Meibomian Glands Recognition

Image and Video Analysis

Luigi Ariano
luigi.ariano@stud.unifi.it

Edoardo Bonanni
edoardo.bonanni@stud.unifi.it

Master's Degree in Computer Engineering

Università degli Studi di Firenze

2020-2021

Abstract

In this report, we tackle the problem of meibomian glands recognition by using the extracted features from Gabor filter for a Support Vector Machine (**SVM**) classifier. The Gabor filter is necessary to get basic information of that tiny glands due to their particular extension which follows the shape of the tarsal plate, that is different in both lids.

The SVM allows us to recognize the area of meibomian glands in every scanned image by differentiating initially upper and lower lids and ignoring the part not affected.

1 Introduction

The tarsal glands of Meibom (glandulae tarsales) are large sebaceous glands located in the eyelids. Lipids produced by the meibomian glands are the main component of the superficial lipid layer of the tear film that protects it against evaporation of the aqueous phase and is believed also to stabilize the tear film by lowering surface tension. Hence, meibomian lipids are essential for the maintenance of ocular surface health and integrity.

Meibomian gland dysfunction (**MGD**) is a common eye condition, it can get when there's a problem with a few dozen tiny glands in your eyelids that help make the oil layer of your tears. The most common MGD happens when the gland openings get clogged, and less and less oil reaches the eye surface.

Our task, basically, is to identify the regions of the glands from an image so that they can be analyzed for medical use. An example of Meibomian gland is shown in figure 1.

It was provided to us 392 images of meibomian gland, divided into:

- 178 images of the lower gland;
- 214 images of the upper gland.

This division is due to the particular extension of glands which follows the shape of the tarsal plate, that is different in both lids. In this way, we can train two different SVM classifier and consider the particular features of each set of glands to improve the accuracy to recognize them.

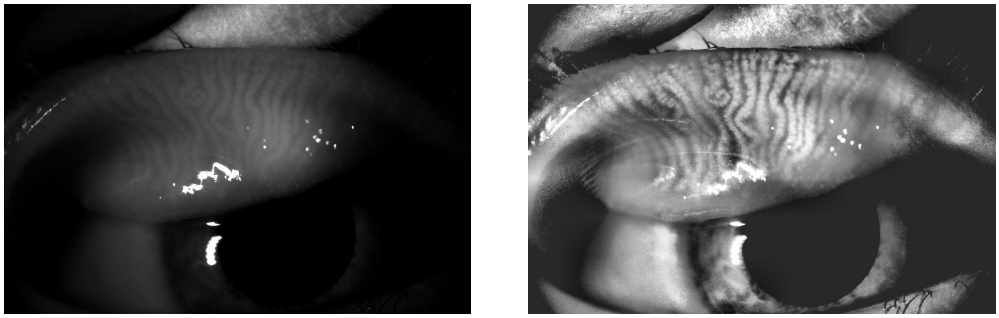

Figure 1: On the left side, there is an upper meibomian gland image and on the right the corresponding figure with CLAHE applied.

2 Preprocessing Operations

2.1 CLAHE

The Histogram equalization usually increases the global contrast of many images, especially when the image is represented by a narrow range of intensity values. Through this adjustment, the intensities can be better distributed on the histogram utilizing the full range of intensities evenly. This allows for areas of lower local contrast to gain a higher contrast. Histogram equalization accomplishes this by effectively spreading out the highly populated intensity values which use to degrade image contrast.

Adaptive Histogram Equalization (**AHE**) is different from normal histogram equalization because AHE uses several methods each corresponding to different parts of image and used them to redistribute the lightness value of it. Ordinary AHE tends to overamplify the contrast in near-constant regions of the image, since the histogram in such regions is highly concentrated. As a result, AHE may cause noise to be amplified in near-constant regions.

Contrast Limited AHE (**CLAHE**) is a variant of adaptive histogram equalization in which the contrast amplification is limited, so as to reduce this problem of noise amplification. An example of CLAHE application is shown in figure 1.

2.2 Mask creation

After the previous step, where we applied the CLAHE algorithm on every image of the dataset, we divide each image in superpixels through the Simple

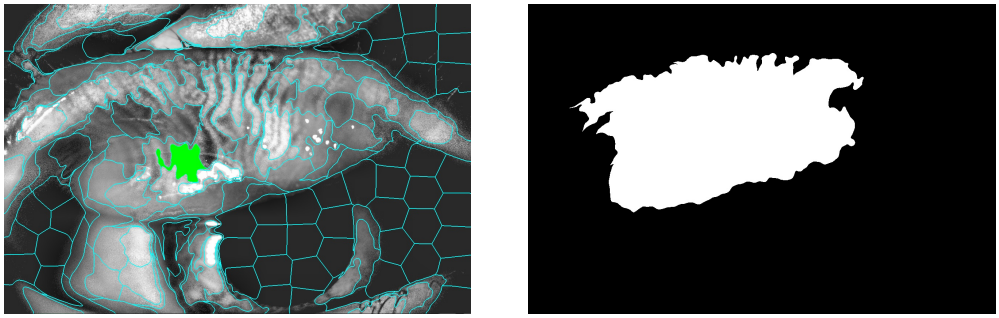

Figure 2: On the left side, there is the CLAHE image which is segmented with **slic** function and it can be selected by hand. On the right side, there is the mask generated by selecting the corresponding gland's region.

Linear Iterative Clustering (**slic**)[1] function into 180 segments. With this, we can select by hand every area which corresponds to gland's region, color this region (green) and create a mask with the same size of the original image. Every mask has a white area relative to the gland region and, therefore, the region that we selected and colored before.

In conclusion, we associate a label **255** to that area and **0** to the remaining part of the image, that is coloured in black.

3 Gabor filter

The further step have been to applied the Gabor filter: it is orientation-sensitive filters, used for edge and texture analysis. It can be viewed as a sinusoidal plane of particular frequency and orientation, modulated by a Gaussian envelope.

A Gabor filter oriented in a particular direction gives a strong response for locations of the target images that have structures in this given direction. For example, if the target image is made up of edges in the diagonal direction, a Gabor filter will give a strong response only if its direction matches with the direction of the edges. In our case, we want a higher response from the glands (i.e. figure 3) and, given that the shape is vertical in most cases, we don't use too much orientation otherwise it could give a strong response even for lashes or other elements in the image.

[1] scikit-image library

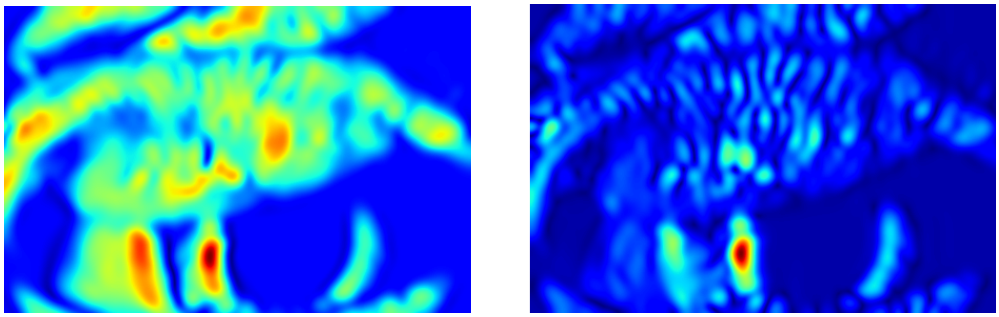

Figure 3: There are two images that represents some features of Gabor filter applied to the figure 1.

The impulse response is defined by a sinusoidal wave (a plane wave for 2D Gabor filters) multiplied by a Gaussian function. Because of the multiplication-convolution property, the Fourier transform of a Gabor filter's impulse response is the convolution of the Fourier transform of the harmonic function (sinusoidal) and that one of the Gaussian function. The filter has a real and an imaginary component representing orthogonal directions and the components may be formed into a complex number or used individually:

$$g(x, y; \lambda, \theta, \psi, \sigma, \gamma) = \exp(-\frac{X^2 + \gamma^2 Y^2}{2\sigma^2}) \cos(2\pi \frac{X}{\lambda} + \psi), \tag{1}$$

$$X = x \cos\theta + y \sin\theta, \; Y = -x \sin\theta + y \cos\theta \; . \tag{2}$$

In this equation, λ represents the wavelength of the sinusoidal factor, θ represents the orientation of the normal to the parallel stripes of a Gabor function, ψ is the phase offset, σ is the sigma/standard deviation of the Gaussian envelope and γ is the spatial aspect ratio and specifies the ellipticity of the support of the Gabor function.

In conclusion, we obtain four images from the application of that function on each image of the dataset that represents the features derived. For us, the parameters that give a better accuracy are:

- gabor_kernel_size = 129;
- $\lambda = 1.01, 1.015$;

- $\theta = 0, \pi/2$;
- $\psi = 0, \pi/2$;
- $\sigma =$ gabor_kernel_size/5;
- $\gamma = 1.5$.

4 Dataset

In this section, we described the dataset created to reach our goal. The first step has been to partition the four features images obtained from the previous step, for each figure provided, into a 10x10 cells grid. Every feature matrix is multiplied by its transpose to obtain covariance matrices, i.e. 4x4 matrices. After that, we consider also the mean coordinate x, y of each cell as an additional feature to recognize the right area. Hence, we have 18 features total for each cells that represent our dataset.

We have in addition the labels 0/1 for each cell, so the label is 1 if the 70% of cell pixels belong to the gland otherwise 0.

Because of the dataset unbalance, the cells' labels at 0 are much more than those at 1, therefore, we remove the 30% of black cells.

In conclusion, we split the dataset into 70% train set and 30% test set to train the SVM classifiers.

5 Experimental Results

After dataset creation, we train a Support Vector Machine classifier to get our results. In particular, we chose the C-Support Vector Classification and tried different kernel types: linear, sigmoid, polynomial, RBF and RBF with best parameters.

As shown in the tables, the best results is obtained with RBF kernel, which is in the form of a Gaussian function, defined as:

$$K_{RBF}(x, x') = \exp\left(-\gamma \left\|x - x'\right\|^2\right) , \tag{3}$$

where $\gamma = \frac{1}{2\sigma^2}$, σ is a free parameter and x, x' represent a feature vectors in some input space, i.e., covariance matrices in our case. For every kernel

	SVM kernel				
Lower Glands	**linear**	**sigmoid**	**polynomial**	**RBF**	**RBF-BP**
Accuracy	0.74	0.70	0.75	0.92	0.92
F1-score	0.37	0	0.37	0.88	0.87
Selected Meibomian	0.26	0	0.25	0.90	0.90
Lost Meibomian	0.74	1	0.75	0.10	0.09
Incorrectly selected part	0.05	0	0.02	0.05	0.06

Table 1: Table shows the lower glands results for different SVM kernel types: RBF kernel provides the best results for every metrics. **RBF-BP** represents the RBF kernel with the best C and gamma parameters.

	SVM kernel				
Upper Glands	**linear**	**sigmoid**	**polynomial**	**RBF**	**RBF-BP**
Accuracy	0.70	0.70	0.74	0.92	0.92
F1-score	0	0	0.32	0.86	0.86
Selected Meibomian	0	0	0.21	0.85	0.86
Lost Meibomian	1	1	0.79	0.15	0.14
Incorrectly selected part	0	0	0.02	0.05	0.06

Table 2: Table shows the upper glands results for different SVM kernel types: RBF kernel provides the best results for every metrics. **RBF-BP** represents the RBF kernel with the best C and gamma parameters.

type used, we defined a pipeline with **make_pipeline**[2] function: first data are standardized with **StandardScaler**[2] function, then it's applied the Principal Component Analysis (**PCA**)[2] and finally the SVM is trained.

The StandardScaler function standardize features by removing the mean and scaling to unit variance, so the standard score of a sample x is calculated as z = (x - u) / s where u is the mean of the training samples and s is the standard deviation of the training samples.

PCA is a dimensionality-reduction method that is often used to reduce the dimensionality of large datasets, by transforming a large set of variables into a smaller one that still contains most of the information in the large set. Reducing the number of variables of a data set naturally comes at the expense of accuracy, but it's a gain in terms of dimensionality; because smaller datasets are easier to explore and visualize and make analyzing data much easier and faster for classifiers without extraneous variables to process.

[2]scikit-learn library

We set the 99.5% as number of components to keep.

The best parameters, for RBF-BP, are find through **GridSearchCV**[2] function which tries all the combinations of the values passed in the C and gamma dictionaries, that are predefined values for hyperparameters, and evaluates the model for each combination using the Cross-Validation method. Hence after using this function we get accuracy/loss for every combination of hyperparameters and we can choose the one with the best performance. In our case, we have:

- **C** = 0.01, 0.1, 1, 10;
- **gamma** = 0.001, 0.01, 0.1.

It was used several metrics to evaluate our results. The accuracy is obtained by the **accuracy_score**[2] function that computes the fraction of correct predictions, therefore, if the entire set of predicted labels for a sample strictly match with the true set of labels, then the subset accuracy is 1.0; otherwise it is 0.0. The second metric is **f1_score**[2] can be interpreted as a weighted average of the precision and recall, where the precision is the number of true positive results divided by the number of all positive results, including those not identified correctly, and the recall is the number of true positive results divided by the number of all samples that should have been identified as positive. Then, F1-score is the harmonic mean of precision and recall:

$$F1 = \frac{2 * (precision * recall)}{precision + recall} \, . \tag{4}$$

Selected Meibomian, **Lost Meibomian** and **Incorrectly selected part** are metrics created by ourselves. The first one represents the fraction of meibomian's area that is correctly recognized in all images, vice versa, the third one is the gland's area that is incorrectly found for every image. Lost Meibomian is the fraction of the meibomian's area that is not selected as part of the gland.

Tables show a better results with lower glands than the upper ones, probably because lower glands have a larger extension on dataset images and therefore they are more easily to recognize due also to Gabor filter use.

6 Conclusions

In this report, we described the meibomian recognition for lower and upper glands.

The classification shows very good results by using a SVM classifier, which gets in input a features covariance matrix extracted from Gabor filter application to all 10x10 cells obtained on every dataset image.

The best results are with RBF kernel even better than RBF-BP, because this last selects a wrong larger area of the image that do not contains meibomian glands.

The Gabor filter, basically, extracts several features that allow us to recognize the correct image's area, not picking wrong areas that do not correspond to meibomian.

Lower glands are more easily to recognize than the upper ones, probably because they have a larger extension on dataset images even if the results are great for both of them.