Noisy iris recognition: a comparison of classifiers and feature extractors

Vinícius M. de Almeida
Federal University of Ouro Preto (UFOP)
Department of Computer Science (DECOM)
viniciusmdea@gmail.com

Vinícius A. P. Queiroz
Federal University of Ouro Preto (UFOP)
Department of Computer Science (DECOM)
vini.apq@gmail.com

Abstract

In this work, we present an Iris Recognition method utilizing 2D-Gabor Filter as feature extractor and Adaboost as the classifier. We also compare the results with Artificial Neural Networks and HOG, mixing each combination of both classifiers and feature extractors. The irises are previously normalized by rubbersheet, considering that the irises are already segmented. The normalized iris are then divided in 12 patches of same size. Results are shown in Sec. 5.

1. Introduction

Biometric autentication is a standard security process for many governamental and private organizations. It consists on recognizing a person from its individual phsyical characteristics, such as fingerprints, iris textures or hand palms.

We focus on iris recognition in this paper, since it has a high singularity factor[1] and is imutable with aging. This makes iris a good biometric measure, to be used in a recognition system.

The process of recognition can be divided in two main different process. One is verification and the other is classification. The first one consists in, given two input, images, it returns a value corresponding to the verification if they are from the same person. An example is: a person is trying to enter a place claiming that he is someone. Than, it takes a picture of some biometric measure and the algorithm verifies if he is really that person. The second receives an image to be compared with a database, and then it returns who that person is.

This paper focus on iris verification, where it verifies if the person is really the person that he claims. For this purpose, we compare some methods based on the AdaBoost and Artificial Neural Networks classifiers and the 2D-Gabor Filters and Histogram Oriented Gradients as feature extractors.

This paper is divided in five main sections: In Section 2 we introduce some important concepts needed to fully understand the remaining of this paper. In Section 3 we provide some information related to the Databases used in the experiments in this work. In Section 4 we discuss about how our experiments are performed, and the parameters utilized in all experiments. Then, in Section5 we show our results obtained from all experiments and compare each one of the methods. Finally, in Section 6 we discuss the obtained results and talk about what can be done in future works.

[1]The singularity factor is a measure that gives the probability of having two identical biometry in two same persons in the world

2. Technical Background

This section is meant to provide the technical background on some of the concepts utilized on this work.

2.1. Gabor Filters

Gabor Filters is a band-pass filter which is based on multiple Gaussian distributions, and is used for feature extraction and texture analysis. It represents well the texture contained in an image since it minimizes the uncertainty in both the space and frequency domains. A 2D variant of the Gabor Filters can be rotationed in multiple directions.

Some researches [1] show that the profile of primary-cortex simple cells of mammals resembles the output of Multi-Directional Gabor Filters.

The formula for computing the 2D-Gabor Filter of a given image is as follows:

$$G(x;y;\theta;f) = \exp\left\{-0.5\left[\frac{x'^2}{\delta_{x'}^2} + \frac{y'^2}{\delta_{y'}^2}\right]\right\}\cos(2\pi f x') \tag{1}$$

where $x' = x\cos\theta + y\sin\theta$, $y' = y\cos\theta - x\sin\theta$; f is the frequency of the sinusoidal plane wave along the direction θ, departing from the x-axis; and $\delta_{x'}$ and $\delta_{y'}$ are the standard deviations of the Guassian distribution along the x' and y' axes respectively.

More information about Multi-directional Gabor Filters can be found in [4, 3, 7].

2.2. Adaboost

Adaboost is an adaptive method, which uses the results from weak subsequent classifiers to obtain a high performance. It is based on a weighted sum of the classifications, giving more weight to the weak classifiers that has more accuracy.

In this paper, we treat our feature set as the weak classifiers. Thus, AdaBoost selects the best feature subset from our pool of features. Its output is somewhat similar to applying Principal Component Analysis on our feature set.

More information about AdaBoost can be found in Rojas' paper [6].

2.3. Histogram Oriented Gradients

Histogram Oriented Gradients (HoG) is an algorithm used to extract features from an image, which is invariant to geometric and photometric transformations. It is very flexible, as you can change the geometry of the extracting cell and its size. An example is shown in 1

Figure 1. Example of HoG applied to extract features from an image

2.4. Artificial Neural Networks

Artificial Neural Networks (ANN) is an analogy made from human neurons and how they are connected between each other. It is known that the human brain is actually made of lots of neurons connected in a complex way, and they forward propagate the electric impulses received from multiple "input sensors", and each area of the brain processes these impulses in different ways.

In ANN, to simulate the different areas from the brain, it is assigned a weight to each neuron. We train Artificial Neural Networks to find out which weights are best suited for each classification problem.

To train the ANN, it is first needed to initialize the weights for each neuron. It is very important that these weights are not symmetrical, as this prevents the algorithm

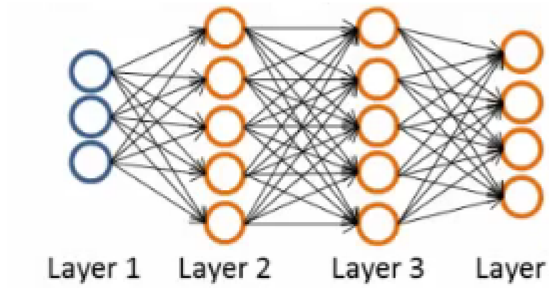

Figure 2. Example of a Artificial Neural Network, with $L = 4$, $K = 4$, $s1 = 3$, $s2 = 5$, etc…

to fail. Thus, instead of initializing all weights as zeros, it is recommended to randomly initialize all weights.

Secondly, it calculates the output hypothesis, passing through the input values $x^{(i)}$, and applying them to the activation functions of each layer ($a_l(x^{(i)})$) multiplying by each neuron weight. This process is called *forward propagation*.

Then, the algorithm evaluates the error caused by each neuron, by evaluating a cost function that compares the output hypothesis and the ground truth. Lastly, it subtracts the error from each weight. This process is called *back propagation*. This algorithm is computed several times, with a goal to minimize the cost function, given as

$$
\begin{aligned}
J(\Theta) = & \frac{1}{m}\sum_{i=1}^{m}\sum_{k=1}^{K}\Big[-y_k^{(i)}\log((h_\Theta(x^{(i)}))_k) \\
& -(1-y_k^{(i)})\log(1-(h_\Theta(x^{(i)}))_k)\Big] \\
& +\frac{\lambda}{2m}\left[\sum_{l=1}^{L-1}\sum_{i=1}^{s_l}\sum_{j=1}^{s_{l+1}}(\Theta_{ji}^{(l)})^2\right]
\end{aligned} \quad (2)
$$

where $\Theta_{ji}^{(l)}$ is the weight matrix of layer l, containing the weights for each connection between neuron j in layer l and neuron i in layer $(l+1)$; m is the number of examples; K is the number of labels, or outputs; L is the number of layers; s_l is the number of neurons (or units) in the layer l; $x^{(i)}$ is the i^{th} example; $y_k^{(i)}$ is the label of the i^{th} example, corresponding to the class k; $h_\Theta(x^{(i)})_k$ can be seen as the probability of an example $x(i)$ belong to class k given a specific weight Θ (if the activation function is a *sigmoid*); and λ is the regularization coefficient.

The rightmost term of Eq. (2) is called the regularization term, parameterized by λ, and used to avoid overfitting on training data (that is, when the classifier labels most training data well, but fails on predictions over the test sets).

An example of an ANN architecture is given in Figure 2.

2.5. ROC curve

Receiver Operating Characteristics (ROC) curves are performance measures well spread around the machine learning community. It is invariant to skewed classes and gives a good knowledge of how the algorithm performs at a first glance. It also turns easily possible to compare multiple methods along the same image, given the ROC curves of each method.

We plot the False Positive Rate and the True Positive Rate along the x-axis and y-axis, respectively. To do so, we vary a threshold of acceptance given a binary classification. This makes possible to determine the best operating points and thresholds for different applications.

To analyze a ROC curve, we try to determine the area under the curve (AUC), or we analyze the distance between the closest point to the upper-left corner of the plot (which corresponds to the ideal classifier, with 100% True Positive Rate and 0% False Positive Rate).

An example of a ROC curve is shown in Fig. 3. The point where the negative indentity line crosses the curve is called Equal Error Rate point, and is also a performance measure that can be analyzed in ROC curves. More information about ROC curve analysis can be found in [2].

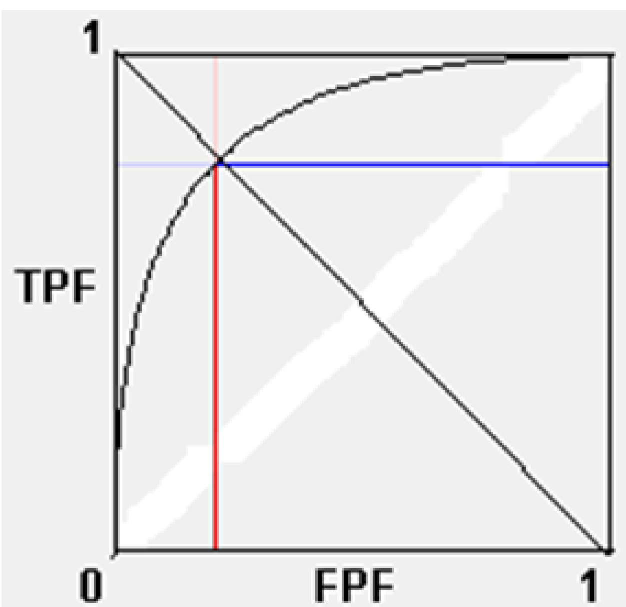

Figure 3. Example of a ROC curve

3. The Benchmark

All experiments were performed in a part of the UBIRIS.v2 iris dataset[2] . It consists in more than 11000 iris images captured at-a-distance and on-the-go, with noises such as occlusion by hair, eyelids or glass frames, uneven lightning conditions, and others.

More information about the dataset can be found in [5].

4. System Architecture

In 2009, a competition in iris recognition, called NICE:II was realized, with 67 competitors. The Techshino Biometrics Research Center took second place in this competition, with an algorithm that was based on AdaBoost and Multi-Directional Gabor Filters [7]. We base our paper on their work.

But the competition organizers provided, along with some images of UBIRIS.v2, a binary mask for each iris image, which would give the ground truth for iris location, and would exclude noises such as eyelids, reflections and the pupil. As we do not have these masks, we simply undervalue these noises and perform feature extraction regardless of these noises.

We also feed our algorithm with already-segmented and normalized iris images. These irises were segmented by Daugman's differntial-operator and normalized by rubber-sheet simplification. More information of both can be found in [7].

Nevertheless, we propose this sequence of processes to perform a somewhat accurate iris recognition process:

1. Select 480 intra-class and 480 inter-class matches from the database;
2. Divide selected matches in equally sized training, validation and test set;
3. Select and apply feature extractor (Gabor filters or HOG);
4. Calculate the dissimilarity between the images;
5. Select and train classifier (AdaBoost or ANN);
6. Validate results and change parameters, if necessary;
7. Report accuracy and plot ROC curve of predictions on test set.

4.1. Feature extraction and Dissimilarity calculation

For the purpose of constructing a feature extractor, we utilized both HOG and 2D-Gabor Filters and compared their results. To do so, we first divided both input irises images in twelve patches, according to the scheme adopted by [7]. Then, we applied the chosen feature extractor in both whole images and their patches.

Next, we computed the dissimilarity of each corresponding subimage. To do so, for the HOG case, we simply took

[2] Available at http://iris.di.ubi.pt/ubiris2.html

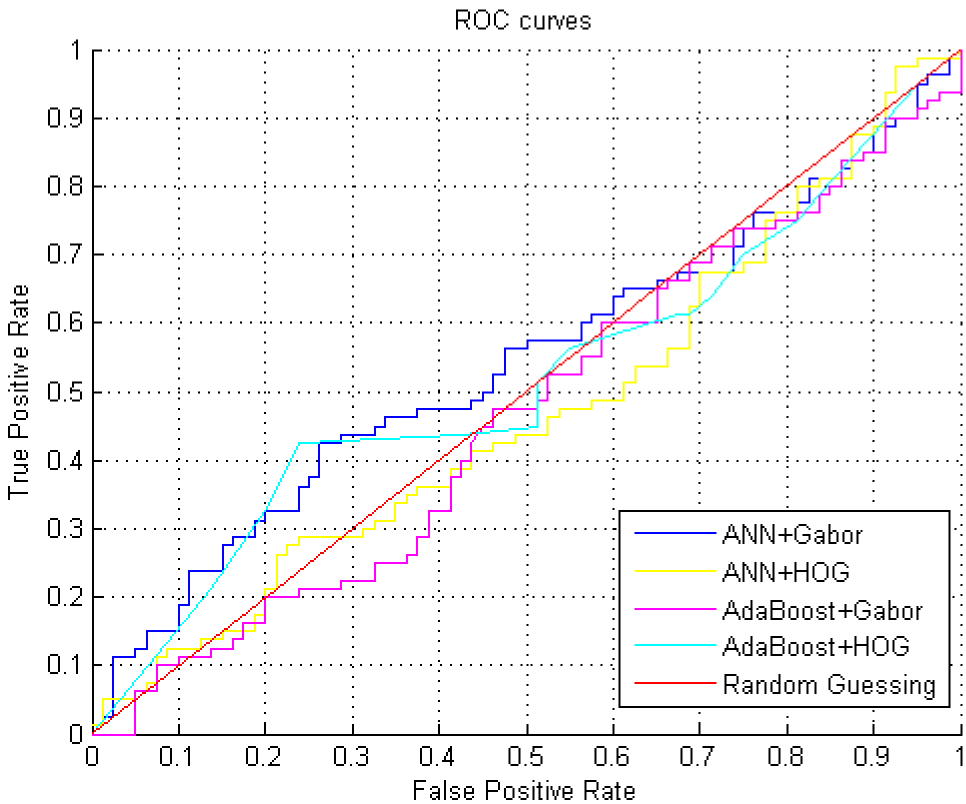

Figure 4. Results from experiments with the four different combinations of methods.

the sum of absolute differences between the intensities resulted from the HOG function. For the Gabor case, we computed the dissimilarity by appling a XOR function element-wise between each pair of corresponding subimages.

After computing the dissimilarities of these subimages, we get a 1×13 length feature vector (12 patches plus the whole irises) for each example, which consists of a pair of images and its label. The label is either $+1$ if the matches are inter-class or -1 if the matches are intra-class.

4.2. Classifier

After constructing our learning sets with the method described above for feature extraction, we train and validate our system with one of the two selected classifiers: AdaBoost or Artificial Networks.

For the AdaBoost case, we choose a number of iterations and train the algorithm. As for the ANN case, we have to choose the number of maximum iterations (or epochs), the number of hidden layer neurons (or units, s_2) and the value of the regularization parameter (λ).

5. Experiments and Results

We have four combinations of methods, and the results obtained from each of them are shown in this section.

We achieved a 58.25% accuracy in classification in our test set, with AdaBoost with 33 iterations and Gabor Filters. AdaBoost with 5 iterations and HOG achieves better performance, with 64.38% accuracy in the test set. It was noticed that after a certain number of iterations - one for each case - AdaBoost started to overfit the data, thus, failing to generalize, and obtaining poorer results in the validation set.

ANN with $s_2 = 1000$, $\lambda = 0.1$ and 13 maximum iterations and HOG achieves the best performance, with 65.00%. And ANN with $s_2 = 200$, $\lambda = 0.03$ and 100 maximum iterations with Gabor performs worse than random guessing, with only 45.00% accuracy. The ROC curves are shown and compared in Fig. 4.

It is worth mentioning that we can obtain a good classifier by reversing the polarity of the classifier, if it performs worse than random guessing. That is, if the classifier says the example belongs to one class, we say that it belongs to the other one. Then, the results are actually inverted, and a better accuracy is obtained.

The scores for plotting the ROC curves are the sigmoid outputs from the output layer, when classifying with ANN's, and the weighted sum of the classifiers for each example, when classifying with AdaBoost.

6. Conclusion

From the ROC curves and the accuracies measured, we notice that HOG features performs better than Gabor Filters in both cases, and that both ANN and AdaBoost have similar performances. There is a lot of improvements that still can be made in this system, such as a noise removing preprocessing, a more robust segmentation, and also, it should be tested with the combination of more than one direction of the 2D-Gabor Filter. Another good result could be obtained by combining both HOG and Gabor features.

7. Acknowledments

We would like to thank Ph.D. David Menotti for all the knowledge provided, so that we could accomplish this task.

References

[1] B. W. Andrews and D. A. Pollen. Relationship between spatial frequency selectivity and receptive field profile of simple cells. *The Journal of physiology*, 287:163–176, 1979.

[2] T. Fawcett. An introduction to ROC analysis. *Pattern Recognition Letters*, 27:861–874, 2006.

[3] L. Ma and Y. Wang. Iris recognition based on multichannel Gabor filtering. *Proc. Fifth Asian Conf. Computer*, pages 1–5, 2002.

[4] J. R. Movellan. Tutorial on Gabor Filters. *Response*, 49:1–23, 2002.

[5] H. Proenca, S. Filipe, R. Santos, J. Oliveira, and L. Alexandre. The UBIRIS.v2: A database of visible wavelength images captured on-the-move and at-a-distance. *IEEE Trans. PAMI*, 32(8):1529–1535, August 2010.

[6] R. Rojas. AdaBoost and the Super Bowl of Classifiers A Tutorial Introduction to Adaptive Boosting. Technical report, 2009.

[7] Q. Wang, X. Zhang, M. Li, X. Dong, Q. Zhou, and Y. Yin. Adaboost and multi-orientation 2D Gabor-based noisy iris recognition. *Pattern Recognition Letters*, 33:978–983, 2012.