Context Based Multiscale Classification of Images *

Jia Li
EE Department
Stanford Univ., CA 94305
jiali@isl.stanford.edu

Robert M. Gray
EE Department
Stanford Univ., CA 94305
rmgray@stanford.edu

Abstract

This paper presents an algorithm to segment images into four classes: background, photograph, text and graph. There are two important aspects about the algorithm. The first is that the algorithm takes a multiscale approach, which adaptively classifies an image at different resolutions. The multiscale structure enables accurate classification at class boundaries as well as fast classification overall. The second is that the context information, which is accumulated in the process of classification, is used to improve the classification accuracy.

1 Introduction

The classification problem we consider here is to segment images into four classes: background, photograph, text, and graph. One direct motivation of the problem is to improve printing quality by applying different algorithms to different types of images. By photograph, we mean a continuous-tone image. The definition for text is rather narrow here. It means normal bi-level text. The graph class includes artificial graphs and overlays, produced by computer drawing tools.

A critical issue in classification is how localized a classifier should be to decide the class of an area. In general, if an image region is of a pure class, the larger it is, the easier it is to decide its class. It is usually difficult, even for human beings, to classify a small region without knowing what is around it. Actually, it may be impossible to judge a small region's class without consideration of its context since the same region may appear in different class area. Generally speaking, the smaller a region is, the more context information we need to classify it. Based on this thought, we design a classifier, which begins with large blocks and no context and, if necessary, then moves to smaller blocks using context extracted from larger blocks. Initially, only blocks with extreme features are classified. As it is probable that unclassified blocks are at the boundaries of classes, the classifier increases its scale, i.e., subdivides the blocks in its further classification. As some context information has been obtained from the classification at the first scale, the context can compensate for the smaller block size at the higher scale. To sum up, the classifier classifies from low scale to high scale, i.e., decreases the block size by half, at every turn, and keeps accumulating context information. By such a strategy, the classifier will have more context information when it classifies smaller blocks. The idea of classifying across several scales has been applied in previous work [1, 2], but our viewpoint is very different. A typical way to benefit from multiscale analysis is to form features across several scales so that the properties of an image block at several resolutions can be incorporated into classification. In our algorithm, the multiscale structure is used to efficiently build context information. The number of scales used is naturally adaptive so as to avoid unnecessary computation due to multiscale calculations.

2 The Algorithm

2.1 Global structure

The classification features we use in this algorithm are the two statistical characteristics of the wavelet coefficients in high frequency bands, which are developed in detail in [3]. The first one, denoted by $\bar{\chi}^2$, is the goodness of matching between the observed distribution and the Laplacian distribution, and the second one, denoted by L, is the likelihood of the wavelet coefficients having a highly discrete distribution.

The overall structure of the classification algorithm is shown by a flow chart given in Fig. 1. The details of each step are explained later. To understand the flow chart, we need to clarify notation. We refer to different scales as different resolutions, denoted by r. The maximum resolution allowed in classification is set by the customer and is denoted by R. As shown in the flow chart, most operations are repeated at different resolu-

*This work was supported by the National Science Foundation under NSF Grant No. MIP-931190 and by a gift from Hewlett-Packard, Inc.

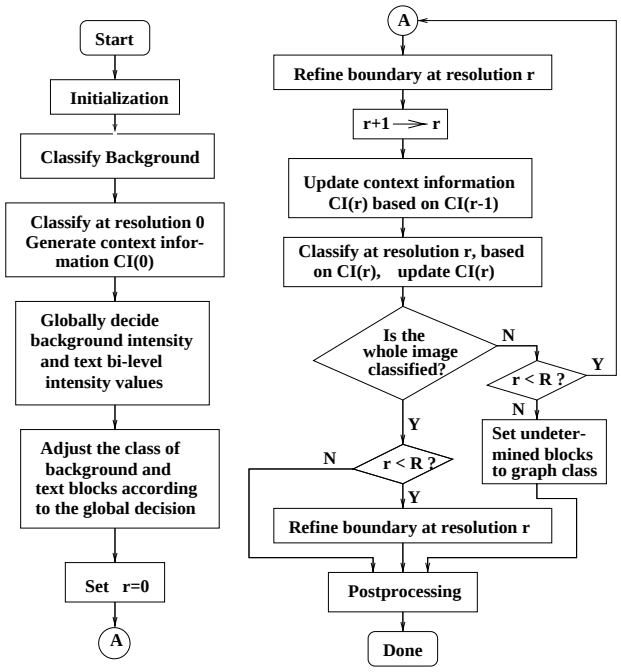

Figure 1: The flow chart of the classification algorithm

tions. We start with resolution zero and increase it by one until the whole image is classified or the resolution exceeds R. With the resolution increased by one, the width and height of image blocks are reduced by half. At every resolution, if the features of an image block strongly suggest that it is purely text, graph or photograph, then the block is classified to the specific class. Otherwise, the block is labeled as undetermined. We do not need to consider background class here because background blocks are identified before the multiscale classification, as shown in Fig. 1. It is proper to classify background without the help of multiscale structure since background blocks are simply the ones with unique intensities. A higher resolution classification is applied if undetermined blocks exist.

The object $CI(r)$ represents the context information already achieved at resolution r. It is essentially a matrix with every element storing the characteristic statistics of an classified image block. The dimension of $CI(r)$ increases with r when the image is divided into more blocks at higher resolutions. At the beginning of classification at a certain resolution r, the context information $CI(r)$ is basically inherited from the context information built at previous resolution, i.e., $CI(r-1)$. However, instead of being fixed through the classification at resolution r, $CI(r)$ is updated for every newly classified block. The characteristic statistics of every newly classified block will be added to $CI(r)$ to serve as context information for later classification.

2.2 First run classification

After classifying blocks with unique intensities as background, the algorithm processes classification at resolution zero. We start with block size S. For every $S \times S$ block, we classify it as photograph, graph or text if its features strongly suggest its class. Otherwise, the block will be marked as undetermined. For every block classified, its characteristic statistics will be stored in $CI(0)$. In the list below, we give the conditions for a block to be a certain class and the corresponding characteristic statistics stored in $CI(0)$ for every class. Recall that the classification features we use are the goodness of fit to the Laplacian distribution, denoted by $\bar{\chi}^2$, and the likelihood of having a highly discrete distribution, denoted by L.

Conditions to be a certain class:

1. Photograph: $\bar{\chi}^2 < C_\chi$, where C_χ is a chosen threshold.

2. Text: $L = 1$ and the pixel intensities in the block concentrate at two values, i.e., nearly bi-level image block.

3. Graph: $L = 1$ and the number of concentration values are more than 2; or, $C_l < L < 1$, where C_l is a chosen threshold.

Characteristic statistics stored in $CI(0)$ for the three classes:

1. Photograph: $\bar{\chi}^2$, L, mean value and standard deviation.
2. Graph: $\bar{\chi}^2$, L and mean value .
3. Text: The bi-level values.

2.3 Global decision on background and text

To achieve globally consistent classification, our classifier will globally analyze the image according to the result from the first run classification. Global information is often necessary even for human classification. The classifier can learn what should be a proper background intensity and what should be proper text bi-level values for the image. After the determination of the global background intensity and text bi-level values, the original background blocks with different intensities are changed to class graph or class text, and the original text blocks with different bi-level values are changed to class graph.

2.4 Boundary refinement

As mentioned before, a drawback of classifying based on a large image block is that a large block is more likely to contain blended classes and have one class dominate the others, which in turn leads the classifier to mark it as a single class. This drawback is the main constraint for the classifier to use large blocks. On the other hand, large blocks provide more stable feature values. This fact becomes more crucial when the graph class is added in the classification, because the graph class, being an intermediate type between text and photograph, has comparatively vague features. To solve the conflict, we apply a boundary refinement mechanism at every resolution. For all the blocks which are newly classified at a certain resolution, if a block has neighboring blocks of different classes, adjustment is done to refine the boundary between the two classes.

2.5 Update context information

At the beginning of classification for every resolution r, the context information matrix $CI(r)$ is basically inherited from $CI(r-1)$. As every block at resolution $r-1$ is divided into four subblocks at resolution r, the four subblocks will copy the characteristic statistics of the original block if the original block has been classified. One may notice that the class of a subblock may be different from the class of the original block because of the boundary refinement. Also because of the boundary refinement, a subblock may contain mixed classes. Consequently, in some cases, the characteristic statistics is recalculated instead of copying from that of the original block.

2.6 Context based classification

Suppose the current resolution is r and the context information is thus $CI(r)$, the classifier will scan the image and classify all the undetermined blocks which neighbors to blocks classified already. These blocks will be classified based on both their features and the context information provided by their classified neighbors. If a block cannot be classified even with the context information, it will still be marked as undetermined. Otherwise, it is classified and its statistics will be added to $CI(r)$. After one scan, since extra blocks are classified and $CI(r)$ is updated, the classifier will repeatedly scan the image to classify the remaining undetermined blocks in the same way as before. The process will not stop until the whole image is classified or no more blocks can be classified based on the current context $CI(r)$, which is equivalent as $CI(r)$ has no new information to update in the scan. In this case, the algorithm will proceed to a higher resolution.

Before we present the context based classification process in the list below, we give two definitions. One is the micro classifier $\mathbb{C}$. It is used to classify a block based on the classes and characteristic statistics of its adjacent blocks. By adjacent blocks, we refer to the four blocks above , below, on the right and on the left of a block. The details about the $\mathbb{C}$ are explained after the main algorithm. Another definition is the context list $\mathfrak{L}$, which is a list of records about the classes of the adjacent blocks of all the unclassified blocks. Hence, for every unclassified block, there is a corresponding element in $\mathfrak{L}$, which contains the classes of its four adjacent blocks. We use r to denote the current classification resolution and N to denote the total number of image blocks at this resolution.

1. Let $0 \rightarrow m$, $1 \rightarrow j$.
2. If $j \leq N$, proceed the following steps; otherwise, go to step 3.
   (a) If B_j is unclassified, proceed the following steps; otherwise, go to step 2b.
      i. If B_j is adjacent to a classified block, proceed the following steps; otherwise, go to step 2(a)ii.
         A. Use micro classifier $\mathbb{C}$ to classify B_j.
         B. If $\mathbb{C}$ can decide the class of B_j, store the characteristic statistics of B_j to the context information matrix $CI(r)$ and let $m+1 \rightarrow m$.

Figure 2: One sample image and its classification result image. The error rate is 3.64%. White: photograph, light gray: graph, dark gray: text, black: background.

C. If $\mathbb{C}$ can not decide the class of B_j, add the block B_j to the list $\mathfrak{L}$.

D. Go to step 2b.

ii. Add block B_j to the list $\mathfrak{L}$.

(b) Let $j+1 \rightarrow j$, go back to step 2.

3. If $\mathfrak{L}$ is not empty and $m > 0$, proceed the following; otherwise, go to step 4.

(a) Let $0 \rightarrow m$.

(b) Update $\mathfrak{L}$:
For every B_i in $\mathfrak{L}$, if B_i has newly classified adjacent blocks, i.e., blocks classified after the addition of B_i to $\mathfrak{L}$ or the latest updating of B_i in $\mathfrak{L}$, update B_i's record about the classes of its adjacent blocks.

(c) If the updated $\mathfrak{L}$ is different from the previous one, i.e., there exists at least one block B_i in $\mathfrak{L}$ which has new record about the classes of its adjacent blocks,

i. Set pointer $\mathbf{P}$ to the first element in $\mathfrak{L}$

ii. If $\mathbf{P}$ is NOT at the end of $\mathfrak{L}$, proceed the following steps; otherwise, go back to step 3.

A. Suppose the element pointed by $\mathbf{P}$ is the record of block B_i.

B. If B_i has newly classified adjacent blocks,
first, use micro classifier $\mathbb{C}$ to classify B_i,
second, if the class of B_i is decided by $\mathbb{C}$, store the characteristic statistics of B_i to the context information matrix $CI(r)$, delete B_i from list $\mathfrak{L}$, and let $m+1 \rightarrow m$.

C. Move $\mathbf{P}$ to the next element in $\mathfrak{L}$, go back to step 3(c)ii.

(d) Go back to step 3.

4. Stop.

Now we explain the details of the micro classifier $\mathbb{C}$, i.e., how the classifier makes decisions combined with context information. In particular, when classifying a block, the classifier takes the classes and statistical properties of the adjacent blocks into account. This information is available in $CI(r)$. The specific decision rule for each class is listed below.

1. Adjacent to a text block:
If B is bi-level and its two values are the same as those of the text block, it is classified as text, otherwise, no decision is made.

2. Adjacent to a graph block:
If the feature L of block B is close to that of the graph block and the mean value of B is also close to that of the graph block, block B is classified as graph, otherwise, no decision is made.

Image ID	1	2	3	4	5	6	7	8	9
P_e	5.65%	0.000%	0.108%	3.64%	11.0%	1.19%	0.841%	14.5%	0.000%

Table 1: Ratios of classification errors with respect to human labeling for 9 sample images

3. Adjacent to a photograph block:
   If the mean value of B is close to that of the photograph block and its L is not very close to 1 (extreme value for text and graph), it is classified as photograph, otherwise, no decision is made.

One may wonder what would happen if a block is adjacent to several classes and is classified as different classes based on its neighbors. In this case, conflict may happen between graph and photograph, or text and graph. We set priorities to solve the conflict. In our algorithm, text has higher priority than graph and photograph has higher priority than graph. The reason to set higher priority to text is that the requirement for a block to be text is very strict. If the requirement is met, the probability of making mistake is low. The reason to set photograph prior to graph comes from practical considerations. If a graph is scanned or printed by a photograph standard, its quality is not likely to be lowered, but not vice versa.

2.7 Post processing

The post processing removes 'spikes' at the boundary of two classes so that the boundary is smooth and the separation of any two classes is distinct. It also cleans small isolated class regions. Because of the multiscale structure, the chance of 'spikes' appearing at boundaries and small isolated class regions is actually very low. Experiments show that even without post processing, the classified images are usually clean. As a result, the load on post processing is generally small. Thus, the post processing takes rather small amount of time.

3 Results

We experimented the algorithm on 9 images, with sizes around 1650×1275, on a sparc 10 workstation. The classification error rates with respect to human labeling are listed in Table 1. The time to classify any of the images ranges from 25 to 40 seconds. The average processing time for every image is 29 seconds. One classification example is shown in Fig. 2. Note that in Fig. 2, as the image is printed in grey-scale, colored text cannot be shown here. The colored text is defined as graph in our truth set.

References

[1] Gross, M. H., Koch, R., Lippert, L. and Dreger, A., "Multiscale Image Texture Analysis in Wavelet Spaces," *Proceedings of 1st International Conference on Image Processing*, vol. 3, Nov. 1994.

[2] Mark Tabb and Narendra Ahuja, "Multiscale Image Segmentation by Integrated Edge and Region Detection," *IEEE Transactions on Image Processing*, vol. 6, no. 5, pp. 642-655, May 1997.

[3] Jia Li and Robert M. Gray, "Text and Picture Segmentation by the Distribution Analysis of Wavelet Coefficients," *Proceedings of International Conference on Image Processing*, to appear, Oct. 1998.