Hindawi Publishing Corporation
Mathematical Problems in Engineering
Volume 2016, Article ID 5289413, 6 pages
http://dx.doi.org/10.1155/2016/5289413

Research Article

A Modified Adaboost Algorithm to Reduce False Positives in Face Detection

Cesar Niyomugabo, Hyo-rim Choi, and Tae Yong Kim

GSAIM, Chung-Ang University, Seoul, Republic of Korea

Correspondence should be addressed to Tae Yong Kim; kimty@cau.ac.kr

Received 3 June 2016; Accepted 7 September 2016

Academic Editor: Jinyang Liang

We present a modified Adaboost algorithm in face detection, which aims at an accurate algorithm to reduce false-positive detection rates. We built a new Adaboost weighting system that considers the total error of weak classifiers and classification probability. The probability was determined by computing both positive and negative classification errors for each weak classifier. The new weighting system gives higher weights to weak classifiers with the best positive classifications, which reduces false positives during detection. Experimental results reveal that the original Adaboost and the proposed method have comparable face detection rate performances, and the false-positive results were reduced almost four times using the proposed method.

1. Introduction

Face detection is a computer technology that determines the location, size, and posture of the face in a given image or video sequence [1]. Face detection is an active research area in the computer vision community; locating a human face in an image plays a key role in applications like face recognition, video surveillance, human computer interfaces, database management, and querying image databases [2, 3]. The most successful face detection method was developed by Viola and Jones based on the Adaboost learning algorithm [4]. Although their method was successful in face detection, it faces false alarm challenges, which may increase in the presence of a complex background. False positives in an application can be a source of errors and need additional postprocessing to remove them. As our contribution to reduce the number of false positives, we propose a probabilistic approach to modify the weighting system of the Adaboost algorithm, which includes the expansion of key ideas and supporting experimental results over the preliminary version [5].

2. Face Detection and Adaboost

A number of promising face detection algorithms have been published. Among these, the Adaboost method stands out because it is often referred to by other face detection studies. In this section we present the outline and main points of some of face detection algorithms.

2.1. Face Detection. Madhuranath developed the "*modified Adaboost for face detection.*" In their method, multiple strong classifiers based on different Haar-like types trained on the same set of input images are combined into a single modified-strong classifier [6]. Viola and Jones [4] presented the fundamentals of their face detection algorithm. This algorithm only detects frontal upright faces; however, a modified algorithm was presented in 2003 that detects profile and rotated views [7]. In "Face Detection Using a Neural Network" [8], the authors computed an image pyramid to detect faces at multiple scales. A fixed size subwindow was subjected to each image in the pyramid. The content of the subwindow is corrected for nonuniform lightening and subjected to histogram equalization. The processed content is passed to parallel neural networks that carry out the actual face detection. The outputs are combined using logical AND to reduce the amount false detection rate. In its first form this algorithm also only detects frontal upright faces.

Schneiderman and Kanade [9] calculated an image pyramid and fixed size subwindow scans through each layer of this pyramid. The content of the subwindow was wavelet

analyzed and histograms were prepared for the different wavelet coefficients. These coefficients were fed to different trained parallel detectors sensitive to various orientations of the object. The orientation of the object is determined by the detector that yields the highest output. In contrast to the basic Viola–Jones algorithm, this algorithm detects profile views. AL-Allaf reviewed face detection studies based on different ANN approaches [10], whereas C. S. Patil and A. J. Patil combined skin color information and Support Vector Machine to detect faces [11].

One of the fundamental problems with real-time object detection is that the size and position of a given object within an image are unknown. An image pyramid was computed to overcome this obstacle, and a detector scans each image in the pyramid. However, this process is rather time consuming; thus, Viola and Jones presented a new approach based on Adaboost algorithm to solve the problem. However, one of the disadvantages was a high false-positive rate.

2.2. Adaboost Learning Algorithm. First introduced by Freund and Schapire [12], the Adaboost algorithm is short for Adaptive Boosting. This algorithm strengthens overall performance when used with other weak learning algorithms. A weak learning algorithm consists of a learning algorithm that classifies the input data better than random.

The Adaboost algorithm is adaptive in that misclassified data from previous classifiers are boosted during training by assigning them higher weight than that of the correctly classified data. The training database is input data set and associated classification labels. Adaboost repeatedly calls a weak learning algorithm over the training data set. Most optimal parameters of weak learning algorithms are computed at each stage, which minimizes the classification error. A weak learning classifier with optimal parameters at a given training stage is called a best weak classifier [4]. The input data set is initially weighted equally; however, the weak learning algorithm puts emphasis on the misclassified data more than the correctly classified data during the training process. This is accomplished by raising the weights of the misclassified data during each stage with respect to the correctly classified data. The main steps for the Adaboost algorithm to classify data efficiently are presented in the following.

Pseudocode for the Adaboost Learning Algorithm

(1) Given data and corresponding labels $(x_1, y_1), \ldots, (x_n, y_n)$ where x is the input data and $y = 1, 0$ are the labels for positive and negative examples with the size of the input data being n.

(2) Initialize weights w of the input data equally; $w_{1,i} = 1/2n$, where $i = 1, \ldots, n$ and i is the ith index of the data.

(3) For $t = 1, \ldots, T$, where T is the number of stages of training,

(3.1) normalize the weights w of the tth stage:

$$w_{t,i} = \frac{w_{t,i}}{\sum_{j=1}^{j=n} w_{t,j}}, \tag{1}$$

(3.2) select the best weak classifier $h_t()$ in terms of parameters of the tth stage r_t which minimizes the classification error between the weak classifier output $h_t(x_i, r_t)$ over the ith index of the data and the corresponding label y_i, over all indices I of the data, $I = 1, \ldots, n$:

$$\varepsilon_t = \sum_{i=1}^{i=n} w_{t,i} \left| h_t(x_i, r_t) - y_i \right|, \tag{2}$$

(3.3) compute the tth stage exponent β_t;

$$\beta_t = \frac{\varepsilon_t}{1 - \varepsilon_t}, \tag{3}$$

a weak classifier $h()$, which classifies the input data better than random will result in ε_t which is less than 0.5; thus β_t will be less than 1.0,

(3.4) classify the ith index of the data x_i with this weak classifier $h_t()$, compare with the actual label y_i, and store the error in classification e_i over all indices $I = 1, \ldots, n$, of the data:

$$e_i = \begin{cases} 0, & \text{if } h_t(x_i, r_t) = y_i \\ 1; & \text{if } h_t(x_i, r_t) \neq y_i, \end{cases} \tag{4}$$

(3.5) update the weights w of the input data during this tth stage using the exponents β_t computed in step (3.3). Since the weights are being normalized in step (3.1) the weights of the incorrectly classified data are boosted; this is the basic idea behind Adaboost.

$$w_{t+1,i} = w_{t,i} \beta_t^{1-e_i}. \tag{5}$$

(4) The final strong classifier, $C(x)$, is the weighted majority of the individual weak classifiers chosen in step (3.2) of each stage t.

$$C(x) = \begin{cases} 1; & \text{if } \sum_{t=1}^{T} \alpha_t h_t(x, r_t) \geq \frac{1}{2} \sum_{t=1}^{T} \alpha_t \\ 0; & \text{otherwise,} \end{cases} \tag{6}$$

where $C(x)$ is the strong classifier, $\alpha_t = \log(1/\beta_t)$ is the weight, $h_t(x, r_t)$ is the weak classifier of the tth stage, x is the input data, r_t are the parameters, and $h_t()$ is the Haar-like feature type.

Freund and Schapire [12] showed that the training error on the final hypothesis is upper bounded and if the individual weak hypothesis classifies the input data better than random, the training error decreased exponentially with an increase in the number of stages. The generalization property of Adaboost is a gradient-descent method in the space of weak classifiers, as shown by Schapire and Singer [13].

2.3. Face Classification Using Adaboost. Faces and nonfaces with their corresponding labels become the input data set for the Adaboost algorithm. Each Haar-like feature $h()$ is evaluated as the sum of the pixels in the image corresponding to the white portion subtracted from the sum of the pixels in the image corresponding to the black portion. A weak classifier has been correctly classified, if the value of the Haar-like feature at a particular location (x, y) on the image is greater than the threshold θ, and the polarity, p, determines the sign of this inequality in

$$p * h(x, y) > p * \theta. \tag{7}$$

Classification of the weak classifier depends on the evaluation of the Haar-like feature $h()$ on the image H at the location (x, y), the threshold θ, and polarity p. Image H, location (x, y), polarity p, and threshold θ are the weak classifier parameters, and the output of the weak classifier is represented as $h(H, x, y, p, \theta)$. Thus, the data in *the Adaboost learning algorithm* is image H, and the parameters become the location, polarity, and threshold (x, y, p, and θ). During the training stages of a single weak classifier, the Haar-like feature $h()$ is evaluated at each location (x, y) across all n images H_i, $i = 1, \ldots, n$. The weighted sum of the error between the correct and the actual classification $|h_t(H_i, x, y, p, \theta) - y_i|$ for all location (x, y) is computed. The location (x_t, y_t), threshold θ_t, and polarity p_t of the Haar-like feature $h_t()$ of the tth stage, which minimizes the weighted error, are chosen as the weak classifier parameters $h_t(H_i, x_t, y_t, p_t, \theta_t)$. This minimum weighted error for the tth stage, represented as ε_t, is shown in

$$\varepsilon = \min_{x,y,p,\theta} \sum_{i=1}^{i=n} w_i \left| h_t(H_i, x_i, y_i, p_i, \theta_i) - y_i \right|. \tag{8}$$

The exponent β_t used for updating the weights is computed based on this weighted error, as shown in

$$\beta_t = \frac{\varepsilon_t}{1 - \varepsilon_t}. \tag{9}$$

Typically, β_t will be less than unity. The classification of the image H_i based on the optimal weak classifier $h_t(H_i, x_t, y_t, \theta_t)$ is performed as shown in

$$e_i = \begin{cases} 0, & \text{if } h_t(H_i, x_i, y_i, p_t, \theta_i) = y_i \\ 1; & \text{if } h_t(H_i, x_i, y_i, p_t, \theta_i) \neq y_i. \end{cases} \tag{10}$$

The classification error e_i is used to update the weights of the $(t + 1)$th stage, $W_{t+1,i}$ based on the weights of the tth stage and $w_{t,i}$ according to the Adaboost method of updating the weights, as shown in

$$w_{t+1,i} = w_{t,i} \beta_t^{1-e_i}. \tag{11}$$

As β_t is less than unity, the correctly classified images are weighted lower than the misclassified images. The process of normalizing the weights in the next training stage results in lower weights of the misclassified images than those of the present stage. When all training stages T are complete, the final strong classifier is the weighted majority of the optimal weak classifiers $h_t(H_i, x_t, y_t, p_t, \theta_t)$ of each stage, as shown in

$$H(x) = \begin{cases} 1; & \text{if } \sum_{t=1}^{T} \alpha_t h_t(H_t, x_t, y_t, p_t, \theta_t) \geq \frac{1}{2} \sum_{t=1}^{T} a_t \\ 0; & \text{otherwise,} \end{cases} \tag{12}$$

where $H(x)$ is the strong classifier, $\alpha_t = \log(1/\beta_t)$ is the weight, $h_t(H_t, x_t, y_t, p_t, \theta_t)$ is the weak classifier of tth stage, θ_t is the threshold, and $h_t()$ is the Haar-like feature type.

3. Probabilistic Weighting Adjusted Adaboost

The main objective of our proposed method is to reduce the number of false-positive results. We want to reduce the number of regions in the image that are classified falsely as faces. Therefore, our contribution changes the weighting system of the original Adaboost algorithm based on a probabilistic approach [14].

3.1. Best Weak Classifier Selection. As in the Viola–Jones method, we used training data made of positive (cropped face) and negative (random images without faces) images and used Haar-like features to build the full dictionary of weak classifiers. Note that weight was normalized in step (3.1) of the pseudocode for the Adaboost learning algorithm, which makes the total weighted error sum to 1. As in the Viola–Jones method, a weak classifier is selected as the "best weak classifier" once its total weighted error ε_t is less than 0.5 [4]. In the proposed procedure, a weak classifier was classified as the best one when the weighted positive error was less than 0.05 to keep the positive detection rate at about 95%.

For a given pattern x_i each best weak classifier h_j provides $h_j(x_i) \in \{1, 0\}$, and the final decision of the committee H of selected best weak classifiers is $H(x_i)$, which can be written as the weighted sum of the decision of the best weak classifier as follows:

$$H(x_i) = \alpha_1 h_1(x_1) + \alpha_2 h_2(x_2) + \cdots + \alpha_m h_m(x_i), \tag{13}$$

where $h_1, h_2, \ldots, h_m$ denote m best weak classifiers selected from the pool. The constants $\alpha_1, \alpha_2, \ldots, \alpha_m$ are weights assigned to each classifier decision in the committee. Recall that every h_j just answers "*yes*" (1) or "*no*" (0) to a classification problem, and the result is a linear combination of classifiers followed by a nonlinear decision (sign function).

In the Viola–Jones method, the weight given to each best classifier only depends on the total error that each classifier committed in a training set, as shown in (9). In the original Adaboost method, if two best weak classifiers have the same error, their opinions are given the same weight no matter how different their probabilities of classifying positive or negative images may be. We introduce a new weighting system; the weight given to the opinion ("yes" or "no") of the best weak classifier considers the ability of the best weak classifier to classify positive images on one side as well as negative images. Therefore, this information will be very useful to build a system that reduces the false-positive rate by giving more weight to the best weak classifier with a high probability of classifying the positive images correctly.

3.2. Best Weak Classifier Classification Probability and Weighting. A weak classifier from the pool is voted to be the best weak classifier once once it classifies the input data better than random [4]. Now, let us consider that the output from a given best weak classifier is made for a given number of *well classified* images, *false-positive* (negative images classified as positive) and *false-negative* images (positive images classified as negative). The "*false-positive error probability*" and the "*false-negative error probability*" can be computed as follows [14]:

$$P_{\mathrm{FP}} = \frac{E_{\mathrm{FP}}}{E_{\mathrm{Total}}}, \qquad P_{\mathrm{FN}} = \frac{E_{\mathrm{FN}}}{E_{\mathrm{Total}}}, \tag{14}$$

where E_{Total} is total error, E_{FP} is the false-positive error, E_{FN} is the false-negative error, P_{FP} is the false-positive error probability, and P_{FN} is the false-negative error probability.

After calculating the probabilities we used them to build a new weighting system that considers both error and probabilities for weighting each classifier's opinion. In fact, two classifiers with the same error but with different classification probabilities will have different weights because the probability is considered while assigning weights to the classifiers. The weight in the original method (12) can be rewritten as

$$\alpha = \log\left(\frac{1-\varepsilon_t}{\varepsilon_t}\right), \tag{15}$$

where ε_t is the total error of a considered classifier at stage t. The new alpha called "*probabilistic alpha*" is computed as follows considering the previously calculated probabilities:

$$\alpha' = \left[\log\left(\frac{1-\varepsilon_t}{\varepsilon_t}\right)\right] \times \left(\frac{1-P_{\mathrm{FN}}}{P_{\mathrm{FP}}}\right). \tag{16}$$

The proposed alpha is inversely proportional to the false-positive error probability; thus, when a given classifier has a high false-positive error rate, its weight is lowered; otherwise the inverse is true. Once a best weak classifier produces high P_{FP}, it is given a relatively small weight, which produces a strong classifier that reduces the number of false positives. Note also that when a classifier produces an increasing P_{FN}, the value of the numerator of the term after the multiplication sign in (16) decreases and this also lowers the weight of the proposed alpha. The probabilistic alpha on updating weight allows greater update from misclassified images compared to the original method. This is because the proposed alpha will always be greater than the original alpha according to (16). Hence, misclassified images are relatively highly weighted using the proposed method; therefore, accuracy is higher for the *best weak classifiers*. The following shows the pseudocode for the proposed algorithm.

Pseudocode for the Proposed Algorithm

(1) Images $(x_1, y_1), \ldots, (x_n, y_n)$, where $y = 1, 0$ for face and nonface images, are given.

(2) Initialize weights $w_{1,i} = 1/m, 1/l$, where m and l are the numbers of positive and negative images, respectively.

(3) For $t = 1, \ldots, T$

(3.1) normalize the weights w of the tth stage

$$w_{t,i} = \frac{w_{t,i}}{\sum_{j=1}^{j=n} w_{t,j}}, \tag{17}$$

(3.2) select the best weak classifier $h_t()$ with respect to the weighted error:

$$E_{\mathrm{Total}} = \varepsilon_t = \min \sum_{i=1}^{i=n} w_{t,i} c_i, \tag{18}$$

where

$$c_i = \begin{cases} 0, & \text{if } h_t\left(H_i, x_i, y_i, p_t, \theta_i\right) = y_i \\ 1; & \text{if } h_t\left(H_i, x_i, y_i, p_t, \theta_i\right) \neq y_i, \end{cases} \tag{19}$$

(3.3) compute false-positive error over all indices i of face images, $i = 1, \ldots, P$;

$$E_{\mathrm{FP}} = \sum_{i=1}^{i=P} w_{t,i} c_i, \tag{20}$$

(3.4) compute false-negative error over all indices i of nonface images, $i = 1, \ldots, N$;

$$E_{\mathrm{FN}} = \sum_{i=1}^{i=N} w_{t,i} c_i, \tag{21}$$

(3.5) compute weak classifier classification probabilities;

$$P_{\mathrm{FP}} = \frac{E_{\mathrm{FP}}}{E_{\mathrm{Total}}}, \qquad P_{\mathrm{FN}} = \frac{E_{\mathrm{FN}}}{E_{\mathrm{Total}}}, \tag{22}$$

(3.6) compute alpha α_t';

$$\alpha_t' = \left[\log\left(\frac{1-\varepsilon_t}{\varepsilon_t}\right)\right] \times \left(\frac{1-P_{\mathrm{FN}}}{P_{\mathrm{FP}}}\right), \tag{23}$$

(3.7) update the weights

$$w_i^{(m+1)} = w_i^m e^{(\alpha_t' c_i)}. \tag{24}$$

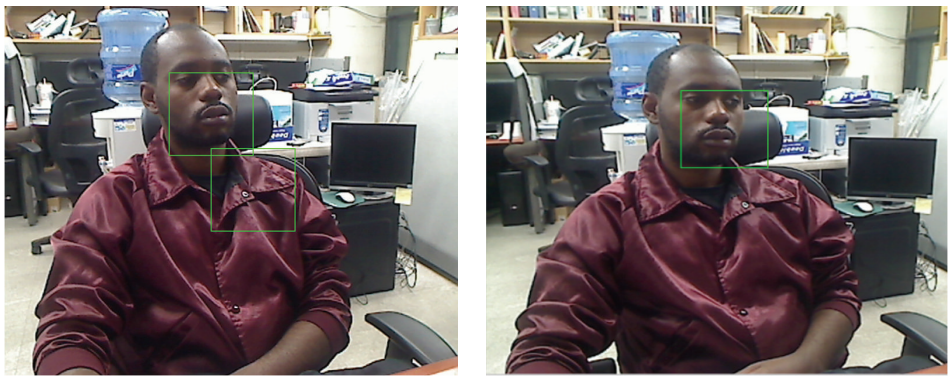
(4) The final strong classifier, $H(x)$, is

$$H(x) = \begin{cases} 1; & \text{if } \sum_{t=1}^{T} \alpha_t' h_t\left(H_t, x_t, y_t, p_t, \theta_t\right) \geq \frac{1}{2} \sum_{t=1}^{T} \alpha_t' \\ 0; & \text{otherwise,} \end{cases} \tag{25}$$

where α_t' is the probabilistic weight and $h_t()$ is the weak classifier of the tth stage.

Table 1: Comparison of the Viola–Jones and the proposed algorithm; the number of false-positive results is almost four times lower using the proposed algorithm.

	True positive (detecting faces)	False positive	False negative (missing faces)	Precision, TP/(TP + FP)
Viola–Jones algorithm	220	495	20	30.8%
Proposed algorithm	216	115	24	65.3%



(a) (b)

Figure 1: Detected faces by the Viola–Jones method (a) and the proposed method (b).

4. Experimental Classification Results and Analysis

We trained and tested data sets, such as the CMU/MIT face data set for training and Georgia Tech face database made of 50 distinct persons (*15 images per subject*) for testing. The Georgia Tech images were taken at different times with different lighting, facial expressions (open/closed eyes, smiling/not smiling), and facial details (glasses/no glasses). All images were taken against a complex background with the subjects in an upright and frontal position (with tolerance for some side movement). We evaluate the performance of the algorithms using the precision. The precision for a class is the number of true positives divided by the sum of true positives and false positives, which are items incorrectly labeled as belonging to the class. High precision means that an algorithm returned substantially more relevant results than irrelevant.

Experimental results obtained from the Viola–Jones and the proposed probabilistic method are described in Table 1. The number of false-positive results is reduced about four times using the proposed method and the precision of the proposed algorithm is twice as high as that of Viola–Jones algorithm without much deteriorating the accuracy. The processing speed is around 30 fps for both methods.

The faces detected by the Viola–Jones method and the proposed method are shown in Figure 1. Figure 1(a) shows a false-positive result and Figure 1(b) shows the same face detected without the false-positive by the proposed probabilistic weighting.

5. Conclusion

The challenge in face detection using Adaboost is the number of false-positive results that accompany actual faces detected on a complex background. In this study, we presented a probabilistic weighting adjusted Adaboost algorithm for detecting faces in a complex background that reduced false-positive errors. We want to reduce the number of regions in an image that are classified as faces but that are not faces. Therefore, we propose a modified version of the Adaboost algorithm that classified weak classifier probabilities for weighting the decisions of each best weak classifier. Classifiers with the same total error have the same weight in the original Adaboost algorithm. However, classifiers with the same error in our proposed probabilistic approach have different probabilities and different weights. In this new weighting system, the classifier's weight is inversely proportional to the false-positive error probability; thus, when a given classifier has a high probability of false-positive errors, its weight is decreased. Experimental results reveal that the proposed algorithm reduces the number of false positives almost four times compared to that of the original Adaboost.

Competing Interests

The authors declare that they have no competing interests.

Acknowledgments

This research was supported by the Basic Science Research Program through the National Research Foundation of

Korea (NRF) funded by the Ministry of Education, Science, and Technology (NRF-2015R1D1A1A01058394) and by the Chung-Ang University Research Grants in 2016.

References

[1] C. Zhipeng, H. Junda, and Z. Wenbin, "Face detection system based on skin color," in *Proceedings of the International Conference on Networking and Digital Society*, May 2010.

[2] S. K. Singh, D. S. Chauhan, M. Vasta, and R. Singh, "A robust color based face detection algorithm," *Tamkang Journal of Science and Engineering*, vol. 6, pp. 227–234, 2003.

[3] R. Rai, D. Katol, and N. Rai, "Survey paper on vehicle theft detection through face recognition system," *International Journal of Emerging Trends & Technology in Computer Science*, vol. 3, no. 1, pp. 256–258, 2014.

[4] P. Viola and M. J. Jones, "Robust real-time face detection," *International Journal of Computer Vision*, vol. 57, no. 2, pp. 137–154, 2004.

[5] C. Niyomugabo and T. Kim, "A probabilistic approach to adjust Adaboost weights," in *Proceedings of the SAI Computing Conference (SAI '16)*, pp. 1357–1360, London, UK, July 2016.

[6] H. Madhuranath, T. R. Babu, and S. V. Subrahmanya, "Modified adaboost method for efficient face detection," in *Proceedings of the 12th International Conference on Hybrid Intelligent Systems (HIS '12)*, IEEE, Pune, India, 2012.

[7] P. Viola and M. J. Jones, "Fast multi-view face detection," Mitsubish Electronic Research Laboratories TR2003-096, 2003.

[8] H. A. Rowley, S. Baluja, and T. Kanade, "Neural network-based face detection," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 20, no. 1, pp. 23–38, 1998.

[9] H. Schneiderman and T. Kanade, "A statistical method for 3D object detection applied to faces and cars," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR '00)*, Bangalore, India, 2000.

[10] O. N. AL-Allaf, "Review of face detection systems based artificial neural networks algorithms," *The International journal of Multimedia & Its Applications*, vol. 6, no. 1, pp. 1–16, 2014.

[11] C. S. Patil and A. J. Patil, "A review paper on facial detection technique using pixel and color segmentation," *International Journal of Computer Applications*, vol. 62, no. 1, pp. 21–24, 2013.

[12] Y. Freund and R. E. Schapire, "A decision-theoretic generalization of on-line learning and an application to boosting," *Journal of Computer and System Sciences*, vol. 55, no. 1, pp. 119–139, 1997.

[13] R. E. Schapire and Y. Singer, "Improved boosting algorithm using confidence-rated prediction," in *Proceeding of the 11th Annual Conference on Computation Learning Theory*, pp. 80–91, Madison, Wis, USA, 1998.

[14] R. V. Hogg and E. A. Tanis, *Probability and Statistical Inference*, Prentice Hall, 2006.