably large region with consistent color.	 Image 1. ¹

¹ Image sources are listed at the end of the report, under the section *End Notes*.

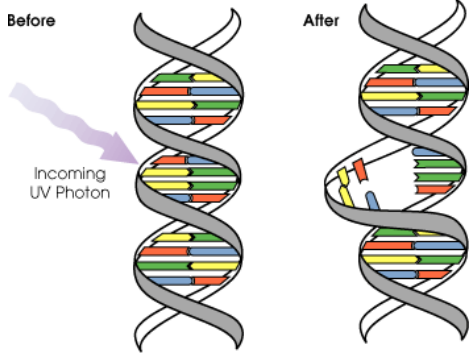

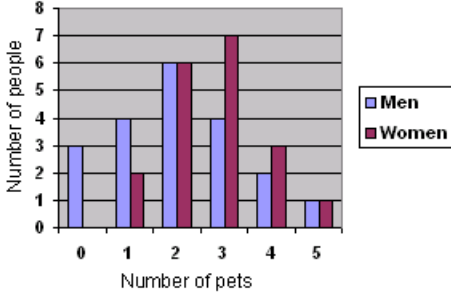
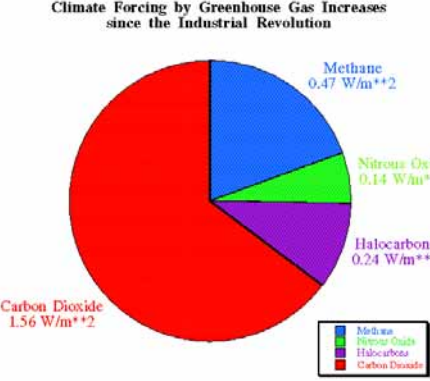
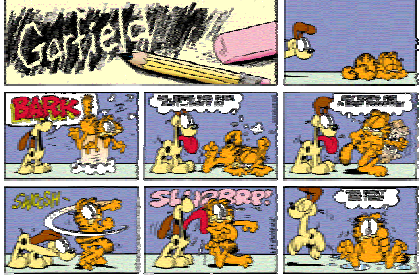
Synthetic	<p>Synthetic images are computer-created images that are artificially made but not acquired with a camera. This class includes scanned synthetic images as well. Most synthetic images have single-colored background. Embedded text blocks are sometimes associated with this class of images. Regular shapes commonly found on synthetic images include circles, sectors, rectangles and squares, among others.</p>	 <p>Image 2.</p>
------------------	---	--

Fig. 4. Level 1 classification.

3.3 Level 2 classification — *maps, figures, artwork, cartoons* vs. *icons*

Category	Description	Example images
Maps	<p>Maps are photogrammetric representations, usually on a plane surface, of a region or an area on Earth, including physical features and/or political boundaries, where each point corresponds to a geographical or celestial position according to a definite scale or projection. They typically contain a large number of text regions and single-colored blocks, and usually have larger dimensions and higher resolutions. Maps are also characterized by intersecting curve lines, normally depicting land features such as roads, rivers, railways or territory boundaries. Some examples are street maps and floor plans.</p>	 <p>Image 3.</p>
	<p>The class “figures” includes non-artistic graphic designs, that are used to explain rather than to represent, created by image creation software such as Microsoft Excel®.</p>	

<p>Figures</p>	<p>They are usually used for presentation, illustration and demonstration purposes. Charts, scientific posters, graphs, presentation slides are some examples of figures.</p> <p>Figures are hypothesized to possess the following characteristics:</p> <ul style="list-style-type: none"> • Text blocks are often found in diagrams. • Many figures, like presentation slides, have headings in the top. • Straight lines and arcs are common. • They usually have fewer colors than image of other categories. • Some figures contain simple shapes like circles, sectors and rectangles. 	 <p>Image 4.</p>  <p>Image 5.</p>
<p>Cartoons</p>	<p>Cartoon images are comic strips that are usually taken from the web or scanned from comic books. Multiple cartoon images are often found to exist on the same picture, i.e. the entire picture is divided into rectangular blocks by vertical and horizontal bars, showing a cartoon image in each rectangle: the “Garfield” cartoon picture displayed on the right, for instance. Cartoons are sometimes seen in the form of grayscale images since many of them are acquired from “black-and-white” comic book scans. Those in color contain many large single-colored regions as cartoons are simple color drawings that do not have many inter-crossing lines like for the case of maps.</p>	 <p>Image 6.</p>
	<p>Artwork, as opposed to figures, refers to artistic drawings and computer graphics like a screen</p>	







<p>Artwork</p>	<p>capture of the Counter Strike game, shown on the right. Generally they tend to contain a larger number of colors and fewer consistent-colored blocks than the other image classes. Color gradients are commonly seen on artwork images too.</p>	 <p>Image 7.</p>  <p>Image 8.</p>
<p>Icons</p>	<p>Icons are defined as small synthetic images (fairly often square-shaped) that have special uses. Examples include operating system icons that are associated with the file system and programs, website navigation buttons and website logos. They are very distinct from other classes of images because most of them have small dimensions and file sizes as a result of low image quality and resolution. Another key discriminative feature of icons is that many of them have the file extension “.ico”, which makes them easily identifiable. Therefore, we expect high classification accuracy for icon images.</p>	 <p>Image 9.</p>

Fig. 5. Level 2 classification.

3.4 Level 3 classification — maps

Category	Description	Example images
Plans	Plans are maps depicting a relatively small district or region, such as a building, i.e. a floor plan, drawn on a large scale and with considerable details. Plan maps normally contain fewer colors but more right-angled corners and regularly shaped objects, such as rectangles.	 <p>Image 10.</p>
Choro-graphic maps	Chorographic maps represent relatively large regions, such as cities, countries or continents, on a small scale. Chorographic maps may include the natural configuration and features of a region, in addition to political boundaries and major towns. Usually this type of maps does not contain many colors or regular shaped objects. Besides, straight lines are hardly found but curve lines are very common.	 <p>Image 11.</p>
Relief maps	Relief maps show land or sea bottom relief in terms of height above or below a datum by any method, such as contours, hachures, shading, or tinting. A relief map is often found containing similar colors, but of different brightness forming color gradients.	 <p>Image 12.</p>

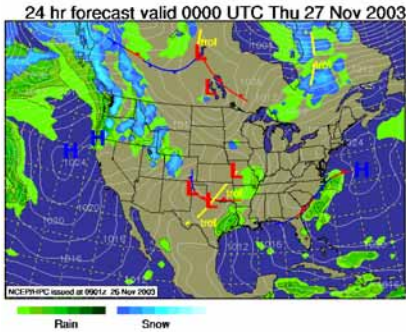

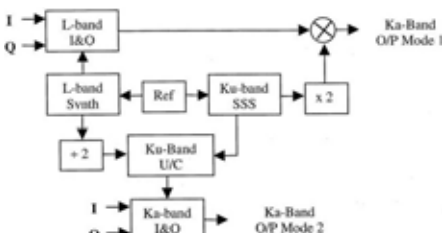
Weather maps	Weather maps, or climate maps, depict various weather elements of a region such as temperature, humidity, wind, cloud cover, and air pressure, by using different colors, different color gradients, and occasionally isobaric and isothermic lines. Most weather maps do not contain many text regions.	 <p>Image 13.</p>
Zip code maps	Zip code maps, and similar classes like telephone area code maps, are easily distinguishable by many similarly sized blocks of homogeneous color, as shown in the example image.	 <p>Image 14.</p>

Fig. 6. Level 3 classification of maps.

3.5 Level 3 classification — figures

Category	Description	Example images
Block diagrams	Block diagrams use simple labeled shapes like rectangles, circles, rhombuses, and triangles, connected by straight lines to represent the relationship of parts or phases.	 <p>Image 15.</p>
Venn diagrams	Venn diagrams are pictorial representations using circles or ellipses so positioned as to represent an operation in set theory. Sometimes, an enclosing rectangle depicting the “universal set” also exists on Venn diagrams.	

		<p>Image 16.</p>
Bar graphs	<p>Bar graphs are figures that present statistical information using rectangular bars of different height plotted in a coordinate system. They are prominently characterized by a set of parallel aligned rectangles.</p>	<p>Image 17.</p>
Pie charts	<p>Pie charts represent statistical information, especially that pertaining to proportions and percentages, in the form of sectors of a circle, as shown by the example pie chart on the right. Obviously, the most distinctive feature associated with pie charts is a circle (for a 2D pie chart) and an ellipse (for a 3D pie chart).</p>	<p>Image 18.</p>
Line graphs	<p>Line graphs are graphical plots that represent relationships of variables of horizontal axis and vertical axis using lines connecting a set of data points. Sometimes the graph contains grid lines, as in the case of the example shown on the right.</p>	<p>Image 19.</p>
	<p>Tables are a condensed, orderly arrangement of data, especially</p>	

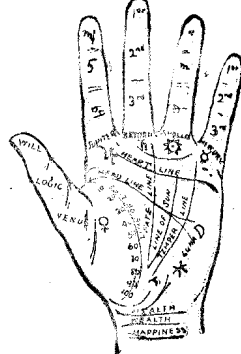
Tables	those in which the data are arranged in columns and rows. Figures of this type are normally featured by having a matrix of equally sized rectangles.	<div><p>Travel Expense Report</p><table><tr><th></th><th>Meals</th><th>Hotels</th><th>Transport</th><th>subtotals</th></tr><tr><td>San Jose</td><td></td><td></td><td></td><td></td></tr><tr><td>25-Aug-97</td><td>37.74</td><td>112.00</td><td>45.00</td><td></td></tr><tr><td>26-Aug-97</td><td>27.28</td><td>112.00</td><td>45.00</td><td></td></tr><tr><td>subtotals</td><td>65.02</td><td>224.00</td><td>90.00</td><td>379.02</td></tr><tr><td>Seattle</td><td></td><td></td><td></td><td></td></tr><tr><td>27-Aug-97</td><td>96.25</td><td>109.00</td><td>36.00</td><td></td></tr><tr><td>28-Aug-97</td><td>35.00</td><td>109.00</td><td>36.00</td><td></td></tr><tr><td>subtotals</td><td>131.25</td><td>218.00</td><td>72.00</td><td>421.25</td></tr><tr><td>Totals</td><td>196.27</td><td>442.00</td><td>162.00</td><td>800.27</td></tr></table></div> <p>Image 20.</p>		Meals	Hotels	Transport	subtotals	San Jose					25-Aug-97	37.74	112.00	45.00		26-Aug-97	27.28	112.00	45.00		subtotals	65.02	224.00	90.00	379.02	Seattle					27-Aug-97	96.25	109.00	36.00		28-Aug-97	35.00	109.00	36.00		subtotals	131.25	218.00	72.00	421.25	Totals	196.27	442.00	162.00	800.27
	Meals	Hotels	Transport	subtotals																																																
San Jose																																																				
25-Aug-97	37.74	112.00	45.00																																																	
26-Aug-97	27.28	112.00	45.00																																																	
subtotals	65.02	224.00	90.00	379.02																																																
Seattle																																																				
27-Aug-97	96.25	109.00	36.00																																																	
28-Aug-97	35.00	109.00	36.00																																																	
subtotals	131.25	218.00	72.00	421.25																																																
Totals	196.27	442.00	162.00	800.27																																																
Illustration diagrams	Illustration diagrams refer to figures that clarify or provide an example or visualization. They usually accompany a text, often found in books, published journal or web pages explaining certain concepts.	<div></div> <p>Image 21.</p>																																																		

Fig. 7. Level 3 classification of figures.

3.6 NPIC at a glance

NPIC system classifies images on three granularities using a machine learner. Basically, it works by first building up a decision tree of rules based on the training examples, and then, makes predictions on a set of testing examples. Each example is made up of a list of attribute values. Two types of attributes are used in NPIC, namely, continuous attributes and textual attributes. For instance, image's height is a continuous attribute whose value might have some correlation with the image category. The machine learner will first find a threshold, and then construct a rule, say, "images with height less than 100 pixels are likely to be icons". Textual attributes contain words that may be found to correlate with image classes. The machine learner, for instance, may find images with file extension "*ico*" to highly correlate with the *icon* class.

3.7 An overview of features used in NPIC

A collection of textual and visual features are extracted from images and used by the machine learner for classification. **Table 1** below gives a detailed description of main features used in NPIC, together with references to past published papers that used a particular feature.

Feature	Description	References
Textual Features		
Filename	Image filename without extension.	Swain et al (1997),
File extension	Extension of the file, if any. Common image extensions are jpg, gif, png, ico, etc.	Hu et al (2003) and Munson et al (2001)
Comments	Comments in Image metadata header.	New
Image URL	URL components of the location of the image on the Web (if applicable).	Swain et al (1997),
Page URL	URL components of the enclosing page of the image.	Hu et al (2003) and Munson et al (2001)
Visual Features		
Height	Image dimensions in pixels.	Lienhart et al (2002), Swain et al (1997) and Hu et al (2003)
Width		
X resolution	Number of pixels per inch (dpi) along X and Y dimensions	New
Y resolution		
File size	Size of image in bytes.	New
C₁	Most common color.	Swain et al (1997) and Hu et al (2003)
C₁ Fraction	Fraction of pixels in the image that have color C ₁ .	
Background color	Background color of image.	New
Color depth	Number of bits used to represent one of the three components of RGB, e.g. color depth of 8 means 24 bits-per-pixel.	New
F₁	Fraction of pixels with the neighbor metric greater than zero.	Lienhart et al (2002)
F₂	Fraction of pixels with the neighbor metric greater than 3/4 of the maximum.	Lienhart et al (2002)
F₂/F₁	The ratio of F ₂ to F ₁	Lienhart et al (2002)

L1 distance	$L1 = \sum (h_i - k_i)$, where $H = \{h_i\}$ is the image histogram, and $K = \{k_i\}$ represents the average histogram in each category.	Rubner, Tomasi and Guibas (2000)
L2 distance	$L2 = \sum (h_i - k_i ^2)^{1/2}$	Rubner et al (2000)
L-∞ distance	$L-\infty = \sum (h_i - k_i ^{100})^{1/100}$, a large value of 100 is chosen to represent infinity.	Rubner et al (2000)
Jeffrey divergence distance	$L_{JD} = \sum ((h_i \log(h_i / m_i) + k_i \log(k_i / m_i)))$, where $m_i = (h_i + k_i) / 2$	Rubner et al (2000)
Chi² distance	$L_{\text{chi-square}} = \sum ((h_i - m_i)^2 / m_i)$ where $m_i = (h_i + k_i) / 2$	Rubner et al (2000)
Quadratic distance	$d_A(H, K) = \sqrt{((\mathbf{h} - \mathbf{k})^T \mathbf{A} (\mathbf{h} - \mathbf{k}))}$, where \mathbf{h} and \mathbf{k} are vectors that list every entry in \mathbf{H} and \mathbf{K} . Cross-bin information is incorporated via a similarity matrix $\mathbf{A} = [a_{ij}]$ where a_{ij} denotes similarity between bins i and j .	Rubner et al (2000)
EMD	Earth Movers Distance: $\text{EMD}(P, Q) = \frac{\sum_{i=1}^m \sum_{j=1}^n d_{ij} f_{ij}}{\sum_{i=1}^m \sum_{j=1}^n f_{ij}}$	Rubner et al (2000) and Qin, Charikar and Li (2004)
Rectangles	2 features: Number of rectangles whose sides are parallel to the image frame, fraction of entire image occupied by rectangles.	Qin et al (2004)
Circles	Number of circles with certain radius.	new
Corners	Number of corners found on the image.	new
Lines	5 features: Number of horizontal, vertical and slanted lines; average line length and average line gradient.	new

Table 1. Features in NPIC with references indicating past published works for a given feature

Feature extraction method will be discussed in greater details in Section 4. Now, let us discuss the kinds of standardization performed after feature vectors are extracted from the images.

Textual features are created by obtaining tokens from the filename, extension, and path information from the URL (when available) of the image. For this, simple tokenization is done to create a more meaningful inventory of features, e.g. `Singapore_map.jpg` → `singapore map jpg`) and thus, reducing the problems with sparse data. In cases where certain features are not available or do not exist for a particular image, a whitespace character (“ ”) and a number zero are used for textual and continuous fields respectively as a replacement. In addition, the text fields in the feature vector are tokenized and standardized to lower cases. URLs thus appear as a sequence of constituent words, with words such as “www” and “html” removed since they are too common to be useful to build a discriminative classifier.

We have chosen to use many color features as visual features since they are relatively straightforward to calculate given raster data. We follow the literature and use both the HSV and RGB color spaces for analysis. For neighbor metrics, we create features using the standard RGB and HSV color spaces, as well as reduced HS and H spaces. Color histogram features are calculated using a simplified 9-bit RGB color space.

Colors are most commonly represented in RGB color space and they are usually expressed as six-digit hexadecimal numbers. Each of the three primary colors, i.e. *red*, *green* and *blue*, is denoted by two hexadecimal digits. However, a problem arises when comparing colors in this representation. For example, we have three pixels with colors `#FFFFFF` (white), `#FFFFFFC` (grey, but very close to white) and `#000000` (black). If they are simply taken as text field entries, we are losing the information that the first pixel has a very similar color as the second than the third. This is because that these three values are mere encodings of colors, and the machine learner is only able to find out a pattern if repeated values are detected.

On the other hand, if a color is to be represented in a one-dimensional scalar, what value should be used to represent the visual differences of different colors? The scheme whereby the scalar is calculated by adding up the three RGB channel values does not prove to be effective. The two color encodings `#0000FF` and `#FF0000` have exactly the same scalar value, i.e. 255, under this scheme despite the fact that they are visually very different — the first one is blue whereas the second one is red. After

some investigation into different color encoding schemes, the HSV color space scheme seems to be a suitable choice.

This scheme describes a color using three different axes, i.e. hue, saturation and value. They are defined as follows¹:

- *Hue* refers to a particular color within the visible spectrum, as defined by its dominant wavelength. Its value ranges from 0 to 360, and two pixels with similar hue values have similar colors as perceived by human beings.
- *Saturation* refers to the intensity of a specific hue. It is based on the color's purity, i.e. a highly saturated hue has a vivid intense color, while a less saturated *hue* appears more muted and grey. With no saturation at all, the hue becomes a shade of grey. Its value ranges from 0 to 1 as defined in the system.
- The third coordinate, *value*, is the brightness of a color. For example, the yellow color is brighter than the blue color, so it has higher *value* than blue. The *value* ranges from 0 to 1 as well.

During feature extraction, some of the output obtained from *xpm* is in RGB, and a conversion to HSV is made possible via the library `Graphics::ColorObject`. As shown in **Figure 8**, in the three-dimensional color space cone, the nearer any two chosen points, the more similar their colors are. However, expressing the three coordinates, H, S and V in a single scalar is beyond the scope of this report. On the other hand, the *hue*, among the three components of HSV, gives the best measure of a color, hence, many features, like background color and the most prevalent color, are represented using *hue*.

For the rectangle, line, circle and corner detection features, we need specific settings for the spatial and scaling constraints of the detectors. Using a *laissez faire* approach, we use a wide variety of parameter settings to create different features and forward

¹ Definitions quoted from www.wordiq.com/definition/HSV_color_space.

these to the machine learner to decide which group of parameter settings should be used.

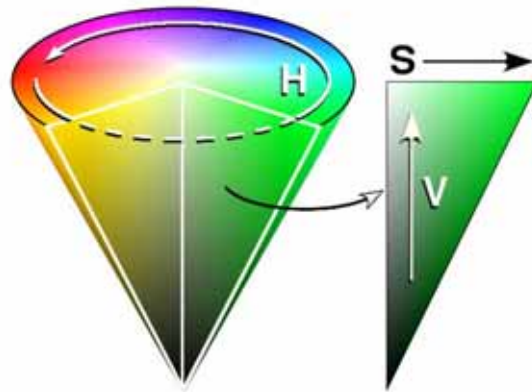


Fig. 8. HSV color space cone; image taken from www.answers.com, 2005.

3.8 Hypothesis on the correctness of class labels

Since NPIC performs image classification using machine learning, a large training dataset is required. Fortunately, despite the existence of some noisy data, Google Image Search is a good source of images. All images used in NPIC are, therefore, acquired from Google, with the exception of the icon class for which a large clean collection has been found on the web¹.

However, due to the large image corpus size, it is very expensive to manually annotate every image by giving it a true class label. Hence, we hypothesize that images retrieved in this way conform to their actual classifications. A reliability test of this hypothesis is performed in Section 5.1 where a small sample of images in each category is taken and manually annotated, and then the annotation results (correct labels) are compared with the labels given by Google. The test result agrees with the hypothesis to a high degree of accuracy.

¹ <http://www.wrclub.net/down/listdown.aspx?id=610>

4. Methodology and system implementation

Through our observations of previous classification systems, one architecture for improved image classification incorporates CBIR visual features with textual ones. This captures both the high accuracy and semantic nuances that textual features can garner, but enables classification to base purely on visual features when text either does not give any useful hint or is simply not available.

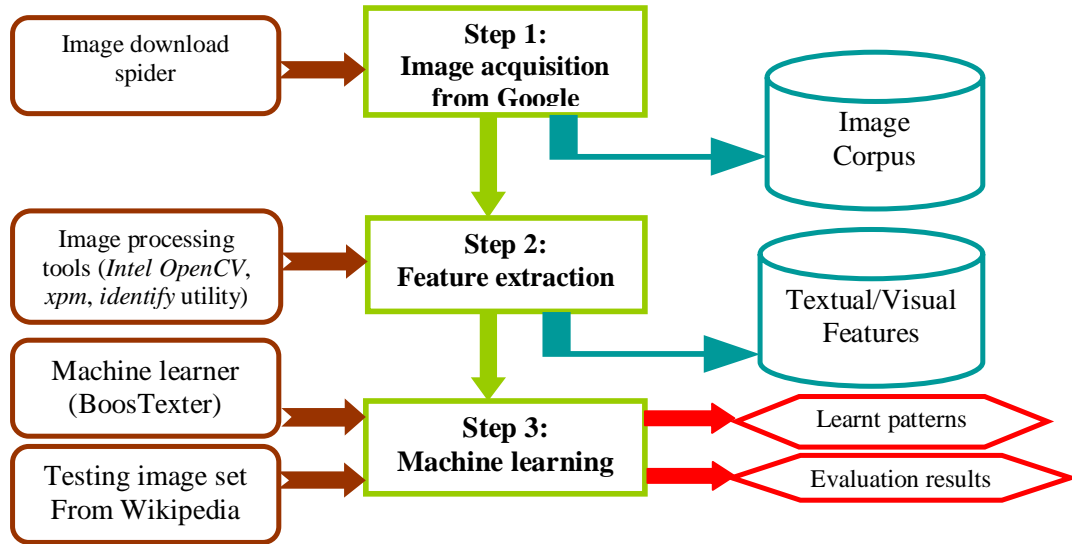


Fig. 9. System architecture

In a nutshell, NPIC performs its task in three steps. Given a taxonomy of image classification, NPIC:

- 1) Constructs a dataset of sample images for each class using Google Image Search engine.
- 2) Extracts both textual and visual features from each sample image to create feature vectors for learning.
- 3) Builds discriminative models for each set of sibling classes in the hierarchy¹ that originate from a common parent. Images can then be programmatically classified by generating their feature vector representations (step 2), followed by classification against the inferred models. The same approach is applied to all the three levels of classification, but we address synthetic images with a little bit

¹ Refers to the classification hierarchy in **Figure 3**.

more emphasis as they often carry semantic content and data that are of interest to scholars and as well as the image analysis and digital library community.

4.1 Automatic corpus collection using image search

Given this classification, NPIC needs to collect labeled image samples to extract features for supervised learning. However, as mentioned before, publicly available labeled datasets do not exist and creating one through manual effort of annotating and selecting clean images is quite costly. However, as most machine learning algorithms are robust to small amounts of noise in their training data, we can opt to create an image dataset by automatic means that may contain small amounts of mislabeled data. In this way, NPIC relies on the ratio of correctly labeled to mislabeled instances to provide sufficient evidence to discard mislabeled instances.

We do this by employing web image search engines. By searching for keywords that are indicative of the desired image category, we can form a noisy collection of images to use for training. The returned image dataset from any search is noisy, as image search engines occasionally return false matches. As long as the number of false hits is minimal, the image sets should generate useful training features for classification.

We follow this procedure to build image datasets for each of the image classes in the hierarchy. After associating each image class with a set of representative keywords (as shown in **Table 2**), we input these terms to Google’s Image Search to find matching images. We build this dataset from the bottom up, as sample images from each child class can serve as positive examples for its parent. Given a ranked list of images for a class, we programmatically extract the URLs of the images for the first n hits and then download them using *wget*. To help minimize the skew of the dataset, we extract a balanced corpus for each level (10000, 5000 and 600 images for each of the three levels, respectively), balancing the number of images extracted from each keyword. We follow this procedure for all of the categories, except for *icons*, as we have access to a clean collection of icons.

Level 1	photograph	aerial, birthday, bedroom, central library, concert, face
Level 2	artwork icons cartoons	painting, drawing, artwork <separate icon collections used> cartoon, disney, anime, garfield
Level 3	plans weather table illustration	floor, plan, fire escape weather, climate data, excel illustration, DNA molecule, engine

Table 2. Some representative keywords for classes in our image hierarchy

4.2 Feature extraction

We use standard utilities to extract both sets of features: the *identify* utility from the ImageMagick library to extract image metadata from the header; and for visual features, the *OpenCV* suite of visual detectors and the *xpm* package to examine the raster data.

4.2.1 Extraction of textual features using “identify”

ImageMagick is a robust collection of tools and libraries that can be used to manipulate images in various formats. The *identify* utility (with *-verbose* option on) describes the characteristics of images and outputs them in a simple text format. For example, running *-verbose* on the image “middle_east_95.jpg” yields the information presented in **Figure 10**. We extract the image feature values from the output and standardize them where necessary: file sizes in the form of “10kb” and “1.2mb” are converted to “10240” and “1258291” respectively; and X11 color name is converted to the standard hexadecimal notation, e.g. grey74 → #bdbdbd.

```
Image: middle_east_95.jpg
Format: JPEG (Joint Photographic Experts Group JFIF format)
Geometry: 1443x1699
Class: DirectClass
Type: true color
Depth: 8 bits-per-pixel component
Colors: 261308
Profile-iptc: 424 bytes
Resolution: 200x200 pixels/inch
Filesize: 423.3kb
Interlace: None
Background Color: grey100
Border Color: #DFDFDF
Matte Color: grey74
Dispose: Undefined
Iterations: 0
Compression: JPEG
Tainted: False
User Time: 0.180u
Elapsed Time: 0:01
```

Fig. 10. Output of *identify -verbose*

4.2.2 Extraction of color features using *xpm*¹ module

Image color metric is derived from *xpm*, a widely-available Perl library. This module allows access to individual pixels of an image in *xpm* format given a pair of *x* and *y* coordinates; the traversal is made possible with a simple double loop.

Xpm accepts only images in *xpm* format, and the *convert* utility is used to perform the conversion. With the *xpm* library it is easy to find the most prevalent color, C_1 , by keeping a color counter for each unique color found in the image and returning the color with the maximal count.

Color segmentation features F_1 and F_2 are calculated in a more complicated way. To do this, we first need to define the neighbor metric to be the color distance between two adjacent pixels. We used four different neighbor metric distance measures, d_1 , d_2 , d_3 and d_4 as defined in **Figure 11**, using RGB and HSV color representations. F_1 and

¹ The motivation to introduce some of the image features described in this section is due to Lienhart and Hartmann (2002).

F_2 are defined as the fractions of pixels in the image with a neighbor metric greater than zero and 3/4 of the maximum achievable distance respectively.

$$d_1 = |r_1 - r_2| + |g_1 - g_2| + |b_1 - b_2|$$

$$d_2 = |h_1 - h_2| + |s_1 - s_2| + |v_1 - v_2|$$

$$d_3 = |h_1 - h_2| + |s_1 - s_2|$$

$$d_4 = |h_1 - h_2|$$

where r = red, g = green, b = blue, and
 h = hue, s = saturation, v = value

Fig. 11. Four ways to calculate the distance metric

To calculate F_1 and F_2 , a total of $w(h - 1) + h(w - 1)$ comparisons have to be made since the distance between each adjacent pair of pixels has to be evaluated, where w and h are the width and height of the image respectively. As depicted in the **Figure 12** below, 12 comparisons have to be made for an image with 3 x 3 pixels.

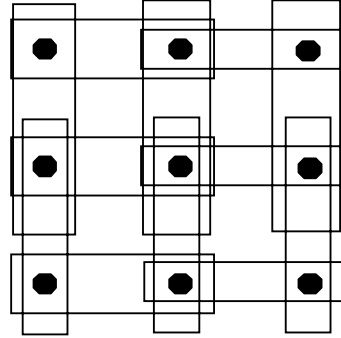


Fig. 12. 12 comparisons are made for an image with 3 x 3 pixels; each box represents a comparison of the two enclosed pixels.

4.2.3 Acquisition of other visual features using Intel OpenCV Library

Intel OpenCV, or Open Computer Vision, is a powerful image processing library consisting of a collection of algorithms and sample codes for various computer vision problems. Not only does it allow for efficient and easy direct access to pixels of images in various formats, but it also provides useful tools such as Canny edge

detector, image segmentation, connected component analysis and geometrical shape detector.

We start by introducing some simple distance measures based on color histogram. A color histogram is a representation of an image derived by counting the ‘color’ of each pixel. For example, the painting in Image 22 presents the color histogram of the image, showing the percentage of pixels that have each particular color.

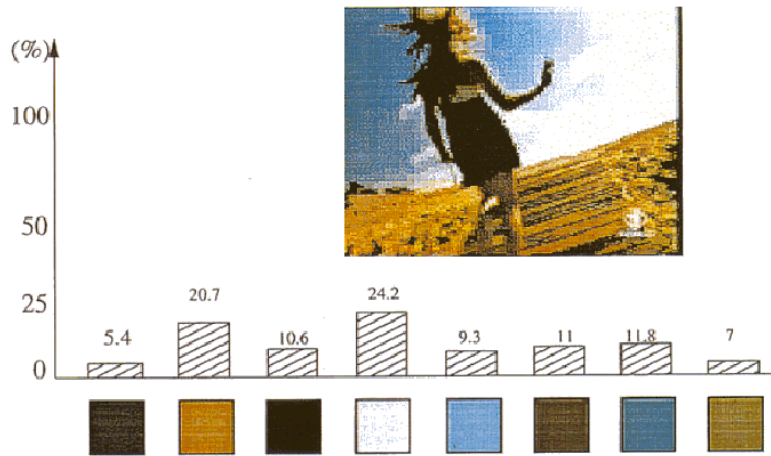


Image 22. An example image together with its color histogram.

Therefore, we are interested in this question: Is there a correlation between the color histogram of an image and its class? To answer this, we first find histograms of all training images in a class, then obtain an *average* histogram for the class. Then for each testing image, we calculate the difference between the class *average* and the image’s histogram. A total of n features are generated, where n is the number of classes in the classifier (e.g., 5 for Level 2). A number of different distance measures are used: Minkowski-form (L_1 , L_2 and L_∞), Jeffrey Divergence, Chi^2 , Quadratic distances as well as EMD. Their detailed calculations are formulated in **Table 1**.

Next, we use various OpenCV shape detectors to detect circles, rectangles, lines and corners in images. These features contribute to useful information in determining image class. Different sets of parameter settings are used for the detectors and it is left to the machine learner to decide which set makes a better classifier.

4.3 Machine learning and cross validation

After the feature vectors of images have been constructed, a decision tree is built by machine learning. We choose to use a decision list algorithm called BoosTexter (Schapire and Singer, 2000). It uses a technique called boosting to combine many simple and moderately inaccurate decision rules into a single, highly accurate decision rule. The simple rules are trained sequentially; conceptually, each rule is trained on the examples which are most difficult to classify by the preceding rules. After many rounds of boosting (say, 300 rounds, as used in NPIC), it stores the generated weak classifiers in a strong hypothesis file, *.shyp* which will be later used in testing.

Cross validation is employed in the testing process. The basic idea of cross validation is to obtain more accurate estimations about how well the machine learner predicts unseen data. This is done by setting aside some fraction of the known data and using it to test the prediction performance of the hypothesis induced from the remaining data. k-fold cross-validation means that k-experiments are run, each time setting aside a different $1/k$ of the data to test on. The final average test results reflect more stable performance measures than any single split of the labeled data into training and testing data. An algorithm is illustrated in **Figure 13**.

1. Given N examples, divide them into k equal portions, each portion containing roughly equal number of examples in each category.
2. Repeat for $i = 1$ to k
 - a) Put i^{th} portion into the BoosTexter test file.
 - b) Put the rest of the N examples into the training file.
 - c) Run BoosTexter.
 - d) Retrieve the test accuracy for each boosting round.
3. Find the average test accuracy of each boosting round across the k-runs.
4. Output the best average test accuracy across different boosting rounds.

Fig. 13. Algorithm for cross validation

5. Evaluation

We test the performance of NPIC system on two datasets of images. The first is the original corpus of 15,600 images that are automatically downloaded from Google Image Search. The second corpus consists of a subset of 1,300 images (200, 500 and 600 images for Level 1, 2 and 3, respectively) retrieved from the Wikipedia Commons. The Wikipedia Commons is a license-free repository of media files free for anyone to use in any way. These datasets are available from our NPIC website, to facilitate further research in the field¹. These datasets are entirely independent of each other.

There are several questions that we would like to answer with our evaluation:

- 1) How accurate is Google Image Search in determining image class?
- 2) How well does NPIC perform?
- 3) How do the different textual and visual features interact to achieve its performance?
- 4) How does the performance change with corpus size for different levels of classifications?
- 5) Is there any general trend in the set of mislabeled images, i.e. error cases?

The above questions will be addressed in Section 5.1 to 5.5 respectively.

5.1 Reliability of Google image retrieval

Section 3.8 makes a hypothesis on the accuracy of categories of images in the corpus as assigned by Google Image Search. Due to the considerable amount of work involved in manual annotation, we decide to assume we have been working with a clean image corpus (one that does not contain noisy data). To test the validity of this assumption, we manually annotate a subset of these images and assessed the reliability of using Google as a substitute. More specifically, random samples of 200 images are taken for each of the three levels of classification from the image corpus. The number of correctly categorized images are recorded and shown in **Table 3**.

¹ <http://wing.comp.nus.edu.sg/npic/>

Level	# total	# correct	Accuracy
Level 1	200	199	99.5%
Level 2	200	188	94%
Level 3 (Map)	200	185	92.5%
Level 3 (Figure)	200	182	91%

Table 3. Accuracy of image category labels

The above results show a strong agreement between current image labels in the corpus with their actual labels. It can be observed that as the classification granularity becomes finer, i.e. higher level of classification, the accuracy is also lower. This is consistent with our expectation.

5.2 Testing on Google and Wikipedia datasets

As mentioned above, we run experiments on NPIC using both the Google and Wikipedia datasets. Wikipedia dataset is hand-labeled by the first author. For the Google dataset, we perform a five-fold cross validation; that is, we use 4/5 of the data to train a model and 1/5 for testing, and repeat this process five times and averaging the performance. For the Wikipedia dataset, the entire Google dataset is used for training a model, and tested on the Wikipedia set. A boosted decision list learner, BoosTexter, is used as the machine learner, as its inferred rules are easy to interpret. The learner is asked to do 300 rounds of boosting (i.e., 300 serial rules inferred) for each classifier. The rules also easily lend themselves to an analysis of which features are helpful. **Table 4** shows the resulting accuracy of our experiments.

Level	Class	Average C.V. accuracy (Google)			Testing accuracy (Wikipedia)		
		Text (T)	Visual (V)	V + T	T	V	V + T
Level 1	Synthetic	99.4%	95.9%	99.9%	94%	93%	95%
	Natural	99.7%	93.5%	99.9%	90%	92%	94%
	Total	99.6%	94.7%	99.9%	92%	92.5%	94.5%
Level 2	Map	94.3%	87.6%	98.5%	78%	77%	86%
	Figure	90.5%	82.9%	98.7%	74%	78%	90%
	Icon	100%	77.6%	100%	95%	91%	96%
	Cartoon	89.2%	73.6%	97.6%	69%	84%	81%

Level 3 (Figure)	Artwork	92.5%	67.0%	93.2%	73%	74%	79%
	Total	93.3%	77.7%	97.6%	77.8%	80.8%	83.4%
	Block diagram	84%	86%	84%	72%	82%	86%
	Venn diagram	88%	86%	90%	70%	88%	90%
	Bar graph	84%	78%	82%	78%	78%	74%
	Pie chart	82%	86%	90%	80%	86%	86%
	Line graph	80%	78%	80%	66%	74%	76%
	Table	78%	68%	82%	72%	72%	76%
	Total	82.6%	80.0%	84.3%	73.1%	79.9%	81.4%
Level 3 (Map)	Plan map	86%	76%	86%	82%	78%	84%
	Chorographic map	86%	80%	88%	78%	82%	82%
	Relief map	90%	68%	84%	70%	70%	72%
	Weather map	84%	64%	84%	74%	66%	72%
	Zip code map	96%	72%	92%	88%	72%	86%
	Total	88.4%	72.0%	86.8%	78.4%	73.6%	79.2%

Table 4. Performance of NPIC on the two datasets, with different feature sets.

We observe several trends from the results. First, accuracy increases as we go from the specific Level 3 classifiers towards the Level 1 classifier. This is expected, as the Level 3 classifiers are more fine-grained and are harder 5- or 7-way decision problems. Second, accuracy on the Wikipedia dataset is lower across the board. Specifically, the textual features are less helpful than the visual ones. This is partially due to the fact that URLs are not available in this dataset and that the filenames are not nearly as indicative of the class as in the Google dataset (after all, filenames are partial evidence for relevance in Google’s image search, used to construct the dataset). The visual features show roughly the same performance on both data sets. As such, we feel that the test on the Wikipedia dataset is more realistic and representative of what would be encountered in practice. Third, maps are harder to classify than figures, as the figure subcategories have notably different visual features that are captured by the OpenCV detectors. Fourth, icons do extremely well, as their extension in Windows is a fixed .ico and we start with a clean corpus, unlike any of the other sets. Finally, although the performance is not directly comparable with prior reported results (as the problem specifications and datasets differ), the NPIC classifiers show similar performance. The advantage here is that NPIC system uses a set of very general, coarse features that are inexpensive to compute and applicable to a wide

range of problems. Classifiers aimed at specific tasks (Ng, Chang, Hsu and Tsui, 2005) are bound to do better in their stated problem domain.

Given that image search primarily employs textual features, are the improvements by incorporating visual features significant? We compared the textual versus the combined feature judgments using Student's 2-tailed T-test. Our findings indicate a significant ($p < 0.05$) for both Level 2 classifiers but not the Level 1 or 3 classifiers. We believe the reason for this is simply because there are too few images for the Level 3 classifiers (600 in total for both Level 3 classifiers) and for the Level 1 Wikipedia classifier (1000 in total).

5.3 Feature Performance

To assess the efficacy of the textual, visual and textual + visual feature sets, we explore the resulting classifiers. **Table 5** shows the first 100 features used by each of the four inferred models (with repetitions omitted).

Level	Textual Features	Visual Features
Level 1	jpeg jpg jennifer friends azoft stylefest gif map painting pie a search drawing areas iconfan serials paris online tv sponsors eastburtonhouse	C_1 , F_1 , $L^{-\infty}$, C_1 Fraction, Colors, Height, Background _{Hue}
Level 2	map painting artwork drawing ico cartoon venn graph diagram disney pie anime garfield maps physics www directory chemistry comics com world artwork art maths archie chem. page street au image tintin gif sg city hein edu books chinese asp sun moaa gov nr nice chart assembled region	Width, F_1 , #HorizontalLines, #slantedLines, F_2 , #Rectangles, Quad _{artwork} , Quad _{icon} , Height, AvgLineLength, Background _{Hue} , AveLineGradient, Size, F_2 , EMD _{diagram} , JD _{artwork} , EMD _{artwork} , X-resolution
Level 3 (Figure)	block pie venn bar table diagram data archives illustration barograph none gov charts cty us edu fag articles hisoftware en cfm pubs	#SlantedLines, #circles, $\text{Chi}^2_{\text{block}}$, AvgLineGradient, #Rectangles, #HorizontalLines, Width, X- resolution, Size, $\text{Chi}^2_{\text{diagram}}$, #VerticalLines, Colors, AvgLineLength, EMD _{block} , Background, Y-resolution, EMD _{pieChart} , JD _{pieChart} , L^{-1}_{barGraph} , L^{-2}_{graph} , EMD _{block} , Height, Quad _{block} , $L^{-\infty}_{\text{block}}$

Level 3 (Map)	weather plan relief country map us maps underground sbtvsworld graphics planning wr php province map asp files	#SlantedLines, EMD _{region} , #HorizontalLines, AvgGradient, #Corners, L-1 _{relief} , JD _{plan} , EMD _{relief} , AvgLineLength, EMD _{weather} , L-1 _{zipAreaCode} , L-1 _{relief} , EMD _{weather} , L-2 _{zipAreaCode} , QuadDistzipAreaCode, #VerticalLines, Height, EMD _{plan} , FractionOccupiedByRectangles, L-1 _{weather}
------------------	---	--

Table 5. Salient features found in the BoosTexter models.

From the above table, we see that individual words (each a separate feature) constitute a large fraction the useful features in the Level 1 and 2 classifiers, but a smaller fraction of Level 3 features (validating our earlier claim). We also see that the color histogram distance measures play a larger role in the fined-grained classifiers, and that no one distance measure is best: they all seem to be used by the classifier for discriminating in different instances. Finally, our OpenCV features have been effective for the classes we suspect: circles are used in the *figure* classifier and vertical/slanted lines in the *map* classifier (perhaps for differentiating building plans from other subclasses of maps).

For the OpenCV detectors, the learner found optimal settings through cross-validation separation. For the circle detector, a diameter setting of $d = 0.3 \times \min(\text{height}, \text{width})$ performs best, as lower settings of d would find many spurious results. The rectangle detector, on the other hand, is set to detect only ones that are parallel to the image frame, because we realized that classes such as *table*, *bar graph*, and *building plans* contain upright rectangles instead of tilted ones. The fact that BoosTexter uses this set of settings in the classifier is consistent with our expectation.

5.4 Scale testing on various corpus sizes

To measure how NPIC’s system performance varies with increasing corpus sizes, we devise this test. **Figure 14** shows that, in general, test accuracy increases with corpus sizes. This is expected, as a larger training data in the corpus could give the BoosTexter more information to build the decision tree used for testing. However, as we can also observe, the problem of over-fitting is evident from the trend that test

accuracy shows a slight decrease after the corpus size becomes larger than a certain threshold. Generally, over-fitting occurs when a machine learner picks up patterns by adjusting itself to very specific random features of the training data, which have no causal relation to the optimum decision tree it is trying to build. In this process of over-fitting, the performance on the training examples still increases while the performance on unseen testing data becomes worse.

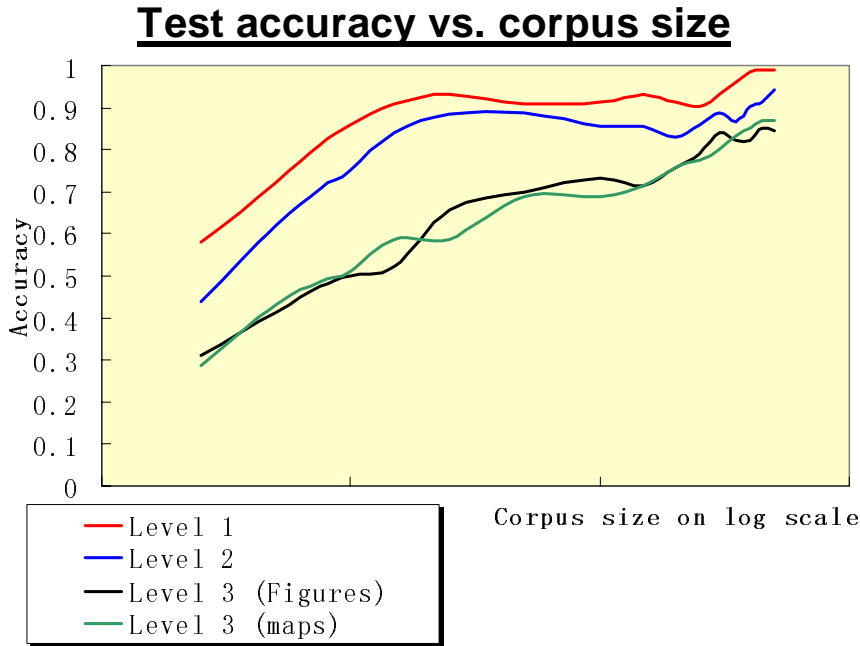


Fig. 14. Performance on different corpus sizes

5.5 An error case

After a thorough study on the cross validation results, we learn that some images are wrongly classified because they possess features that belong to other classes. For example, NPIC classifies the image (Image 22 shown below) as a *figure* but its actual label is a *map*. Through an analysis of the set of feature vectors selected by BoosTexter, we learn the following reasons that cause this misclassification:

- Image contains too few colors (*maps* normally have more colors than *figures*.).
- Image has a white background (this is not common for *maps*).
- Filename “*wrld.gif*” does not give useful information indicative of image class.

WORLD EARTHQUAKES LAST 14 DAYS
Mon Aug 16 06:01:47 CDT 2004

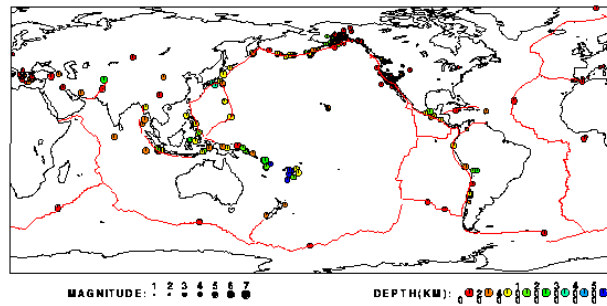


Image 23. An example error case, “wrld.gif”

6. Conclusion and future works

We have introduced NPIC, a system specifically targeting synthetic image classification. This system is fully automated and distinguishes between natural vs. synthetic images, and types synthetic images into five classes, of which *maps* and *figures* are further subdivided. We obtain the image datasets by standard text-based image search using keywords highly correlated with each class. This noisily labeled corpus serves as training data, making our classification scheme semi-supervised. In all cases, performance of the classifiers increases when simple color and shape detection features are added. A key result is that visual features make a stronger contribution than the textual ones when fine grained classification is needed.

NPIC is based on a general framework that relies on the scale of image search engines to sift away noise from the training data. We expect to further improve NPIC in the future by:

- Using the relevance ranking of the images from search engine in weighting examples for training.
- Exploring how to find keywords automatically for training data acquisition. We plan to achieve this by using mutual information which can provide a list of statistically correlated modifiers for a base keyword. We have already done a detailed error analysis on the dataset, and have additional features in mind that may help to improve performance.
- Using NLP (Natural Language Processing) techniques to improve NPIC by employing the concept of a WordNet semantic lexicon. With this, NPIC system will be able to tell, for example, an image named “*car.jpg*” and another one named “*truck.gif*” should belong to the same class, say, *vehicle*.
- Developing more efficient and effective geometrical shape detection algorithms. There is also a need for better visual features.

References

1. Cao, R. and Tan, C.L. (2001). Separation of overlapping text from graphics. The 6th International Conference on Document Analysis and Recognition (ICDAR '01), 2001, pp. 44.
2. Carberry, S., Elzer, S., Green, N., McCoy, K. and Chester, D. (2004). Extending document summarization to information graphics. ACL – 04 Workshop, 2004, pp. 3 – 9.
3. Carson, C., Belongie, S., Greenspan, H. and Malik, J. (2002). Blobworld: Image segmentation using expectation-maximization and its application to image querying. IEEE Transactions on Pattern Analysis and Machine Intelligence 24, 2002, pp. 1026–1038.
4. Getty Institute. (2006). Art and architecture thesaurus, www.getty.edu/research/conducting_research/vocabularies/aat/, 2006.
5. Hu, J. and Bagga, A. (2003). Functionality-based web image categorization. WWW, 2003.
6. Huang, W.H., Tan, C.L. and Loew, W.K. (2003). Model-based chart image recognition. International Workshop on Graphics Recognition (GREC), 2003, pp. 87 – 99.
7. Lienhart, R. and Hartmann, A. (2002). Classifying images on the web automatically. Journal of Electronic Imaging 11, 2002.
8. Ma, W.Y. and Manjunath, B. (1997). NaTra: A toolbox for navigating large image databases. IEEE International Conference on Image Processing, 1997, pp. 568–71.
9. Munson, E. and Tsymbalenko, Y. (2001). To search for images on the web, look at the text, then look at the images. The 1st international workshop on Web Document Analysis, 2001.
10. Ng, T.T., Chang, S.F., Hsu, J., Xie, L. and Tsui, M.P. (2005). Physics-motivated features for distinguishing photographic images and computer graphics. The ACM International Conference on Multimedia, 2005, pp. 239–248.
11. Qin, L., Charikar, M. and Li, K. (2004). Image similarity search with compact data structures. The 13th ACM conference on Information and knowledge management, Washington, D.C., 2004.
12. Rubner, Y., Tomasi, C. and Guibas, L.J. (2000). The earth mover's distance as a metric for image retrieval. International Journal of Computer Vision 40, 2000, pp. 99–121.
13. Schapire, R.E. and Singer, Y. (2000). Boostexter: A boosting-based system for text categorization. Machine Learning 39, 2000, pp. 135–168.

14. Smith, J.R. and Chang, S.F. (1994). Quad-tree segmentation for texture-based image query. The 2nd Annual ACM Multimedia Conference, 1994.
15. Swain, M.J., Frankel, C. and Athitsos, V. (1997). WebSeer: An image search engine for the World Wide Web. International Conference on Computer Vision and Pattern Recognition, 1997.
16. Wang, F. and Kan, M.Y. (2004). Non-photographic Image Categorization. University Research Opportunity Program (UROP) thesis, National University of Singapore, Singapore, 2004.
17. Wang, J.Z., Li, J., Chan, D. and Wiederhold, G. (1999). Semantics-sensitive retrieval for digital picture libraries. D-Lib Magazine 5, 1999.
18. Zhong, Y., Karu, K. and Jain, A. K. (1995). Locating text in complex color images. Pattern Recognition 29, 1995, pp 1523 – 1535.
19. Zhou, J. and Lopresti, D. (1997). Extracting text from www images. The 4th International Conference on Document Analysis and Recognition, Vol 1, 1997, pp. 248 – 252.

End Notes – Image references

1. *The University of Iowa, Department of Physics and Astronomy*, www.physics.uiowa.edu, 2005.
2. *American Air and Water*, www.americanairandwater.com, 2005.
3. *Ask Maps*, www.askmaps.com, 2005.
4. *BBC*, www.bbc.co.uk, 2005.
5. *University of Washington, Department of Atmospheric Sciences*, www.atmos.washington.edu, 2005.
6. *Chili Pepper*, www.chilipepper.de, 2005.
7. *Half Life*, www.half-life2.cz, 2005.
8. *Linux from Scratch*, www.linuxfromscratch.org, 2005.
9. *Sun Microsystems*, www.java.com, 2005.
10. *Conser Design & Construction*, www.conserhomes.com, 2005.
11. *Across Eurasia: London To Singapore Overland South East Asia*, www.weecheng.com, 2005.
12. *UK Ministry of Defence*, www.operations.mod.uk, 2005.
13. *Ski Press*, www.skipressworld.com, 2005.
14. *NIC Components Corps*, www.niccomp.com, 2005.
15. *Communications Research Centre*, www.crc.ca, 2005.
16. *Luis's Home Page*, www.tieguy.org, 2005.
17. *Dr. Michele Borba's Home Page*, www.micheleborba.com, 2005.
18. *Northern Rock Foundation*, www.nr-foundation.org.uk, 2005.
19. *Mac Files*, www.macfiles.org, 2005.
20. *Roberto Scano*, trace.wisc.edu, 2005.
21. *The Ex-Classics Web Site*, www.exclassics.com, 2005.

22. *Computer Science and Electrical Engineering, Oregon Health & Science University, www.cse.ogi.edu, 2005.*

23. *Earth & Atmospheric Sciences, Saint Louis University, www.eas.slu.edu, 2005.*