Visible Watermark Detection in Images

Saurabh Gupta
Senior Analytics Specialist, Opera Solutions, India

Note: Proprietary and Confidential content has been removed/blurred

Abstract

The paper describes the process of identifying images watermarked by an online Rentals listing website. A new algorithm is presented which integrates edge detection to extract the candidate watermark features and then identifies these candidates by using Random Forests classifier to help the client prevent any misuse of images from their websites.

1. Introduction

Watermarking has been proposed as the means for copyright protection of multimedia data. This work concentrates on the problem of extracting images and video frames from the web and checking if a pre-defined watermark is present on those images.

However, watermark detection and identification is a difficult task because background, color and size of watermarks may vary. Many papers show that available linearization methods, including global and adaptive thresholding (which have been well used in identifying characters printed on clean papers) do not work well for typical image and video frames. Furthermore, the image digitalization and compression also introduce noise that may blur the embedded watermark.

In the present paper, we introduce a fast and robust algorithm of object (watermark) identification in image and video frames. The three main components of the detection algorithm are: 1) Initialization: Assuming that the watermark is present on bottom right corner of the image, the basic idea is to crop a certain area proportionate to the image resolution that might contain the watermark. 2) Image Processing: Subsequently, the cropped region is passed through various Image Processing steps like Noise Reduction, Edge Detection and Hysteresis Thresholding which allow background regions of the image to be quickly discarded while candidate watermark regions are extracted by using edge analysis.

Third Internal Technical and Analytical Conference of Opera Solutions (TACOS 2014), April 17^{th} – 18^{th} , 2014, San Diego, USA.

3) Learning: In the third step, the candidate watermark is identified by using a Random Forest classifier trained using Watermark-Positive and Watermark-negative images.

The details of these 3 steps are described in Section 2,3 and 4 respectively. Experiments and results are shown in Section 5. The discussion and conclusion are in the last section.

2. Initialization

Initial processing of the image involves conversion of colored image to grayscale and cropping of candidate watermark region. Cropping area is selected based on the assumption that the size of watermark is proportional to the size of image.

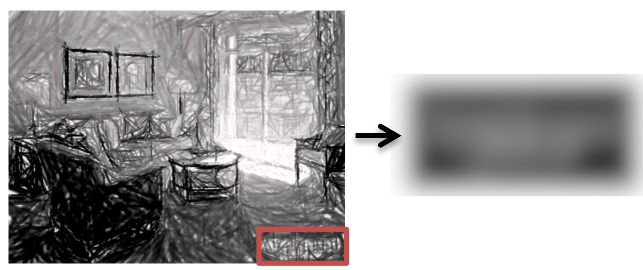

Figure 1. Conversion of image to grayscale and cropping of bottom right area containing candidate watermark region.

3. Image Processing

In this section, we present an algorithm to quickly extract candidate watermark regions in images by exploiting intensity gradient between the edges of watermark and image background.

3.1 Noise Reduction

Since edge detection is susceptible to noise in the image, first step is to remove the noise in the image. Image smoothing is achieved by convolving the image with a low-pass filter kernel. It actually removes high frequency content (noise, edges) from the image.

In our algorithm, we have employed Bilateral Filtering, since it is highly effective in noise removal while keeping edges sharp. Although, other methods like Gaussian filtering are faster, but they blur the edges also, which we don't want to.

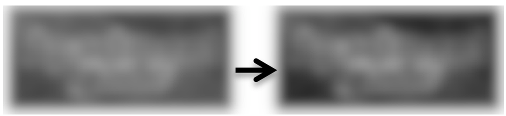

Figure 2. Bilateral filtering removes noise while retaining the edges.

3.2 Finding Intensity Gradient of the Image

Smoothened image is then filtered with a Sobel kernel in both horizontal and vertical direction to get first derivative in horizontal direction (G_x) and vertical direction (G_y). From these two derivatives, we can find edge gradient and direction for each pixel as follows:

$$\text{Edge\_Gradient}\ (G) = \sqrt{G_x^2 + G_y^2}$$

$$\text{Angle}\ (\theta) = \tan^{-1}\left(\frac{G_y}{G_x}\right)$$

Gradient direction is always perpendicular to edges. It is rounded to one of four angles representing vertical, horizontal and two diagonal directions

3.3 Non-maximum Suppression

After getting gradient magnitude and direction, a full scan of image is done to remove any unwanted pixels which may not constitute the edge. For this, at every pixel, pixel is checked if it is a local maximum in its neighborhood in the direction of gradient.

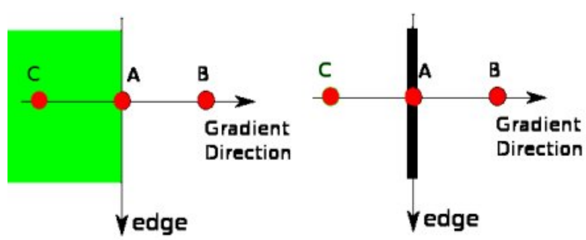

Figure 3. Edge detection using Gradient direction

Point A is on the edge (in vertical direction). Gradient direction is normal to the edge. Point B and C are in gradient directions. So Point A is checked with point B and C to see if it forms a local maximum. If so, it is considered for next stage, otherwise, it is suppressed (put to zero).

3.4 Hysteresis Thresholding

This stage decides which are all edges are really edges and which are not. For this, we need two threshold values, minVal and maxVal. Any edges with intensity gradient more than maxVal are sure to be edges and those below minVal are sure to be non-edges, so discarded.

Those who lie between these two thresholds are classified edges or non-edges based on their connectivity. If they are connected to "sure-edge" pixels, they are considered to be part of edges. Otherwise, they are also discarded.

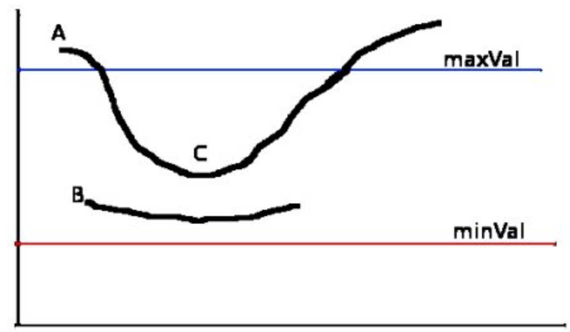

Figure 4. Retention of edges depending on intensity gradient

The edge A is above the maxVal, so considered as "sure-edge". Although edge C is below maxVal, it is connected to edge A, so that also considered as valid edge and we get that full curve. But edge B, although it is above minVal and is in same region as that of edge C, it is not connected to any "sure-edge", so that is discarded. So it is very important that we have to select minVal and maxVal accordingly to get the correct result.

This stage also removes small pixels noises on the assumption that edges are long lines.

For current algorithm, we chose maxVal=80 and minVal=50. So what we finally get is strong edges in the image.

Figure 5. Watermark image after all the transformations

4. Random Forest Classifier – Based Watermark Identification

4.1 Normalization and Feature Extraction

We first normalize the images containing candidate watermark edges, which may have varying resolutions, to rectangles of 110px X 50x by using bilinear interpolation.

This image matrix is converted to a single vector by appending the columns one after another. The feature space we used has 5500 dimensions corresponding to a 110x50 slide window in the normalized watermark region. The region has only 2 types of pixel values: 0 or 255.

4.2 Model Training

The Random Forest Classifier was trained on a database consisting of 300 image samples, with 150 labeled as watermark-positive and the other 150 labeled as watermark-negative.

5. Experiments and Results

Experiments were carried out on a database consisting of 100 images of varying resolutions scraped from different websites. Each image was in JPEG format and was decompressed and converted to grayscale before applying watermark identification.

Table 1 summarizes the performance of the proposed algorithm on test data. The model shows high accuracy with ROC curve area = .98.

Table 1. Results of model validation on dataset consisting of 100 images

		Watermark positive	**Watermark negative**
Test Outcome	**Test outcome positive**	True positive = 54	False positive = 0
	Test outcome negative	False negative = 4	True negative = 42
		Sensitivity = .931	Specificity = 1

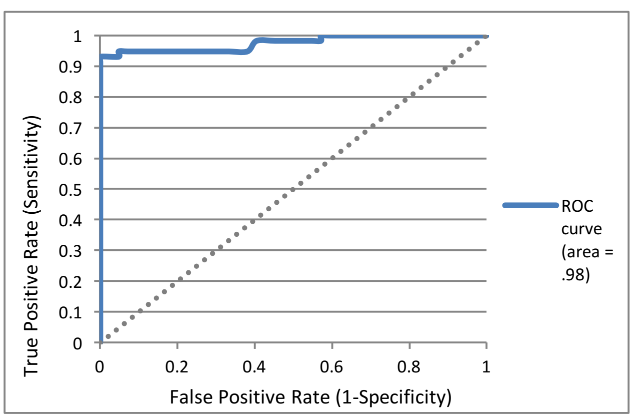

Figure 6. ROC curve to evaluate the quality of the output of watermark classifier

6. Discussion and Conclusion

Watermark identification in image with complex background and compression effects is a challenging problem. In this paper, we have presented an algorithm based on Random Forests classifier. The algorithm first integrates edge detection to extract the candidate watermark features and then identifies these candidates by using Random Forests.

The algorithm presented here achieves high identification rate. However, to identify any misuse of multimedia data and to protect copyright material, watermarking images overtly may not be the best idea. Many steganography algorithms have been developed to embed information covertly in a noise-tolerant signal such as image data. The hidden information or Digital watermark can then be used to identify ownership of the copyright of such signal.

Acknowledgments

I sincerely thank Rohit Sinha for his comments and suggestions on this work.

References

K. Jain and B.Yu, *Automatic text localisation in images and video frames*, Pattern Recognition, 31(12):2055-2076, 1998

V. Wu, R.Manmatha, and E.M. Riseman, *Finding text in images*, In Proc. ACM Int. Conf. Digital Libraries, 1997.

L. O'Gorman and R.Kasturi, *Document Image Analysis*, IEEE Computer Society Press, Los Alamitos, 1995.

Y. Zhong, K. Karu, and A.K. Jain, *Locating text in complex color images*, Pattern Recognition, 28(10):1523-1536, 1995

J. Ohya, A. Shio and S. Aksmatsu, *Recognition characters in scene images*, IEEE Trans. Pattern Analysis and Machine Intelligence, 16(2):214-220, 1994