Eye Localization Using Convolutional Neural Networks and Image Gradients

Werton P. de Araujo[1], **Thelmo P. de Araujo**[1], **Gustavo A. L. de Campos**[1]

[1]Universidade Estadual do Ceará (UECE)
Av. Dr. Silas Munguba, 1700 – 60714-903 – Fortaleza – CE – Brazil

araujo.werton@gmail.com, thelmo.araujo@uece.br, gustavo@larces.uece.br

Abstract. *Eye detection is a preprocessing step in many methods using facial images. Some algorithms to detect eyes are based on the characteristics of the gradient flow in the iris-sclera boundary. These algorithms are usually applied to the whole face and a posterior heuristic is used to remove false positives. In this paper, we reverse that approach by using a Convolutional Neural Network (CNN) to solve a regression problem and give a coarse estimate of the eye regions, and only then do we apply the gradient-based algorithms. The CNN was combined with two gradient-based algorithms and the results were evaluated regarding their accuracy and processing time, showing the applicability of both methods for eye localization.*

1. Introduction

Eye detection and localization on facial images is an important preprocessing step in many image processing methods. Matching the center of the eyes improves face recognition algorithms [Dutta et al. 2015]. Eye gaze methods depend on identifying the precise location of the pupils [Kar and Corcoran 2017]. In order to ensure a good performance, many shape models need to position the initial shape very close to the correct location: a fast, low cost method for eye localization may provide a useful first guess [Abdulameer et al. 2014]. Applications of an accurate algorithm for eye localization abound in the literature.

Kothari and Mitchell [Kothari and Mitchell 1996] proposed a method to locate eye centers based on the characteristics of the gradient field on the iris-sclera region (see Section 3). The method performs well when applied to the region of interest around the eye. To remove the false positives, the authors suggest two heuristics using *a priori* knowledge of human face.

Timm and Barth [Timm and Barth 2011] also used the gradient field characteristics of the human eye images to pinpoint the eye centers in facial images. In order to apply the proposed method to the regions of interest (eyes), the Viola-Jones algorithm [Viola and Jones 2001] was used to detect the face and two rectangular regions were computed, based on the anthropometric relations of the face. The method was then applied to these regions. Although fast and accurate, the Viola-Jones algorithm for face detection has some known setbacks, e.g., heads with tilting angles greater than 5° are usually not detected [Mushfieldt et al. 2013].

In this work, we propose to combine a Convolutional Neural Network (CNN) with each of these gradient-based methods to locate the eyes on facial images. Both methods

are evaluated in order to determine their accuracy and their processing times. Because the methods are primarily intended to be used in preprocessing, the CNN was designed to have few layers (see Section 2).

This paper is organized as follows: Section 2 describes the main features of CNNs. On Section 3, the methods proposed in [Kothari and Mitchell 1996] and [Timm and Barth 2011] are summarized. Our proposal and the experiments are presented in Section 4. Results are shown and discussed in Section 5. Section 6 concludes the paper, with indications for some future work.

2. CNNs Overview

Convolutional Neural Network (CNN) architectures are usually described by three types of layers [Patterson and Gibson 2017]: input layer, feature extraction layers, and classification layers. Unlike neural networks such as the Multilayer Perceptron (MLP), CNN layers are better visualized as tridimensional volumes.

The input layer is composed by each pixel of an input image and has dimensions $W \times H \times D$, where W is the width, H is the height, and D is the depth (number of channels) of the image.

The feature extraction group usually consists of a combination of convolutional layers, ReLU layers, and pooling layers.

In the convolutional layer, a set of K filters of size $F \times F$ is applied to the input layer in order to extract relevant features. Besides K and F, other important parameters of this layer are the stride S and the zero-padding P.

In order to add non-linearity to the feature extraction, a ReLU (Rectified Linear Units) activation function is applied to the filtered images. Due to its fastness, $\max(0, x)$ is a common choice.

A pooling layer is used to reduce the dimensionality increased by the convolution of K filters in the feature extraction process. The most common pooling function computes the maximum of an $F \times F$ region of each plane on the feature volume, with stride S.

The classification group is usually composed by one or more fully connected (FC) layers that perform the affine combination that feeds the output layer. A dropout regularization method (i.e., temporarily dropping the active neurons in the FC layer with probability p at each input) may be applied to reduce over-fitting.

Although CNNs have been mostly used for image classification [Patterson and Gibson 2017], they may also be used to solve regression problems such as the localization of license plates [Ozhiganov 2016] and facial key points [Nouri 2014].

In this work, we designed a CNN to estimate the coordinates of the centers of both eyes.

3. Gradient-Based Algorithms

The circular shape of the human iris and the color contrast it has with the surrounding sclera inspired the use of image gradients to pinpoint the center of the eye in both

monochromatic and color images. As shown in Figure 1, gradients in the iris-sclera border have greater magnitude than the ones in those relatively homogeneous regions. These border gradients point outwards and their support lines (the line that passes at the given point and has the same direction of the vector) tend to meet at the center of the eye.

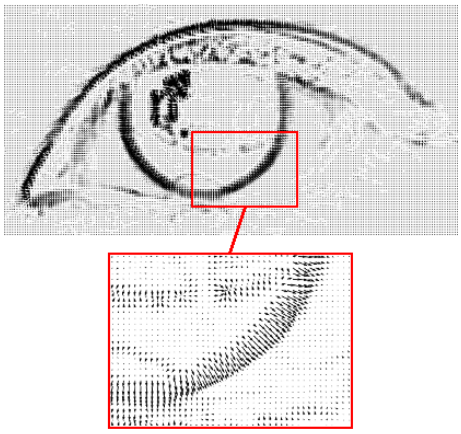

Figure 1. Gradients on the iris-sclera region.

Using this prior information, Kothari and Mitchell [Kothari and Mitchell 1996] proposed an algorithm (Algorithm 1) to determine the location of the eye centers in a gray scale image. The image is divided into a 2-dimensional array of bins and each bin has a counter (initially set to zero) associated with it. For each gradient vector with magnitude greater than a prescribed threshold, a line segment is formed by extending the gradient vector in its opposite direction; the counter is incremented every time a line segment crosses the corresponding bin (see Figure 2). The eye center candidates are in the bins with greater counts. The bounding threshold for the gradient magnitudes and the bin size must be carefully chosen in order to speed up the computations and reduce complexity.

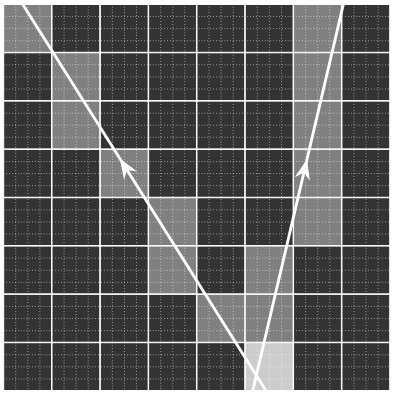

Figure 2. Accumulate bins for two gradient vectors.

Also inspired by the characteristics of the iris-sclera gradients, Timm and Barth [Timm and Barth 2011] noticed that the sum of the absolute values of inner products of the normalized gradient vectors $\mathbf{g}_i$ and the displacement vectors

$$\mathbf{d}_i = \frac{\mathbf{x}_i - \mathbf{c}}{\|\mathbf{x}_i - \mathbf{c}\|} \tag{1}$$

Algorithm 1: Kothari & Mitchell's algorithm.

```
input  : Region of interest (ROI)
output: Eye center position
begin
    compute |g_i| in the ROI;
    compute the magnitude threshold T_g;
    initialize the matrix of bins with zeroes;
    foreach point i in the ROI do
        if |g_i| ≥ T_g then
            foreach point on the line along the direction opposite the gradient
              do
                add 1 to the value of the corresponding bin;
            end
        end
    end
    return the position of the bin with the greatest accumulated value;
end
```

reaches its maximum when $\mathbf{c}$ is the center of the iris (see Figure 3). That is, the center of the iris is given by

$$\mathbf{c}^* = \arg\max_{\mathbf{c}} \left\{ \frac{1}{N} \sum_{i=1}^{N} (\mathbf{d}_i^T \mathbf{g}_i)^2 \right\} . \tag{2}$$

Algorithm 2 implements this idea.

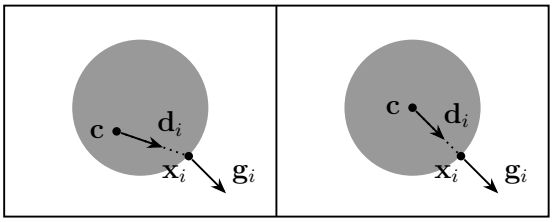

Figure 3. Displacement and gradient vectors.

4. Proposal and Experiments

4.1. Proposal

As already noted, CNNs are usually used for image classification, but they may also be applied to solve regression problems. In this work, we propose to locate the center of the eyes in frontal facial images by combining a Convolutional Neural Network and a gradient-based algorithm. The CNN shall wield a coarse estimate of the eye region and the gradient-based algorithm shall refine the location of the center of the eyes.

We implemented a simplified CNN inspired by the architecture proposed by Daniel Nouri [Nouri 2014]. The rationale for the architectural simplification is the intention to build a system easy and fast to train using specific small databases. Table 1

Algorithm 2: Timm & Barth's algorithm.

input : Region of interest (ROI)
output: Eye center position
begin
  compute $|\mathbf{g}_i|$ in the ROI;
  compute the magnitude threshold T_g;
  initialize the matrix of sums with zeroes;
  foreach *point i in the ROI* **do**
    if $|\mathbf{g}_i| \geq T_g$ **then**
      foreach *point i in the ROI* **do**
        compute $\mathbf{d}_i$ using Eqn. 1;
        compute $\mathbf{d}_i^T \mathbf{g}_i$;
        if $\mathbf{d}_i^T \mathbf{g}_i > 0$ **then**
          add $(\mathbf{d}_i^T \mathbf{g}_i)^2$ to the corresponding matrix element;
        end
      end
    end
  end
  return the position of the matrix element with the greatest accumulated value;
end

shows the number of parameters for each layer connection, adding up to $4\,234\,624$ parameters, which is not an impressive figure when one considers most deep CNNs found in literature.

Table 1. Number of parameters of the CNN.

Connection	Number of Parameters
INPUT $\rightarrow$ CONV1	$4 \times 3 \times 3 + 4$
POOL1 $\rightarrow$ CONV2	$16 \times 2 \times 2 + 16$
POOL2 $\rightarrow$ FC3	$(16 \times 23 \times 23) \times 500 + 500$
FC3 $\rightarrow$ FC4	$500 \times 4 + 4$
Total	$4\,234\,624$

Once the CNN computes the initial location estimate, i.e., right ($\widehat{C}_r$) and left ($\widehat{C}_l$) eye coordinates, two horizontally aligned rectangles are drawn with centroids in $\widehat{C}_r$ and $\widehat{C}_l$, height $= 0.7d$, and width $= 0.9d$, where d is the Euclidean distance between the estimated eye centers (see Figure 4).

These rectangles are the images to be processed by the gradient-based algorithms in order to refine the localization of the eye centers.

4.2. Database

The BioID database has 1521 gray scale 384×286 images of 23 persons, labeled with the coordinates of the center of each eye [BioID 2017].

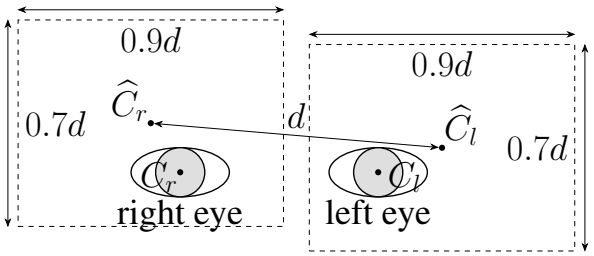

Figure 4. Regions of interest.

In the CNN experiments, all images were scaled down to half the original size and cropped, reducing their dimensions to 96×96. 32 images were discarded because, in the cropping process, one or both eyes were no longer in the image. The resulting set of images was split into training and testing sets with 1420 and 69 images respectively. We duplicated the training set by horizontally flipping the images, resulting in a training set with 2840 images.

4.3. Implementation

The implemented CNN architecture and hyperparameters are show in Figure 5 and Table 2, respectively. Stride, for the convolutional and pooling layers, was set to 1, and zero-padding was set to zero.

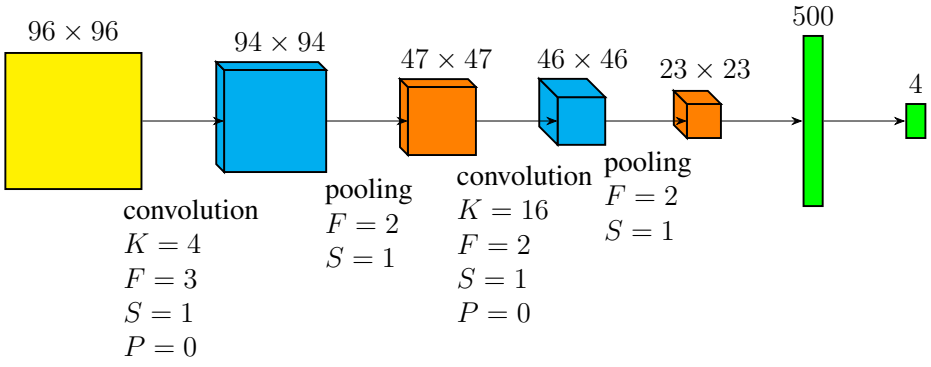

Figure 5. Implemented CNN architecture.

Table 2. CNN's hyperparameters and layer dimensions.

Layer	Number of Filters	Filter Size	Dimensions
INPUT	-	-	$1 \times 96 \times 96$
CONV1	4	3	$4 \times 94 \times 94$
POOL1	-	2	$4 \times 47 \times 47$
CONV2	16	2	$16 \times 46 \times 46$
POOL2	-	2	$16 \times 23 \times 23$
FC3	-	-	500
FC4	-	-	4

A 50% dropout was included in the first fully connected layer (FC3). The other relevant settings were the weight initialization strategy (Glorot Uniform), the ReLU activation function, the error function (mean square error), and the optimization method (Nesterov's momentum of 0.9 with learning rate of 0.01).

The output layer (FC4) has 4 neurons, which correspond to the x and y coordinates for the left eye, and the x and y coordinates for the right eye, in that order (x_l, y_l, x_r, y_r).

A 10-fold cross validation scheme was adopted to train and validate the proposed CNN architecture using the training set with 2840 images. For 2000 epochs, we obtained a mean training error of 2.30 pixels (with standard deviation of 0.062) and a mean validation error of 7.22 pixels (with standard deviation of 2.057). The decreasing behavior of the validation error showed that no over-fitting occurred.

After validating the CNN, the full validation set, with 2840 images, was used to train the network and the obtained weights were saved for the test phase.

Both Algorithms 1 and 2 were then applied to the regions of interest obtained by the CNN regression, but this time using the *original* 384×286 images.

We followed Kothari and Mitchell's suggestion [Kothari and Mitchell 1996] for the two parameters on Algorithm 1: 5×5 for the accumulation bin size; and the root mean square as the magnitude threshold.

In order to bound the magnitude of the gradients to be considered in Algorithm 2, we followed Hume [Hume 2012] and set the threshold to $(0.3\,\sigma + \mu)$, the mean and standard deviation of the gradient magnitudes in the corresponding region of interest.

5. Results

We followed Jesorsky et al. [Jesorsky et al. 2001] and use the *normalized error* to evaluate the results obtained by the CNN regression and the posterior application of Algorithms 1 and 2. The normalized error and its variants are given by:

$$e = \frac{1}{d} \max(e_r, e_l)\,, \tag{3}$$

$$e_{avg} = \frac{1}{2d}(e_r + e_l)\,, \text{ and} \tag{4}$$

$$e_{better} = \frac{1}{d} \min(e_r, e_l)\,, \tag{5}$$

with $d = \|C_r - C_l\|$, $e_r = \|C_r - \widehat{C}_r\|$, and $e_l = \|C_l - \widehat{C}_l\|$, where C_r and C_l are the true right and left eye coordinates, $\widehat{C}_r$ and $\widehat{C}_l$ are the corresponding estimated values, and $\|.\|$ is the Euclidean distance.

Figure 6 shows the normalized error ranges for an average human eye [Farkas et al. 1985]. Results for localization of eye centers may be considered very good if $e \leq 0.05$; as for eye localization, results may be considered good if $e \leq 0.25$.

The results were evaluated in terms of accuracy $(\%)$ for each of the five error ranges: $e \leq 0.05$, $e \leq 0.10$, $e \leq 0.15$, $e \leq 0.20$, and $e \leq 0.25$.

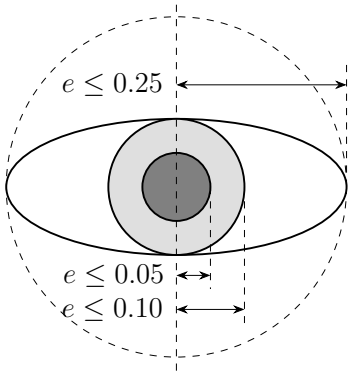

Figure 6. Normalized error ranges for an average human eye.

Table 3 shows the results of the CNN's estimation of eye centers. The results for $e \leq 0.05$ in Table 3 suggest that the CNN is not good enough for accurately locating the center of the eyes. However, the results for the normalized error range $e \leq 0.25$ indicate that the relatively simple CNN used here is suitable to detect a region of interest where the eyes are to be further processed by either Algorithm 1 or Algorithm 2.

Table 3. Accuracy ($\%$) of CNN eye localization vs. normalized error range.

Error Range	Worst eye (Eqn. 3)	Average (Eqn. 5)	Best eye (Eqn. 4)
$e \leq 0.05$	13.0%	20.3%	33.3%
$e \leq 0.10$	49.3%	63.8%	79.7%
$e \leq 0.15$	76.8%	84.1%	87.0%
$e \leq 0.20$	89.9%	89.9%	91.3%
$e \leq 0.25$	92.8%	94.2%	95.7%

Table 4 shows the results obtained by Algorithm 1 for a finer eye center localization. Unfortunately, the results were not accurate enough for pupil localization ($e \leq 0.05$). However, the algorithm performs well for iris detection ($e \leq 0.10$).

Table 4. Accuracy ($\%$) of Algorithm 1 eye localization vs. normalized error range.

Error Range	Worst eye (Eqn. 3)	Average (Eqn. 5)	Best eye (Eqn. 4)
$e \leq 0.05$	26.1%	50.7%	73.9%
$e \leq 0.10$	72.5%	85.5%	92.8%
$e \leq 0.15$	92.8%	94.2%	98.6%
$e \leq 0.20$	95.7%	95.7%	100%
$e \leq 0.25$	95.7%	95.7%	100%

The results obtained with Algorithm 2, as implemented here following [Timm and Barth 2011] and [Hume 2012], show good accuracies for pupil detection, performing very well for iris localization ($e \leq 0.10$). All results were better than the ones obtained with Algorithm 1.

Table 5. Accuracy (%) of Algorithm 2 eye localization vs. normalized error range.

Error Range	Worst eye (Eqn. 3)	Average (Eqn. 5)	Best eye (Eqn. 4)
$e \leq 0.05$	72.5%	85.5%	97.1%
$e \leq 0.10$	91.3%	92.8%	98.6%
$e \leq 0.15$	91.3%	98.6%	98.6%
$e \leq 0.20$	94.2%	98.6%	98.6%
$e \leq 0.25$	98.6%	98.6%	98.6%

Although Algorithm 2 is much more accurate than Algorithm 1, as seen in Tables 4 and 5, the latter is faster than the former and may be very useful as a preprocessing step, giving a reasonably accurate eye localization. Algorithm 1 took, in average, 8.56 ms to locate both eye centers, 11.6 times faster than Algorithm 2 (99.42 ms, in average).

All the experiments were performed on a computer equipped with a 3 GHz Intel Core 2 Duo CPU, 4 GB RAM, and an Nvidia GeForce GTS 450 GPU with 1 GB GDDR5.

6. Conclusion and Future Work

In this paper, we applied a relatively small CNN to solve the regression problem of eye localization in grayscale frontal facial images. The CNN coarse estimate was then used to compute regions of interest (eyes) that serve as input to two gradient-based algorithms intended to pinpoint the eye centers. The original rationale of both algorithms (applying the gradient method and then removing the false positives) was reverted here: The CNN estimates the approximated eye positions and *then* the gradient methods are applied.

Results were evaluated in terms of accuracy and processing time: CNN combined with Algorithm 2 proved to be more accurate the CNN-Algorithm 1 combination and may be used to detect irises. The combination of CNN and Algorithm 1 may still be used to locate eye regions, specially when speed is of relevance, since it proved to be much faster than the other method.

We are currently working on some adaptations of both algorithms and on a method that merges them.

Acknowledgment

WPA acknowledges the financial support from FUNCAP.

References

Abdulameer, M., sheikh abdullah, S., and Othman, Z. (2014). A modified active appearance model based on an adaptive artificial bee colony. 2014:879031.

BioID (2017). The BioID Face Database. https://www.bioid.com/facedb/.

Dutta, A., Günther, M., Shafey, L. E., Marcel, S., Veldhuis, R., and Spreeuwers, L. (2015). Impact of eye detection error on face recognition performance. *IET Biometrics*, 4(3):137–150.

Farkas, L. G., Hreczko, T., Kolar, J. C., and Munro, I. R. (1985). Vertical and horizontal proportions of the face in young adult North American Caucasians: revision of neoclassical canons. *Plastic and Reconstructive Surgery*, 75(3):328–337.

Hume, T. (2012). Simple, accurate eye center tracking in OpenCV. http://thume.ca/projects/2012/11/04/simple-accurate-eye-center-tracking-in-opencv/.

Jesorsky, O., Kirchberg, K. J., and Frischholz, R. (2001). Robust face detection using the Hausdorff distance. In Bigün, J. and Smeraldi, F., editors, *Audio- and Video-Based Biometric Person Authentication, Third International Conference, AVBPA 2001 Halmstad, Sweden, June 6-8, 2001, Proceedings*, volume 2091 of *Lecture Notes in Computer Science*, pages 90–95. Springer.

Kar, A. and Corcoran, P. (2017). A review and analysis of eye-gaze estimation systems, algorithms and performance evaluation methods in consumer platforms. *IEEE Access*, 5:16495–16519.

Kothari, R. and Mitchell, J. L. (1996). Detection of eye locations in unconstrained visual images. In *Proceedings 1996 International Conference on Image Processing, Lausanne, Switzerland, September 16-19, 1996*, pages 519–522. IEEE Computer Society.

Mushfieldt, D., Ghaziasgar, M., and Connan, J. (2013). Robust facial expression recognition in the presence of rotation and partial occlusion. In *Proceedings of the South African Institute for Computer Scientists and Information Technologists Conference*, SAICSIT '13, pages 186–193, New York, NY, USA. ACM.

Nouri, D. (2014). Using convolutional neural nets to detect facial keypoints tutorial. http://danielnouri.org/notes/2014/12/17/using-convolutional-neural-nets-to-detect-\\facial-keypoints-tutorial/.

Ozhiganov, I. (2016). Convolutional Neural Networks for Object Detection. http://rnd.azoft.com/convolutional-neural-networks-object-detection/.

Patterson, J. and Gibson, A. (2017). *Deep Learning: A Practitioner's Approach*. O'Reilly, Sebastopol, CA.

Timm, F. and Barth, E. (2011). Accurate eye centre localisation by means of gradients. In Mestetskiy, L. and Braz, J., editors, *VISAPP 2011 - Proceedings of the Sixth International Conference on Computer Vision Theory and Applications, Vilamoura, Algarve, Portugal, 5-7 March, 2011*, pages 125–130. SciTePress.

Viola, P. and Jones, M. (2001). Rapid object detection using a boosted cascade of simple features. pages 511–518.