Classification of Optical Coherence Tomography images between Diabetic Macular Edema or Normal using Machine Learning.

Bruno Soares de Castro
up202309106@up.pt

Faculty of Engineering at the University of Porto - FEUP
International Master in Computer Vision - IMCV

Introduction

This report details the complete methodology for the automatic classification of Optical Coherence Tomography (OCT) images in order to classify medical images between Diabetic Macular Edema (DME) and Normal. To achieve this goal, we have designed a basic pipeline with simple steps, which is represented in Figure 1.

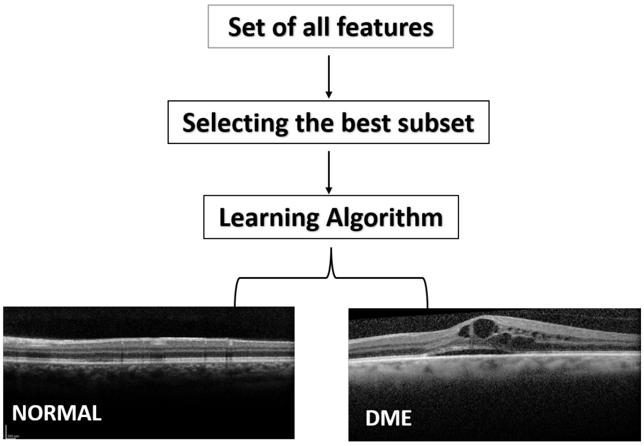

Figure 1: Main step of the proposed methodology

In the Methodology, we will explore in details all the pipeline that was designed to Figure 1.

Methodology

The methodology for implementing the recognition system involved several key steps, including the set of features, feature selection, and, finally, the learning algorithm. The following section outlines the detailed approach taken to achieve each of these tasks.

Dataset Image

The dataset consists of 50 images for DME and Normal, totaling a sum of 100 images.

The set of feature

The Setup contain 2 parts:

- **Create the mask**:
  - Region of interest (ROI): Mask and Perimeter
- **Texture dataset**:
  - `Basic Gray-Level Distribution Statistics (GLDS)`:
    - Parameters:
      - Dx : Array with X-coordinates of vectors denoting orientation. The default is [0, 1, 1, 1].
      - Dy : int, optional Array with Y-coordinates of vectors denoting orientation. The default is [1, 1, 0, -1].
    - Features:
      - Homogeneity, Contrast, Energy, Entropy and Mean.
  - `Statistical Feature Matrix (SFM)`:
    - Parameters:
      - Lr : Parameters of SFM. The default is 4.
      - Lc : Parameters of SFM. The default is 4.
    - Features:
      - Coarseness, Contrast, Periodicity and Roughness.
  - `First Order Statistics (FOS)`:
    - Parameters:
      - —
    - Features:
      - Mean, Standard Deviation, Median, Mode, Skewnewss, Kurtosis, Energy, Entropy, Minimal Gray Level, Maximal Gray Leve, Coefficient of Variation, Percentiles (10, 25, 50, 75, 90) and Histogram Width.
  - `Gray Level Co-ocurrence Matrix (GLCM)`:
    - Parameters:
      - ignore_zeros : Ignore zeros in image f. The default is True.
    - Features:
      - Angular Second Moment (Energy), Contrast, Correlation, Sum of Squares: Variance, Inverse Difference Moment (or Homogeneity), Sum Average, Sum Variance, Sum Entropy, Entropy, Difference Variance, Difference Entropy and Information Measures of Correlation (1-2).
  - `Zernikes' Moments`:
    - Parameters:
      - radius : Radius to calculate Zernikes moments. The default is 9.
    - Features:
      - Orthogonal moments invariants with respect to translation (1-25).
  - `Gabor Transform (GT)`:
    - Parameters:

  - deg: Quantized degrees. The default is 4 (0, 45, 90, 135 degrees)
  - freq: frequency of the gabor kernel. The default is [0.05, 0.4]
- ∗ Features:
  - Mean (1-8) and Standard Deviation (1-8) of the absolute value of detail sub-images of the GT of the image.

– `Fractal Dimension Texture Analysis (FDTA):`
- ∗ Parameters:
  - s : max resolution to calculate Hurst coefficients. The default is 3.
- ∗ Features:
  - Hurst coefficients (1-4)

– `Gray Level Size Zone Matrix (GLSZM):`
- ∗ Parameters:
  - —
- ∗ Features:
  - Small Zone Emphasis, Large Zone Emphasis, Gray Level Non-Uniformity, Zone-Size Non-Uniformity, Zone Percentage, Low Gray Level Zone Emphasis, High Gray Level Zone Emphasis, Small Zone Low Gray Level Emphasis, Small Zone High Gray Level Emphasis, Large Zone Low Gray Level Emphasis, Large Zone High Gray Level Emphasis, Gray Level Variance, Zone-Size Variance and Zone-Size Entropy.

– `Local Binary Patterns (LBP):`
- ∗ Parameters:
  - P : Number of points in neighborhood. The default is [8, 16, 24].
  - R : Radius/Radii. The default is [1, 2, 3].
- ∗ Features:
  - Energy (1-3) and entropy (1-3) of LBP image.

– `Law's Texture Energy Measures (LTE/TEM):`
- ∗ Parameters:
  - l : Law's mask size. The default is 7.
- ∗ Features:
  - Texture energy from LL kernel, Texture energy from EE kernel, Texture energy from SS kernel, Average texture energy from LE and EL kernels, Average texture energy from ES and SE kernels and Average texture energy from LS and SL kernels.

– `Gray Level Run Length Matrix (GLRLM):`
- ∗ Parameters:
  - Ng : Image number of gray values. The default is 256.
- ∗ Features:
  - Short Run Emphasis, Long Run Emphasis, Gray Level Non-Uniformity, Run Length Non-Uniformity, Run Percentage, Low Gray Level Run Emphasis, High Gray Level Run Emphasis, Short Low Gray Level Emphasis, Short Run High Gray Level Emphasis, Long Run Low Gray Level Emphasis and Long Run High Level Emphasis.

– `Discrete Wavelet Transform (DWT):`
- ∗ Parameters:
  - wavelet : Filter to be used. Check pywt for filter families. The default is 'bior3.3'
  - levels : Levels of decomposition. Default is 3.
- ∗ Features:
  - Mean (1-9) and Standard Deviation (1-9) of the absolute value of detail sub-images of the DWT.

– `Shape Parameters:`
- ∗ Parameters:
  - perimeter : Image N1 x N2 with 1 if pixels belongs to perimeter of ROI, 0 else.
  - pixels_per_mm2 : Density of image f. The default is 1.
- ∗ Features:
  - x-coordinate maximum length, y-coordinate maximum length, area, perimeter and perimeter2/area.

The package used for feature extraction was PyFeats version 1.0.1. For each function used in feature extraction, it includes the image and mask with 1 indicating pixels belonging to the ROI and 0 otherwise. In total, we have 144 features from each image.

In this way, we have a dataset composed of 100 individuals with 144 features.

Select the best subset and Learning Algorithm

In order to find the best subset and train the machine learning model, we created the following models:

Table 1: Evaluated models

	Selecting Subset	Learning Algorithm
Model 1	Forward Selection	K-Nearest Neighbors
Model 2	Embedded methods with regularization	Support Vector Machines

For this project, we decided to split the data into 90% for training and 10% for testing. Figure 2 gives the pipeline used for our final objective.

Figure 2: Main step of the proposed pipeline

The pipeline follows the same steps in both models. First, we have to scale our dataset. Second, the red block represents the part that will select the best subset. Finally, our model will use the previously selected subset. This way, we will guarantee that we will not commit any data leakage.

As we can see, this pipeline is part of GridSearchCV. We are performing cross-validation in order to find the best hyperparameters and, at the same time, the best subset. Figure 3 has the grid for model 1 and 2.

```
# Model 1 (KNN)
param_grid = {
    'sfs__estimator__n_neighbors': np.arange(3, 10, 3), # inner knn
    'knn2__n_neighbors': np.arange(3, 10, 3) # outer knn
 }

# Model 2 (SVC)
param_grid = [
# Linear
{'SFM__estimator__C': 2.**np.arange(4, 15, 1),
 'SFM__estimator__penalty': ['l1', 'l2'],
 'SFM__max_features': [5, 10, 20, 30, 40, 50],
 'SVC__kernel': ['linear'], 'SVC__C': 2.**np.arange(4, 15, 1)},

# radial basis function (rbf) kernel
{'SFM__estimator__C': 2.**np.arange(4, 15, 1),
 'SFM__estimator__penalty': ['l1', 'l2'],
 'SFM__max_features': [5, 10, 20, 30, 40, 50],
 'SVC__kernel': [ 'rbf'],
 'SVC__C': 2.**np.arange(-5,16, 1),
 'SVC__gamma': 2.**np.arange(-7,8,1)}
]
```

Figure 3: Grid-search over a parameter grid

For Model 2, we will use L1 and L2 regularization during the process of grid research to find the best subset, and the number of subset features (`max_features`) will be a part of our grid search.

Results

In order to extract the maximum information from the image using the texture, we decided to create a mask and perimeter from the ROI. In Figure 4, the results can be seen.

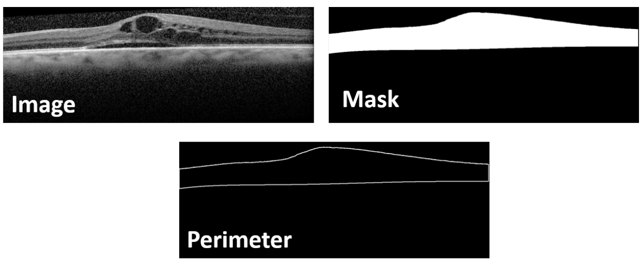

Figure 4: Mask and Perimeter processed from image

Figure 5 shows the cross-validation results for Model 1 using k-nearest neighbors, with the best parameter for the final model being `n_neighbors` = 3. The function that finds the best subset of features, `SequentialFeatureSelector`, is conducting another cross-validation as well. In this case, we are performing nested cross-validation, so the number of folds was 4 for `GridSearchCV`, and during the `SequentialFeatureSelector` process, it was also 4 folds.

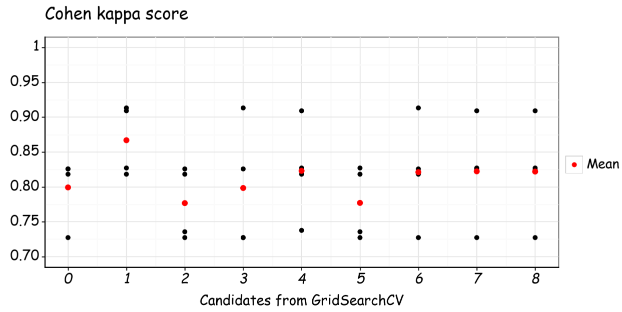

Figure 5: Cross-Validation Model 1

The best subset of features using Forward Selection, in general, was GLDS, SFM, FOS, GLCM, Zernike Moments, GT, FDTA, GLSZM, LBP, GLRLM, and SHAPE. In Appendix A, it is possible to see the name of each selected feature.

Figure 6 shows the cross-validation results for Model 2 using Support Vector Classification, with the best parameters for the final model being `C` = 2.0, `Kernel` = `'rbf'`, and `gamma` = 0.5. As we can see in Figure 3, the `kernel` for the final model was tested with `linear` and `'rbf'`.

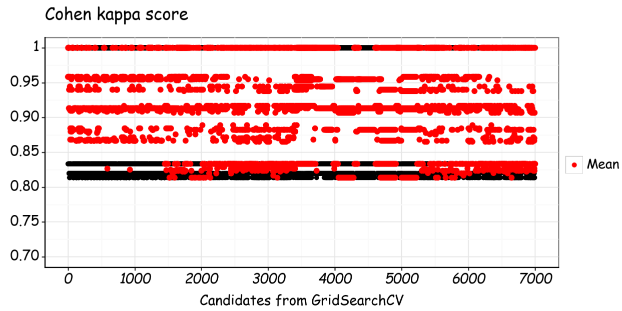

Figure 6: Cross-Validation Model 2

The function `SelectFromModel` is used to find the best subset of features, and `LinearSVC` is used as the estimator from which the feature importance is derived through a `coef_` attribute. The best hyperparameters for `LinearSVC` were `C` = 256.0 and `penalty` = `'l1'`. The best hyperparameter for `SelectFromModel` was `max_features` = 5. Only one cross-validation with 8 folds was carried out for Model 2.

The best subset of features using Embedded methods with `l1` regularization, in general, was GLDS, FOS, LBP and SHAPE. In Appendix B, it is possible to see the name of each selected feature.

Although the number of matches among the hyper-parameters for Model 2 is higher than for Model 1, Model 1 takes more time to run. The reason is that we are conducting another cross-validation within Model 1, and the Forward Select method itself takes more time.

In Table 2, the Cohen's kappa results are provided as the average of the folds from the best candidate obtained from GridSearchCV, Data Train, and Test. In terms of Data Train and Test, after finding the best hyperparameters, the models were refit based on Data Train, and Data Test was used for prediction and comparison.

Table 2: Cohen's kappa score for the final model.

	Cross-Validation	Data Train	Data Test
Model 1 (KNN)	0.8669	1	1
Model 2 (SVC)	1	1	1

In both cases, the models are performing excellently without any problems such as overfitting or underfitting. We have decided to present only one set of measures here, as the data from training and testing for the final `KNN` and `SVC` models yielded the same measures.

Table 3: The main classification metrics for the final models.

	precision	recall	f1-score	support Train	support Test
0	1.00	1.00	1.00	45	5
1	1.00	1.00	1.00	45	5
accuracy			1.00	90	10
macro avg	1.00	1.00	1.00	90	10
weighted avg	1.00	1.00	1.00	90	10

Conclusion

In this experiment, both models have presented excellent metrics of measures of performance. However, when considering the waiting time, the pipeline from the first model takes more time than the pipeline from the second model. The difference is almost three time greater.

The selection method has shown a significant disparity, not only in terms of speed but also in the number of variables. While model 1 selected 72 variables, model 2 selected only 5 variables, yet yielded the same performance.

For the future works perhaps should be important to identify how much important each feature really is and how this relationship between features and the label is.

A Feature selected Model 1

GLDS_Homogeneity, GLDS_Contrast, GLDS_ASM, GLDS_Entopy and GLDS_Mean.

SFM_Coarseness, SFM_Contrast, SFM_Periodicity and SFM_Roughness.

FOS_Mean, FOS_Variance, FOS_Median, FOS_Mode, FOS_Skewness, FOS_Energy, FOS_Entropy, FOS_MinimalGrayLevel, FOS_MaximalGrayLevel, FOS_CoefficientOfVariation, FOS_10Percentile, FOS_25Percentile, FOS_75Percentile and FOS_HistogramWidth.

GLCM_ASM_Mean, GLCM_Contrast_Mean, GLCM_Correlation_Mean, GLCM_InverseDifferenceMoment_Mean, GLCM_SumAverage_Mean, GLCM_SumEntropy_Mean, GLCM_Entropy_Mean and GLCM_MaximalCorrelationCoefficient_Mean.

Zernikes_Moments_radius_9_0, Zernikes_Moments_radius_9_1, Zernikes_Moments_radius_9_4, Zernikes_Moments_radius_9_5, Zernikes_Moments_radius_9_6, Zernikes_Moments_radius_9_7, Zernikes_Moments_radius_9_8, Zernikes_Moments_radius_9_9, Zernikes_Moments_radius_9_10, Zernikes_Moments_radius_9_12, Zernikes_Moments_radius_9_14, Zernikes_Moments_radius_9_17, Zernikes_Moments_radius_9_19, Zernikes_Moments_radius_9_21 and Zernikes_Moments_radius_9_22.

GT_th_0.0_freq_0.05_std, GT_th_0.0_freq_0.4_std, GT_th_1.0_freq_0.05_mean, GT_th_1.0_freq_0.05_std, GT_th_1.0_freq_0.4_std, GT_th_2.0_freq_0.05_mean, GT_th_2.0_freq_0.05_std, GT_th_2.0_freq_0.4_std and GT_th_3.0_freq_0.05_std.

FDTA_HurstCoeff_1, FDTA_HurstCoeff_3 and FDTA_HurstCoeff_4.

GLSZM_SmallZoneEmphasis, GLSZM_LargeZoneEmphasis, GLSZM_GrayLevelNonuniformity, GLSZM_ZoneSizeNonuniformity, GLSZM_ZonePercentage, GLSZM_LowGrayLeveLZoneEmphasis, GLSZM_SmallZoneHighGrayLevelEmphasis, GLSZM_LargeZoneLowGrayLevelEmphassis and GLSZM_ZoneSizeVariance,

LBP_R_1_P_8_entropy.

GLRLM_GrayLevelNo-Uniformity and GLRLM_LongRunLowGrayLevelEmphasis.

SHAPE_XcoordMax and SHAPE_area.

B Feature selected Model 2

GLDS_Homogeneity.

FOS_Entropy.

GT_th_3.0_freq_0.05_std.

LBP_R_2_P_16_energy.

SHAPE_area.