Finding Images and Line-Drawings in Document-Scanning Systems

Shumeet Baluja & Michele Covell
Google, Inc.
shumeet@google.com , covell@google.com

Abstract

The system presented in this paper finds images and line-drawings in scanned pages; it is a crucial processing step in the creation of a large-scale system to detect and index images found in books and historic documents. Within the scanned pages that contain both text and images, the images are found through the use of SIFT-based local-features applied to the complete scanned-page. This is followed by a novel learning system to categorize the found SIFT features into either text or image. The discrimination is based on using multiple classifiers trained via AdaBoost. Through the use of this system, we improve image detection by finding more line-drawings, graphics, and photographs, as well as by reducing the number of spurious detections due to misclassified text, discolorations, and scanning artifacts.

1 Introduction

Accurate text/figure segmentation is important to correctly infer the layout and flow of the primary narrative in scanned books, magazines, and newspapers [1][11]. In addition, once figures are accurately segmented from the surrounding text, the figures themselves are a useful tool in representing and relating books. Pages that include images are often the most useful preview pages from books [5]: they are faster for a person to skim than large amounts of non-illustrated text. Implied relationships between books and documents can also be inferred by finding shared or closely related figures and treating each as an implicit "link" from one book to another, similar to what is currently done with web documents [7]. Being able to find the non-photo-realistic drawings, in addition to photographs, extends the number books to which we can apply an image-link system - we can include historic books, manuscripts, and newsprint (Figure 1).

Our approach to detecting and indexing images is robust to the artifacts introduced by rapid large-scale book-scanning. We successfully strike a balance between missing images and spurious detections. Missing images would limit our ability to visually present the book and to create image-based links between books. It can also lead to captions and within-figure text being included in the primary flow of OCRed text. Spurious image detections can lead to poor choices of representative book pages and to incorrect linking between books.

Our approach uses a general local interest-point detector and descriptor. This operator fires on both text regions and on image regions, generating a high-dimensional description with each firing. The most naïve approach to visually linking books would be to

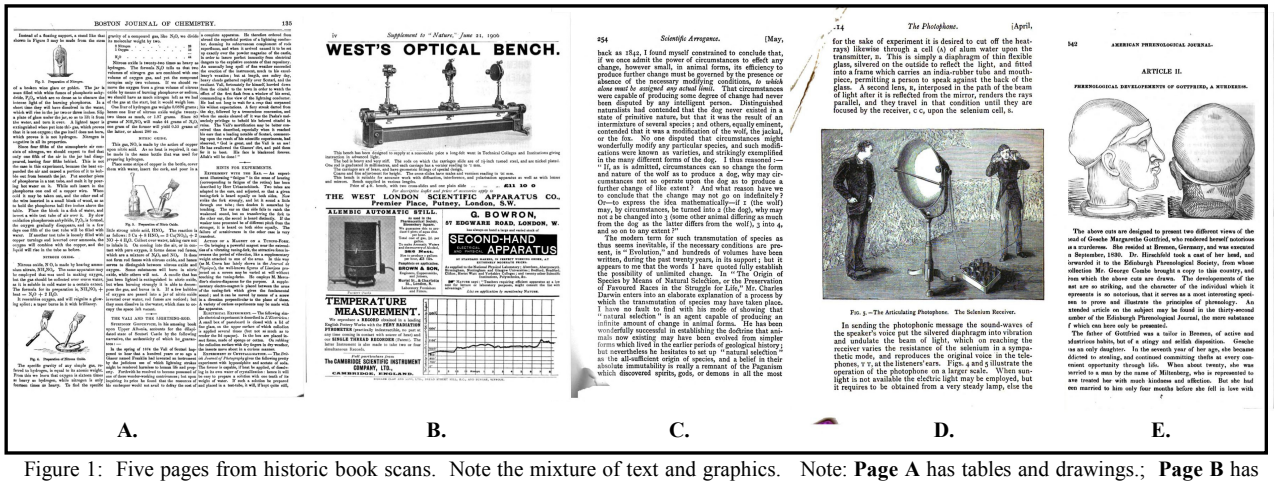


Figure 1: Five pages from historic book scans. Note the mixture of text and graphics. Note: **Page A** has tables and drawings.; **Page B** has inverted text and a large a variety of different fonts; **Pages C and D** have scanning artifacts and page discolorations that are mistaken for images by the current Google Book Scanning system. In **Page E,** text from the backside of the page shows through within the figure.

place all of these local descriptors into a database and to infer links from the preponderance of collisions between books. Even ignoring spurious matches from descriptors in textual regions, this naïve approach will not scale due to the explosive growth in the number of descriptors included in the database. Not only are more descriptors generated in text regions than in equal-sized image regions, but the majority of most book pages are devoted to text, further increasing this unnecessary overhead.

This has led us to the classification work described in this paper. In Section 2, we review local feature detectors and descriptors, with an emphasis on SIFT (used for Figure 5). In Section 3, we review the AdaBoost approach, which we used to train our individual classifiers. Since our space of possible weak classifier is so large and our training examples are so numerous, we discuss sampling methods in Section 4. Finally, we improve on the baseline classifiers using our previous subset sampling to advantage in Section 5 and by postprocessing in Section 6. Each of Sections 4, 5, and 6 provide the corresponding set of experimental results on a large real-world evaluation test. Section 7 concludes and discusses future work.

2 SIFT Features

Because images are often imbedded within large regions of text (as shown in Figure 1), global image metrics such as color histograms, or global shape analysis, do not provide enough granularity for our task. Instead, we use local image features that are rich in terms of local information content, yet stable under local and global perturbations (rotation, skew, and noise). Examples of local features include Scale Invariant Feature Transform [8], Shape Context [3], among others [10]. Although any could have been used, standard SIFT features are employed here.

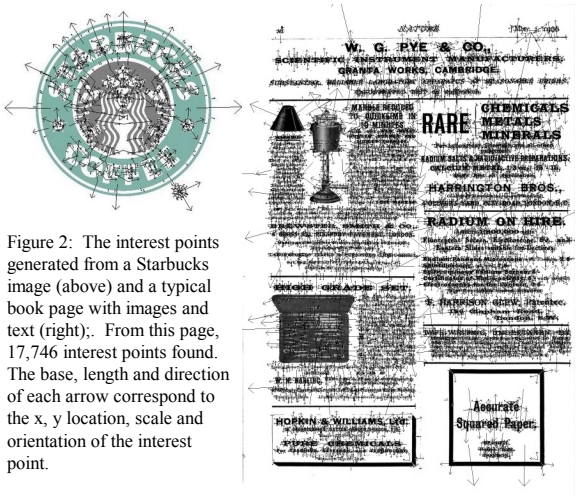


Figure 2: The interest points generated from a Starbucks image (above) and a typical book page with images and text (right);. From this page, 17,746 interest points found. The base, length and direction of each arrow correspond to the x, y location, scale and orientation of the interest point.

SIFT interest-point selection is a three-step process [8]. First, SIFT builds a pyramid of scaled images by iteratively applying Gaussian filters to the original image. Next, adjacent Gaussian images are subtracted to create Difference of Gaussian (DoG) images, from which the characteristic scale associated with each interest point can be estimated by finding the local extrema over the scale space. Given the DoG image pyramid, SIFT selects interest points located at the local extrema of 2D image space and scale space. In the final step, the features are made invariant to rotation by assigning a characteristic orientation to each of the interest points. A gradient map is computed for the region around the interest point and then divided into a collection of subregions in which an orientation histogram can be computed. See Figure 2.

Each SIFT features is represented as a 128-dimensional vector – by concatenating 4x4 orientation histograms with 8 bins (representing the gradient directions). The fundamental task is to determine which histograms (i.e., features) represent text and which represent images. To provide a concrete understanding of the task, sample histograms are shown in Figure 3. Next, we describe the AdaBoost learning procedure to discriminate between the two classes of SIFT features – text and image.

3 AdaBoost Learning

To learn a discrimination boundary between the text and image SIFT features, we use a discrete variant of AdaBoost described in [15]. It has been used successfully in a variety of vision domains, is simple to implement, and is efficient in practice. The main steps of the AdaBoost algorithm are shown in Figure 4. Essentially, AdaBoost is a greedy learner that, at each step, selects the best weak classifier for the weighted errors of the previous step (where a weak classifier performs at least slightly better than random). The weight changes, applied to the training examples in Step 4, are such that the misclassified examples receive a greater weight than the correctly classified examples. Once the weak classifiers are selected, they are combined to form a strong classifier by a weighted

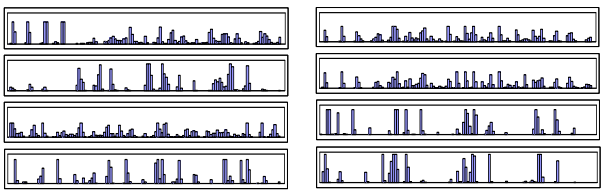


Figure 3: 128 dimensional histogram vectors taken from pages with line drawings (Left) and pages with only text (Right). 4 examples of each. The task is to distinguish the two classes.

sum, where the weights are chosen based on the errors found in each step.

One of the crucial design choices is which weak-classifiers to employ. For these experiments, we use a general classifier that applies an "allocation-mask" to each of the 128 entries in the SIFT-histogram. The mask allocates each entry in the SIFT-histogram to either bin-A, bin-B or leaves it unallocated[1]. A number of comparison functions can be used to compare bin-A and bin-B (such as the absolute/percent difference of bin-A and bin-B, the ratio of bin-A and bin-B, the ratio of the averages of bin-A and bin-B etc.) A total of 5 such comparison functions are considered. Once the allocation-mask and the comparison function are chosen, each weighted training example is run through the weak-classifier. A threshold is automatically computed that maximizes the weighted correct response. Each weak-classifier is therefore a tuple of {allocation-mask, comparison-function, threshold}.

4 AdaBoost Weak-Classifier Exploration

One of the time consuming steps in AdaBoost is computing the accuracy of all the weak classifiers in each iteration. Because the set of possible weak-classifiers to explore is enormous (3^{128} settings of the mask × 5 comparison functions), we cannot evaluate each possible classifier. Instead, we randomly select a small number of them to evaluate in each iteration (in our tests, only 7500 weak classifiers were evaluated per iteration). Additionally, instead of evaluating the classifiers on the entire training set, only a small fraction is used for training. In each iteration (after the selection of a weak-classifier), a new portion of the training set is chosen, and a new set of classifiers to evaluate is selected. Although this does not guarantee that the best weak-classifier will be chosen, it has worked well in practice in visual domains [2].

In the first set of experiments, we evaluate a randomly selected set of classifiers as has been done in previous studies– the allocation mask and the comparison function used is chosen randomly. The threshold is then set automatically based on the weighted errors. In the second approach, we use next-ascent stochastic hillclimbing (NA-SHC) [14] to select the weak classifiers to evaluate. With NA-SHC, we initialize the search with a random allocation mask and comparison function. Then, in each iteration, a random perturbation of the current individual is considered (for example, changing the comparison function or the allocation of a bin in the histogram). The new candidate is evaluated and compared to the old one. If the new candidate is better than or equal to the old solution, the new solution replaces the old one. In both approaches, 7500 weak classifiers are evaluated per iteration.

Input: samples (x_1,y_1) .. (x_n,y_n) where x_s are the SIFT features and $y_s = 0$ for those that are text and $y_s = 1$ for image.

Initialize weights $w_{1,s} = 0.5/T$, $0.5/I$ for $y_s = 0,1$ respectively, where T and I are the number of text and image SIFT samples.

For $m = 1,\ldots,M$ (maximum # of weak classifiers to use):

1. Normalize weights $w_{m,s}$ such that $\Sigma_s w_{m,s} = 1.0$
2. For each weak classifier, C_j, $(0 \le j < J)$ see how well it predicts the classification. Measure the error with respect to the weights w_m: $error_m = \Sigma_s w_{m,s} \mid C_j(x_s) - y_s \mid$
3. Choose the weak classifier (denoted C_m) with the lowest $error_m$.
4. Update the weights:
   If the example is classified incorrectly: $w_{m+1,s} = w_{m,s}$
   Else: $w_{m+1,s} = w_{m,s}B_m$
   where : $B_m = \frac{error_m}{1 - error_m}$

The result of the strong classifier is:

$$S(x): \begin{cases} 1: \sum_{m=1}^{M} \log(\frac{1}{B_m}) * C_m(x) \ge 0.5 \sum_{m=1}^{M} \log(\frac{1}{B_m}) \\ 0: \text{ otherwise} \end{cases}$$

Figure 4: AdaBoost Learning Procedure. In each iteration 7,500 classifiers are evaluated (J=7500). A total of 150 classifiers are chosen (M=150).

The incremental computational expense of using NA-SHC over random search is eclipsed by the much larger expense in evaluating the weak classifiers on the training samples. The performance of NA-SHC is, however, significantly better than random-candidate generation, as is described below.

To evaluate the learning procedures, scanned pages from hundreds of books were considered. For the training samples, pages that contained pure text were placed into class 1, and pages that contained only images (other than perhaps captions) were placed in class 2. The SIFT features from these pages were computed and used for training with AdaBoost. To test the resultant strong classifiers, a similar procedure (with pages gathered from multiple books that were outside the initial training-set) was used for testing.

A total of 8-million SIFT features (4 million from each class) were used for training. The strong-classifiers trained with AdaBoost employed a total of 150 weak classifiers. 7,500 weak-classifiers were evaluated in each iteration before the best found weak-classifier was added to the strong-classifier. For testing, separate test sets were created with a total of 600,000 SIFT features. The results are shown in

[1] Note that the form of the classifier is similar to the ones commonly used in AdaBoost in the face detection tasks [Viola, 2001]; however, no restriction is made on the spatially contiguous mask-regions as are commonly used with pixel-classification tasks.

Table 1 (column 'Single Classifier'). The difference between using random exploration and NA-SHC was statistically significant to the 99.9% level.

Table 1: Results on large test sets with Random and NA-SHC Classifier Systems. Percent correct classifications shown.

	Single Classifier	Multiple Strong Classifiers	
		3 classifiers	5 classifiers
Random Exploration	**83.1%**	**85.8%**	**86.6%**
Hillclimbing NA-SHC	**86.2%**	**89.1%**	**90.0%**

5 Using Multiple Strong Classifiers

The creation of the AdaBoost strong classifiers is a stochastic process: only a small, randomly selected, number of weak-classifiers are chosen for evaluation and the training samples are also chosen randomly. Although the trained strong classifiers have approximately similar overall performance, many errors are not systematic, and appear random. We harness this variability by using a group-of-classifiers approach [12]. We can train a suite of strong classifiers instead of a single one. In this approach, each SIFT-vector is classified by a group of classifiers. Although there are numerous methods to combine classifiers, we chose the simplest: each classifier casts a vote (image or text) for each SIFT-vector. The final classification is the simple majority.

The results are shown in Table 1. The use of multiple strong classifiers improves the performance for both the NA-SHC approach and the random exploration runs (statistically significant to the 99.9% level). The differences between the 3 & 5 classifiers were not significant in either experiment.

6 Post-Processing

In this section, we describe two post-processing heuristics used to remove the majority of the remaining errors. As shown in Figure 4, the standard implementation of AdaBoost is to set the strong-classifier's decision threshold at 50%. However, to make the system more conservative in terms of signaling an 'image' (vs. 'text') SIFT feature, the threshold can be increased. By tuning the threshold and allowing some SIFT features to be missed, the number of false-positives can be reduced dramatically. Empirically, we found that setting the threshold to 59% (9% above the standard setting of 50%) was enough to make a noticeable difference.

Despite increasing the threshold for accepting an 'image' label, a few SIFT text-features are still misclassified as images (see Figure 5, Column C). However, the majority of these are 'single' detections; they are spurious detections that are not supported by other spatially close detections. In contrast, for valid image detections, large, spatially coherent, groups of SIFT features are found above the threshold. We eliminate these isolated false-positives by requiring at least D detections within R pixels for final image detection with D=3 and R=4% of the page width.

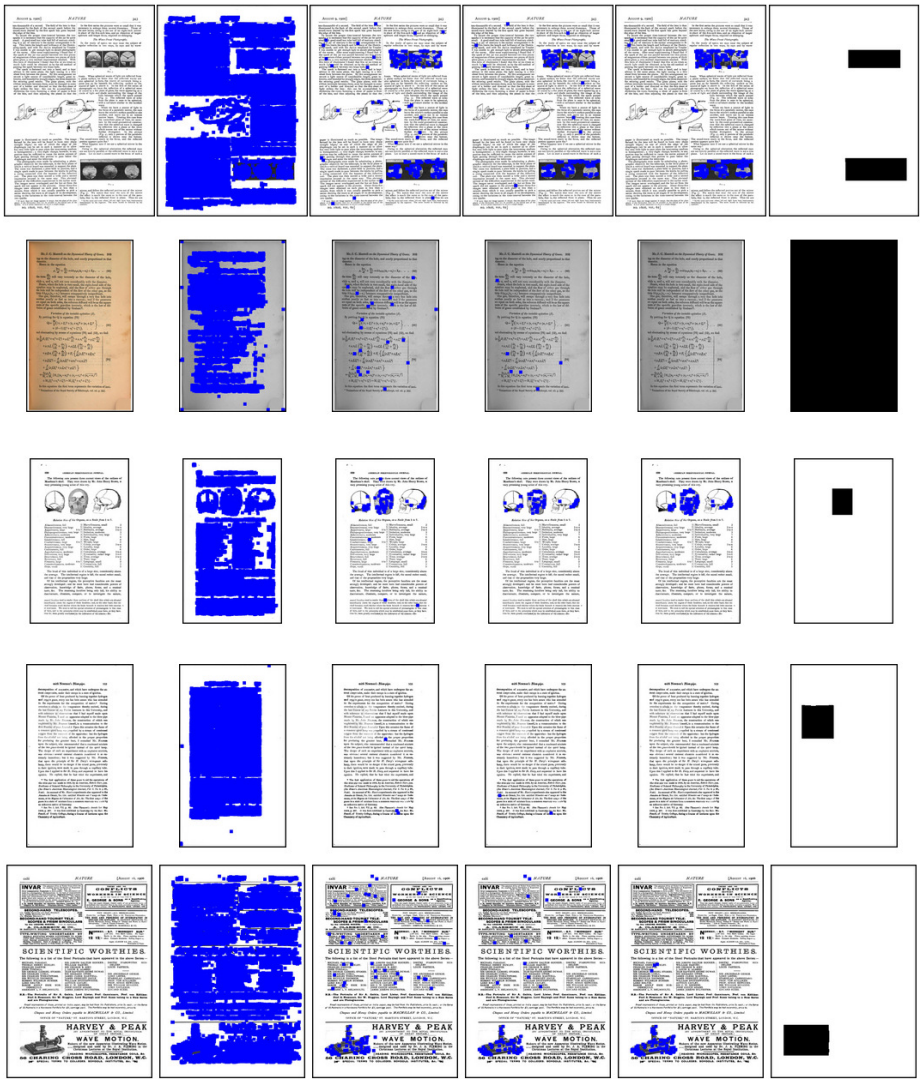
Figure 5 shows our results, compared to those of the Google Book Scanning (GBS) system. The images that were found by GBS are shown in Figure 5, Column F. Details are given in the caption.

7 Conclusions

The system presented in this paper is the first part of an online system to detect and index images found in a large-scale book-scanning system. The primary goal has been met: more of the images in the book pages have indexable points found in them using this approach than if the images had been pre-segmented with the GBS system and then the local features found. This is achieved largely without miscategorizing text or spurious features from discolorations, scanning artifacts, or page markings. This has the important effect of keeping the number of SIFT points in the database small and concentrated on the images. When a new image is found, checking it against the current database of SIFT points is significantly more efficient than if all the SIFT features from the pages were added. Being able to find the non-photo-realistic drawings extends the number books to which we can apply our system - historic books and manuscripts can be included. We can immediately use these image features in near-duplicate images detection systems.

In order to use the current system in a document layout understanding system (*e.g.* [9][11]) the points found in the images must be expanded to regions that encapsulate the entire image, not just the interest points used for indexing the image. The interest points may serve as seeds from which to grow segmentation regions [4], including through the use of texture information. This avenue is open for future study.

A future extension is to use these SIFT classifications as initial points in boundary-box determination and image extraction; this will be pursued as the need arises. Though the results are initially quite promising, comparison with other existing systems is warranted [1][9][13]. Beyond simply detecting images that have previously been encountered, there are many other immediate applications. Currently, in [5], pages with images are most commonly used for selecting preview pages from books; this technique will immediately make that more



A. Original Image | B. All SIFT Features | C. Features run through Adaboost classifiers and are above threshold | D. Remaining features after voting with Multiple Classifiers | E. Multiple Classifiers + Post-Processing | F. Images found by Google's Book scanning system.

Figure 5: 5 example pages. **Columns A**: The original image; **B:** All the detected SIFT features; **C:** All the SIFT features above threshold of 0.59. **D:** All SIFT features that match at least 2 of 3 classifiers above threshold of 0.59. **E:** Post processing included. **F:** Images found by Google book-scanning system (GBS).

Row 1: Three images in the page. GBS finds 2 of 3. We find features on all 3, but not many on the middle one. Also have two errors points above image.

Row 2: Due to page discoloration, equations and shading, GBS detects the entire page as image. We correctly determine no images.

Row 3: GBS finds 1 of 3 images. We find all 3.

Row 4: GBS mistakenly identifies small discoloration of the text as an image. We correctly find no images.

Row 5: Note that lines and different fonts are correctly ignored by both systems. GBS correctly gets the image. We identify the image, but get a small false-positive above the real image.

reliable. Further, as shown in [7], inferring a link structure between images in order to find authoritative images and books on a subject (i.e. art texts, textbooks) can be successfully addressed with this system.

6. References

[1] Antonacopoulos, A., Gatos, B.Bridson, D. (2007) "ICDAR 2007 Page Segmentation Competition", *ICDAR 2007*, 1284-1288

[2] Baluja, S., Rowley, H., (2007) "Boosting Sex Identification Performance" *IJCV*, 71:1 111-119

[3] Belongie,S., Greenspan,H., Malik,J., Puzicha. (2002) Shape matching and object recognition using shape contexts. *PAMI* 24:4

[4] Fan, J., Zeng., G, Body, M. Hacid, MS (2005), "Seeded Region Growing: An Extensive and Comparative Study", *Pattern Recognition Letters*, 26:1139-1156

[5] Google (2008) Book-Search http://www.google.com/books

[6] Jain, A.K., Yu, B. (1998) Document Representation and Its Application To Page Decomposition", *PAMI* 20:3.

[7] Jing, Y., Baluja, S. (2008) VisualRank: Applying PageRank to Large-Scale Image Search, *PAMI*, 30(11) 1877-1890.

[8] Lowe, D.G. (2004) Distinctive Image Features from Scale Invariant Keypoints. *IJCV,* 60(2):91–110, 2004

[9] Mao, S., Rosenfeld, A., Kanungo, T., (2003) "Document Structure Analysis Algorithms: A Literature. Survey" *SPIE*, 197-207

[10] Mikolajczyk, K. and Schmid, C. (2005). A performance evaluation of local descriptors. *IEEE-PAMI*, 27(10).

[11] Namboodiri, A., Jain, A.K. (2007)"Document Structure and Layout Analysis", *Digital Doc Proc: Major Dir. and Recent Adv*

[12] Ruta, D. & Gabrys, B. (2000) "An Overview of Classifier Fusion Methods", *Computing and Information Systems* 7 p1-10.

[13] Shafait, F. Keysers, D., Bruel, T. (2006) "Performance Comparison of Six Algorithms for Page Seg", *Doc. Analysis Sys VII.*

[14] Suarez, A.R., Rodriguez, A.O., Sebag, M. (1999) Automatic Graph Drawing and Stochastic Hill Climbing *Genetic and Evo. Cf.,*

[15] Viola, P. and Jones, M. "Robust real-time object detection". (2001) *Proc. IEEE Wkshp on Stati and Comp. Theo of Vision.*