SUB-IMAGE VERIFICATION USING HIDDEN MARKOV MODELS

Shawn Hymel (Virginia Tech, Blacksburg, VA, USA; hymelsr@vt.edu)

ABSTRACT

Image verification and recognition is an ever-growing field in the realm of computer vision. The use of hidden Markov models (HMMs) has seen wide adoption in face and handwriting recognition. The work presented in this paper expands on such research and proposes a method to model a specific sub-image using the 2-dimensional discrete cosine transform (2D-DCT) and HMMs. This model is then used to identify sub-images in a larger image with similar statistical properties with a 70% detection accuracy.

1. INTRODUCTION

Image recognition is a popular area of research with myriad applications in defense, law enforcement, and commercial sectors. These applications focus on a specific image, such as a face, a particular inanimate object, or a signature, for recognition. Face recognition, in particular, is a popular subject with multiple implementations including the correlation method [1], the eigenface method [2], linear discriminant method [3], neural networks [4], and Hidden Markov Models (HMM) [5]. Almost all proposed methods of face recognition assume that the face is the primary image or the methods perform techniques to locate and crop the desired image. The face is then compared to one or more known mathematical models in a database. The face whose model provides the closest fit to the image in question is determined to be the best match to that image.

With the success of HMM-based pattern recognition, we have developed a technique to construct a mathematical model of an arbitrary sub-image and then locate that sub-image in a larger image (i.e. sub-image verification and location).

2. RELATED WORK

Substantial work has been accomplished in computer-driven face recognition using different algorithms. HMMs have been used for face recognition [5], gesture recognition [6], speech recognition [7], and handwriting recognition [8].

Samaria [5] proposes a top-down as well as a pseudo-2D approach to classifying faces with HMMs. For both approaches, he trains an HMM using the grayscale pixel values from a sliding window in the image. All training and test images were the same size and contained no background. The top-down approach successfully matched the faces with 84% accuracy with a test set of 50 images of 24 individuals. The pseudo-2D approach achieved 95.5% accuracy with 40 subjects, each with 5 images for training and 5 images for testing.

Nefian and Hayes [9] used a similar top-down HMM approach as Samaria, but introduced the concept of using the discrete cosine transform (DCT) on the sliding window in order to reduce the size of the observation vector. The method proposed in this paper achieved a 84% accuracy with 400 images of 40 individuals and required less computation time than [5].

Kohir and Desai [10] used a sliding window in a left-to-right and top-down fashion to produce a set of DCT coefficients. These coefficients were used to train and identify a set of faces. By adjusting the size of the window, they were able to achieve a 99.5% accuracy of successfully identifying 40 subjects among 400 images.

3. ANALYSIS

The principles of the DCT, Gaussian Mixture Models (GMMs), and HMMs are covered in this section.

3.1. Discrete Cosine Transform (DCT)

The discrete cosine transform expresses the frequency of data points as a sum of cosines. The DCT is similar to the Fourier transform but uses only real numbers. The 2-dimensional DCT for an $E \times E$ matrix is given as:

$$c(u,v) = \alpha(u)\alpha(v)\sum_{x=0}^{E-1}\sum_{y=0}^{E-1} f(x,y)\cdot cos\left(\frac{(2x+1)u\pi}{2E}\right) cos\left(\frac{(2y+1)v\pi}{2E}\right) \quad (1)$$

where,

$$\alpha(\omega) = \begin{cases} \frac{1}{\sqrt{E}} \; for\ \omega = 0 \\ \frac{2}{\sqrt{E}} \; for\ \omega = 1, 2, \dots, E-1 \end{cases} \quad (2)$$

The DCT is widely used in compression (e.g. MP3, JPEG) where higher frequency components can be removed while still retaining the bulk of the information.

3.2. Gaussian Mixture Model (GMM)

A Gaussian Mixture Model is a set of one or more Gaussian distributions used to define a set of probabilities. The GMM is given by

$$\Pr[\boldsymbol{x}] = \sum_{i=1}^{G} w_i g(\boldsymbol{x}|\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i) \tag{3}$$

where $\boldsymbol{x}$ is a D-dimensional observation vector, w_i, $i = 1,2,...,G$ are the weights for each Gaussian distribution in the mixture, and $g(\boldsymbol{x}|\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i)$, $i = 1, 2, ..., G$ is the Gaussian distribution for each component, i. Note that the component weights, w_i, must sum to 1.

The Gaussian distribution for a D-dimensional vector is described by the equation:

$$g(\boldsymbol{x}|\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i) = \frac{1}{(2\pi)^{\frac{D}{2}}|\boldsymbol{\Sigma}_i|^{\frac{1}{2}}} \cdot$$

$$\exp\left\{-\frac{1}{2}(\mathbf{x} - \boldsymbol{\mu}_i)'^{\boldsymbol{\Sigma}_i^{-1}}(\mathbf{x} - \boldsymbol{\mu}_i)\right\} \tag{4}$$

with $\boldsymbol{\mu}_i$ and $\boldsymbol{\Sigma}_i$ as the mean and covariance matrix, respectively. We will represent the GMM as the triplet:

$$\Gamma = (\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i, w_i),\ i = 1,2, ... G \tag{5}$$

Given a set of data, we can estimate the parameters of a GMM using the Expectation Maximization (EM) method defined in [12]. The EM method for GMMs is an iterative process, whereby the next set of parameters are computed by

$$\overline{w}_i = \frac{1}{T}\sum_{t=1}^{T} \Pr\,[i|\boldsymbol{x}_t, \Gamma] \tag{6}$$

$$\overline{\boldsymbol{\mu}}_i = \frac{\sum_{t=1}^{T} \Pr\,[i|\boldsymbol{x}_t,\Gamma]\boldsymbol{x}_t}{\sum_{t=1}^{T} \Pr\,[i|\boldsymbol{x}_t,\Gamma]} \tag{7}$$

$$\overline{\sigma}_i^2 = \frac{\sum_{t=1}^{T} \Pr\,[i|\boldsymbol{x}_t,\Gamma]x_t^2}{\sum_{t=1}^{T} \Pr\,[i|\boldsymbol{x}_t,\Gamma]} - \overline{\mu}_i^2 \tag{8}$$

where σ, x, and μ are arbitrary elements of the arrays **σ**, **x**, and **μ**. Note that the *a posteriori* probability for each i is given by the equation

$$\Pr\,[i|\boldsymbol{x}_t, \Gamma] = \frac{w_i g(x|\boldsymbol{\mu}_i,\boldsymbol{\Sigma}_i)}{\sum_{k=1}^{M} w_k g(x|\boldsymbol{\mu}_k,\boldsymbol{\Sigma}_k)} \tag{9}$$

3.3. Hidden Markov Model (HMM)

A Markov model is a mathematical modeling tool that gives observation output probabilities based on a set of states. In a hidden Markov model, these states are unobservable. The model is based on a Markov chain, which exhibits the Markov property:

$$\Pr[X_n = x_n \mid X_{n-1} = x_{n-1} ... X_0 = x_0] = \Pr[X_n = x_n \mid X_{n-1} = x_{n-1}] \tag{10}$$

The discrete-valued HMM is given by the triplet as described by Rabiner [11]:

$$\lambda = (\mathbf{A}, \mathbf{B}, \mathbf{\Pi}) \tag{11}$$

A describes an $N \times N$ matrix that gives the probabilities of transitioning between N states. **B** is an $M \times N$ matrix that contains the probabilities of the model producing 1 of M observables given a particular state. **Π** is a $1 \times N$ matrix that gives the probability of the model initially entering into 1 of N states.

Instead of using the discrete-valued **B** matrix, we replace the output probabilities with a probability density function (*pdf*) for each state to describe continuous-valued observation vectors. One way to accomplish this is by using a mixture of Gaussians for each state. If we assume the same number of Gaussians in each state, then we can represent the HMM as

$$\lambda = (\mathbf{A}, \mathbf{\Pi}, \boldsymbol{\mu}, \mathbf{\Sigma}, \mathbf{W}) \tag{12}$$

where **A** and **Π** are the same matrices from the discrete-valued HMM, **μ** contains the means for all the Gaussians in each state, **Σ** describes the covariance matrices for each state, and **W** is the set of mixture weights in each state. Note that for multi-dimensional, GMM-based HMMs, N describes the number of states in the HMM, D is the number of dimensions in each observation, and G is the number of Gaussians in each state.

The three canonical HMM problems as described by Rabiner in [13] are:

Problem 1: Find the probability that a given model will produce a known observation sequence: $\Pr[\boldsymbol{O}|\lambda]$.

Problem 2: Find the most likely state sequence with a known model and observation sequence.

Problem 3: Adjust the parameters of a model to best match a given observation sequence in order to maximize $\Pr[\boldsymbol{O}|\lambda]$.

Problem 1 is an attempt to evaluate a model's fit to an observation sequence. This is accomplished using the Forward-Backward algorithm.

Problem 2 tries to uncover the "hidden" set of states that produced a set of observables. This can be solved using the Viterbi algorithm. While we do not focus on the Viterbi algorithm, more information can be found in [14].

In problem 3, we wish to train a model to a set of observations. This is accomplished using a particular implementation of Expectation Maximization, known as the Baum-Welch algorithm (BWA).

We focus on the continuous valued versions of the Forward-Backward algorithm and Baum-Welch algorithm for the purposes of pattern recognition.

3.4. Forward-Backward Algorithm

The Forward-Backward algorithm attempts to compute the probability of a given model producing a known observation sequence. The definition of the Forward-Backward algorithm is as follows:

1) Initialization:

$$\alpha_1(j) = \pi_j b_j(\boldsymbol{o}_1),\ j = 1,..,N \tag{13}$$

$$\beta_T(j) = 1,\ j = 1,..,N \tag{14}$$

2) Induction:

$$\alpha_{t+1}(j) = \left[\sum_{i=1}^{N} \alpha_t(i) a_{ij}\right] b_j(\boldsymbol{o}_{t+1}) \quad t = 1,..,T-1,\ j = 1,..,N \tag{15}$$

$$\beta_t(j) = \sum_{j=1}^{N} a_{ij}\, b_j(\boldsymbol{o}_{t+1}) \beta_{t+1}(j) \quad t = T-1,\ldots,1,\ i = 1,..,N \tag{16}$$

3) Termination:

$$\Pr[\boldsymbol{O}|\lambda] = \sum_{i=1}^{N} \alpha_T(i) = \sum_{i=1}^{N} \beta_1(i) \tag{17}$$

Note that the observation $\boldsymbol{o}$ is a D-dimensional vector. Additionally, $b_j(\boldsymbol{o}_t)$ is the probability that the observation $\boldsymbol{o}$ is produced given a state, q. This value can be computed via the *pdf* at $\boldsymbol{o}$ of the GMM for state q, which is given by (3).

3.5. Baum-Welch Algorithm (BWA)

The BWA is used to re-estimate the parameters of a HMM in order to better fit a given observation sequence. Using the values of α and β previously computed in the Forward-Backward algorithm, we can redefine the HMM as follows:

Step 0: Start with initial model λ

Step 1: Compute α and β variables using the Forward-Backward algorithm

Step 2: Compute γ variables

$$\gamma_t(j) = \frac{\alpha_t(j)\beta_t(j)}{\sum_{i=1}^{N} \alpha_t(i)\beta_t(i)} \tag{18}$$

Step 3: Compute ξ variables

$$\xi_t(i,j) = \frac{\alpha_t(i) a_{ij} b_j(\boldsymbol{o}_{t+1})\beta_{t+1}(j)}{\sum_{i=1}^{N}\sum_{j=1}^{N} \alpha_t(i) a_{ij} b_j(\boldsymbol{o}_{t+1})\beta_{t+1}(j)} \tag{19}$$

Step 4: Re-estimate HMM parameters

$$\bar{\pi}_i = \gamma_1(i) \tag{20}$$

$$\bar{a}_{ij} = \frac{\sum_{t=1}^{T-1} \xi_t(i,j)}{\sum_{t=1}^{T-1} \gamma_t(i)} \tag{21}$$

$$\bar{\boldsymbol{\mu}}_i = \frac{\sum_{t=1}^{T} \gamma_t(i) \cdot \boldsymbol{o}_t}{\sum_{t=1}^{T} \gamma_t(i)} \tag{22}$$

$$\bar{\boldsymbol{\Sigma}}_i = \frac{\sum_{t=1}^{T} \gamma_t(i) \cdot (\boldsymbol{o}_t - \bar{\boldsymbol{\mu}}_i)(\boldsymbol{o}_t - \bar{\boldsymbol{\mu}}_i)'}{\sum_{t=1}^{T} \gamma_t(i)} \tag{23}$$

The new parameters create a new model, $\bar{\lambda}$, which is fed back into the BWA to further refine $\Pr[\boldsymbol{O}|\lambda]$. It should be noted that finding the global maximum for $\Pr[\boldsymbol{O}|\lambda]$ is intractable. Therefore, experimentation with the initial set of model parameters is advisable.

4. FEATURE EXTRACTION

The sub-image that was used for this specific test is the popular character Waldo from the children's book, "Where's Waldo?" [15]. One image of Waldo was chosen as the training image (figure 1). A sampling window of 20 × 5 pixels scanned over the 20 × 40 image of Waldo in a top-to-bottom approach. For each increment, the window moved down a single pixel, thus creating an observation sequence of 36 observation vectors.

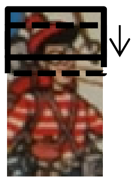

Figure 1: Sub-image sampling using a sliding window

The 2D-DCT was computed for each sampled window, producing a set of DCT coefficients for each color channel: red, green, and blue (figure 2).

Similar to the method detailed in [10], the 2D-DCT coefficients were extracted to create an observation vector from each set of DCT coefficients. In this case, only the first 3 coefficients were used from each color channel, as they contained the most information about the frequency of that color in the window. Figure 3 shows the DCT coefficient sampling process.

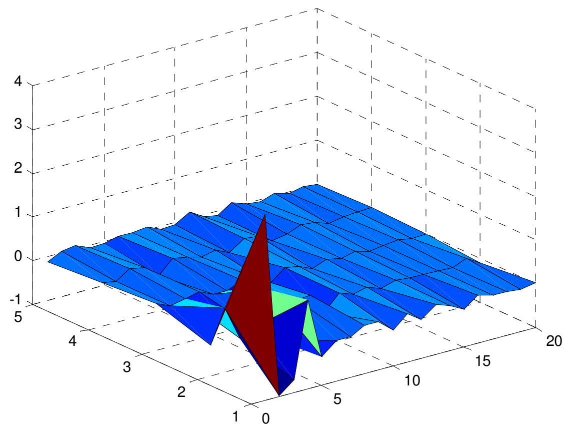

Figure 2: Example of 2D-DCT for the red channel of the sampling window

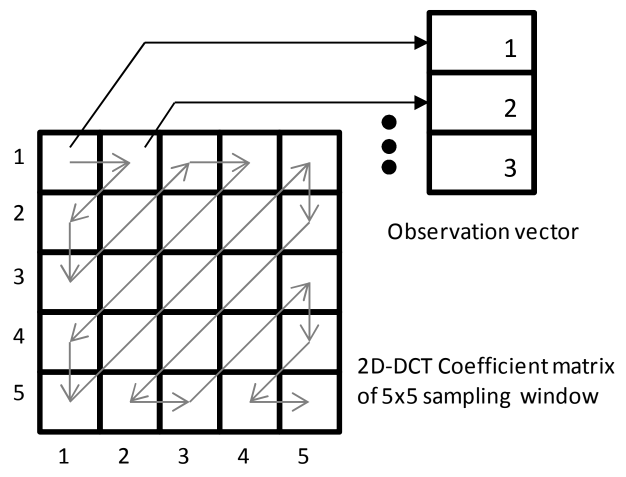

Figure 3: Example of forming a 3-dimensional observation vector from a 5 × 5 sampling window

Each of the 3-dimensional observation vectors from each color channel were concatenated to form one observation sequence of 9-dimensional vectors and 36 vectors long. Figure 4 depicts an example of this observation sequence.

This observation sequence was used to train several GMMs, which were then used to train an HMM.

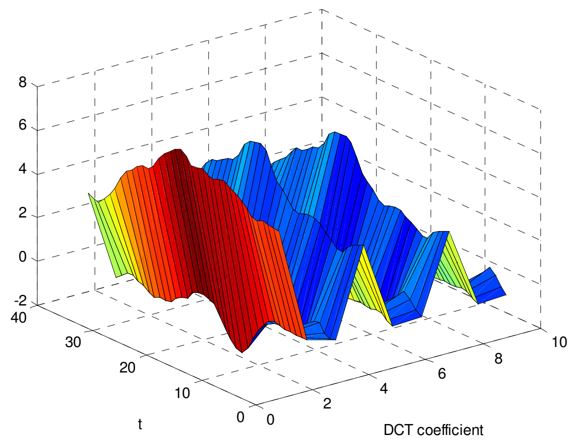

Figure 4: Observation sequence from sampling an entire sub-image

5. MODEL TRAINING

Using the extracted training sequence, we can train several GMMs and a single HMM. The HMM becomes a statistical model for the image of Waldo. Figure 5 shows this training process.

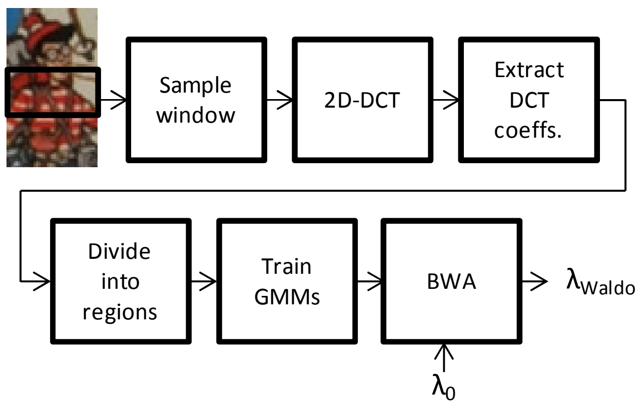

Figure 5: GMM and HMM training process

5.1. GMM Training

Three HMM states were chosen based on subjective intuition. Each state contained a particular region of the sub-image. As per figure 6, this consisted of the hat, face, and shirt regions.

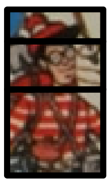

Figure 6: Sub-image regions

The DCT of each region was found using the methods outlined in section 4. The 9-dimension observation vectors from each region were used to train a GMM consisting of 2 Gaussians using the EM method (section 3.2) for that region.

5.2. HMM Training

Using the methods defined in section 3.5, we can train an HMM with the mean vector ($\boldsymbol{\mu}$), covariance matrices ($\boldsymbol{\Sigma}$), and weight distributions ($\mathbf{W}$) from the GMMs found in section 5.1.

Using 3 states (one for each region), we assume a left-to-right HMM topology in order to model the top-down sampling approach. We subjectively choose the starting parameters for $\mathbf{A}$ and $\boldsymbol{\pi}$:

$$\mathbf{A} = \begin{bmatrix} 0.5 & 0.5 & 0 \\ 0 & 0.5 & 0.5 \\ 0 & 0 & 1 \end{bmatrix}$$

$$\boldsymbol{\pi} = [1 \quad 0 \quad 0]$$

which states that the model must enter the first state. It then can stay in the first state or move to the second, and once in the second state, may stay or move to the third. Finally, the model must stay in the third state until the end of the observation sequence.

Using the GMM parameters and the initial model, we use the BWA to adjust the parameters of the HMM until we reach a local maximum for $\Pr[\boldsymbol{O}|\lambda]$. The final HMM parameters are stored for later access.

6. SUB-IMAGE VERIFICATION

In order to "find Waldo", as per the request of the children's books, we scan the larger image using a different window (the same size as the original sub-image used for training) and compare the observation sequence of each search window to the stored HMM (figure 7).

The full image is scanned in a left-to-right fashion, moving to the right by one pixel each time. At the end of the line, the window is moved down one pixel and the next line is scanned. As a result, each search window overlaps the previous window by all but one line of pixels.

For each search window, we scan down the window (section 4) using a scanning window. The 2D-DCT of each scan window is computed, and a set of 36 9-dimensional observation vectors are produced. Using the stored Forward-Backward algorithm, we compute $\Pr[\boldsymbol{O}|\lambda]$ with the extracted observation sequence. This probability is compared to a threshold, P_{th}, and if it exceeds the threshold, we conclude that the search window contains Waldo or a sub-image with similar statistical properties. Figure 8 depicts the verification process.

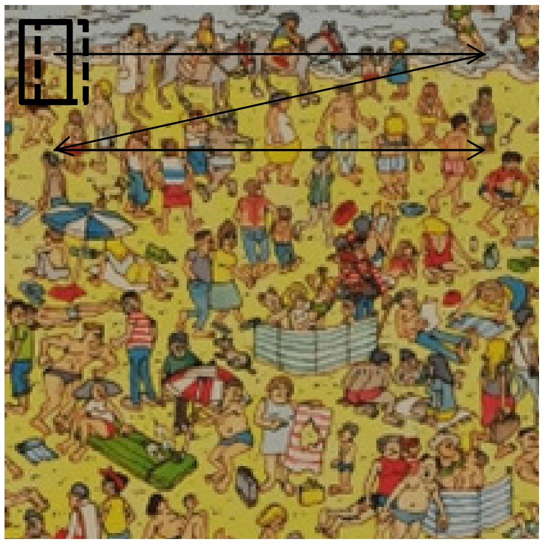
Figure 7: Full image with search window

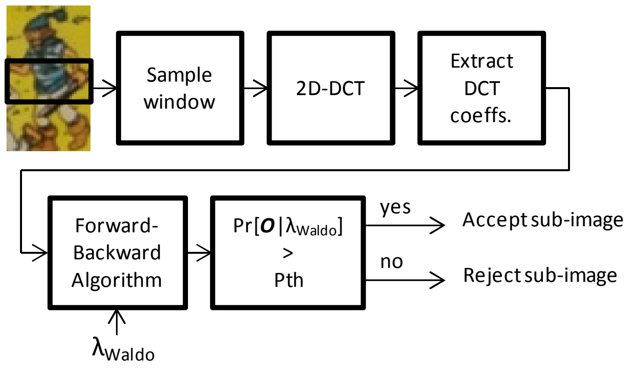

Figure 8: Sub-image verification process

7. TEST PROCEDURES

To test the functionality of the verification process, we chose an image of Waldo on a white background from a photograph of an original piece of Waldo artwork (see figure 1 for this image). This image was used to train an HMM, λ_{Waldo}. An arbitrary threshold was chosen, and another image (200×200 pixels) was used for verification purposes. This full image contained a cropped section of a larger "Where's Waldo?" image, and was guaranteed to contain a sub-image of Waldo. Any sub-image in this full image that exceeded the threshold was highlighted blue.

To test the accuracy of the verification process, the test procedure was repeated on 10 separate full images (200×200 pixels), all of which contained a sub-image of

Waldo. Each sub-image of each full image was considered a single test.

The (x, y) coordinates of Waldo for each full image were known and used as the ground truth. Therefore, 10 sub-images were considered true sub-images of Waldo among the 10 full images. All other sub-images were considered to be negative hits. The true-positive, true-negative, false-positive, and false-negative rates were computed for a set of thresholds ranging from 0 to -9000 (log-likelihood).

8. TEST RESULTS

As shown in figure 9, the verification process successfully highlighted several sub-images around the Waldo character.

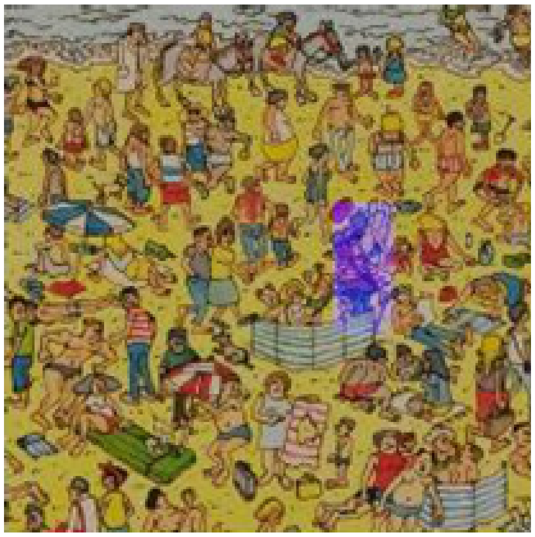
Figure 9: Highlighting matching sub-images

Ten similar full images were used to test the accuracy of the verification process. The receiver operating characteristic (ROC) curve is shown in figure 10.

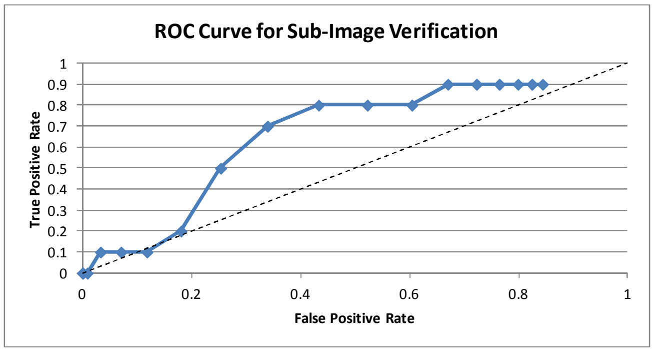

Figure 10: ROC curve

As shown by the figure, a threshold of -4500 leads to a true hit rate of about 70% with a false positive rate of about 34%. Note that many of these false positives can include areas around the desired sub-image. For the purposes of sub-image locating, many false hits can be considered "good enough" to allow the operator to quickly find the desired region.

9. CONCLUSIONS

Using a threshold of -4500, we can achieve a fairly high recognition rate (70%) of sub-images in a larger image. Depending on the conditions, the threshold may be adjusted to allow for more or less hits in the full image.

While the example focused on the famous children's book, these techniques can be used to locate or track sub-images in any number of larger images. For example, searching for a person among a crowd or a particular object among many objects would benefit from the techniques presented in this paper With a human operator, these techniques can be used to locate a number of desired sub-images with similar properties and allow the user to determine which object or sub-image to direct their focus.

10. REFERENCES

[1] D. Beymer, "Face recognition under varying pose," *Proc. of IEEE*, vol. 83, May 1995.

[2] M. Turk and A. Pentland, "Face recognition using eignfaces," *Proc. of Intl. Conf. on Pattern Recognition*, pp. 586 - 591, 1991.

[3] P. Belhumeur, J. Hespanha, and D. Kreigman, "Eigenfaces vs Fisherfaces: Recognition using class specific linear projection," *Proc. of Fourth European Conf. on Computer Vision*, pp. 45-56, April 1996.

[4] S. Lawarance, C. L. Giles, A. C. Tsoi, and a. D. Back, "Face recognition: A convolutional neural-network approach," *IEE Trans. on Neural Networks*, vol. 8, no. 1, pp. 98-113, Jan. 1997.

[5] F. S. Samaria, "Face recognition using hidden Markov models," Ph.D. dissertation, University of Cambridge, 1995.

[6] Theme Section - Face and Gesture Recognition, *IEEE Pattern Recognition and Machine Intelligence*, vol. 19, no. 5, July 1997.

[7] L. Rabiner and B. H. Juang, *Fundamentals of Speech Recognition*, Englewood Cliffs, New Jersey: Prentice Hall, 1993.

[8] J. Hu, M. K. Brown, and W. Turin, "HMM based on-line handwriting recognition," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 18, no. 10, pp. 1039-1045, Oct. 1996.

[9] A. V. Nefian and M. H. Hayes, III, "Hidden Markov models for face recognition," *Proc. of the 1998 IEEE Intl. Conf. on Acoustics, Speech, and Signal Processing*, vol. 5, pp. 2721 - 2724, May 1998.

[10] V. V. Kohir and U. B. Desai, "Face recognition using a DCT-HMM approach," *Fourth IEEE Workshop on Applications of Computer Vision*, pp. 226 - 231, Oct. 1998.

[11] L. Rabiner, "A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition," *Proceedings of the IEEE*, vol. 77, no. 2, Feb. 1989.

[12] A. P. Dempster, N. M. Laird, and D. B. Rubin, "Maximum likelihood from incomplete data via the EM method," *Journal of the Royal Statistical Society*, vol. 39, no. 1, pp. 1 - 38, 1977.

[13] L. Rabiner, "A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition," *Proceedings of the IEEE*, vol. 77, no. 2, Feb. 1989.

[14] A. J. Viterbi, "Error Bounds for Convolutional Codes and an Asymptotically Optimum Decoding Algorithm," *IEEE Transactions on Information Theory*, vol. IT-13, pp. 260-269, April 1967.

[15] M. Handford, *Where's Waldo*, Somerville, Massachusetts: Candlewick Press, 1997.