Exercise 6: ROC curves, image retrieval

Multimedia systems

2018/2019

Create a folder `exercise6` that you will use during this exercise. Unpack the content of `exercise6.zip` that you can download from the course webpage to the folder. Save the solutions for the assignments as the *Matlab/Octave* scripts to `exercise6` folder. In order to complete the exercise you have to present these files to the teaching assistant. Some assignments contain questions that require sketching, writing or manual calculation. Write these answers down and bring them to the presentation as well. The tasks that are marked with ★ are optional. Without completing them you can get at most 75 points for the exercise (the total number of points is 100 and results in grade 10). Each optional task has the amout of additional points written next to it, sometimes there are more optional exercises and you do not have to complete all of them.

Introduction

The exercise includes two assignments: In the first one you will get to know ROC curves that can be used to evaluate and compare classifiers and retrieval systems. In the second assignment you will implement and test several image retrieval systems that operate on different features and evalue them using ROC analysis.

Assignment 1: The theory of ROC analysis

The purpose of this assignment is to learn the theory and practical use of ROC analysis. Therefore you should first read a paper about ROC curves [1] that is available in the exercise material bundle (especially pay attention to sections 1–5 and 7–8). In the following tasks you will work on a given theoretical example of two classifiers (C_1 and C_2). We have a set of samples that we whish to classify in one of two classes and a ground truth class of each sample (denoted as 0 and 1). For each sample a classifier gives use a score based on which we can determine to which class should the sample belong to (score closer to 0 means class 0, score closer to 1 means class 1). Below are the results for 8 samples, their ground truth values (ξ_{id}) and the score values for both classifiers (ξ_{C_1} and ξ_{C_2}).

$$\begin{array}{lcccccccccc} \xi_{\text{id}} = & [& 1 & 0 & 1 & 1 & 1 & 0 & 0 & 0 &] \\ \xi_{C_1} = & [& 0.5 & 0.3 & 0.6 & 0.22 & 0.4 & 0.51 & 0.2 & 0.33 &] \\ \xi_{C_2} = & [& 0.04 & 0.1 & 0.68 & 0.22 & 0.4 & 0.11 & 0.8 & 0.53 &] \end{array} \tag{1}$$

(a) For the example above calculate and draw the ROC curves (by hand) for classifier C_1 as well as classifier C_2. Also calculate the area under the curve (AUC) for both

classifiers. For the classifier C_1 select a decision threshold (working points) $\vartheta_{th1} = 0.33$ and use it to calculate the confusion matrix and the F measure score. Do the same thing for the classifier C_2 using a threshold value $\vartheta_{th2} = 0.1$.

Question: Based on theory from [1] decide which classifier is better in the selected working points and motivate your decision. Which working point is optimal for each classifier?

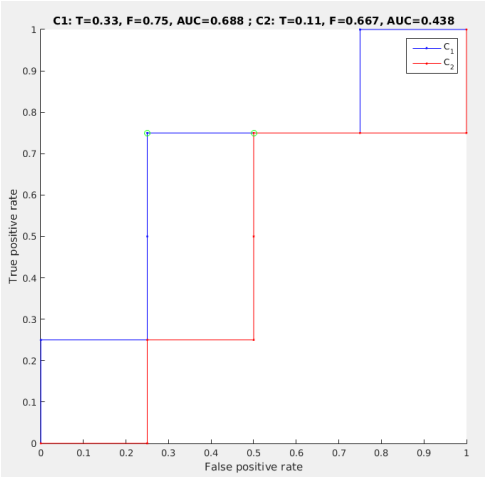
(b) You will now calculate ROC curves for *combined classifiers*. We have two binary classifiers, C_1 ($\vartheta_{th1} = 0.33$) and C_2 ($\vartheta_{th2} = 0.1$), which means that the classifier C_1 classifies a sample as class 1, if its score is $\xi_{C_1}(\mathbf{x}_i) > \vartheta_{th1}$, otherwise it classifies it as class 0, similarly can be said about classifier C_2. The first combined classifier C_3 can be obtained as the intersection of decisions of the two basic classifiers, $C_3 = C_1 \bigwedge C_2$ (C_3 classifies a sample as class 1 if both basic classifiers classify it as class 1). The second one can be obtained as an union of the decisions of the two basic classifiers, $C_4 = C_1 \bigvee C_2$ (C_4 classifies a sample as class 1 if at least one of the basic classifiers classifies it as class 1). For each combined classifier calculate and plot its point in the ROC space together with the ROC curves from previous tasks.

(c) Implement function `[R, auc] = get_roc(scores, groundtruth)` that expects a vector `scores` (ξ_{score}) and a vector `groundtruth` (ξ_{id}) as an input and returns a ROC curve in matrix `R` as well as the area under the curve value in variable `AUC`. The ROC curve should be encoded as a $2 \times N$ matrix, the first row includes *true positive rate* values, the second row includes *false positive rate* values. You should study the Algorithm 1 in [1] for the implementation. Validate the correctness of the algorithm by computing the ROC curve on the data from the first task. The curves should be the same for each classifier.

(d) In the paper [1] (page 2) the author explains how to select an optimal working point for a classifier from a ROC curve. The threshold should be at the point on the curve that is the closest to the point $[0, 1]$ in the ROC space. Extend your function `get_roc` that it will also calculate the optimal working point for the ROC curve and that it will return its position and the corresponding threshold value and F score value as additional results. Test your code on classifiers C_1 and C_2. Based on the AUC decide which classifier is performing better over all threshold values, based on the F score at the optimal working point decide which classifier is better at this point. Write a script that calculates and draws ROC curves for classifiers C_1 and C_2 on the same plot, for both classifiers mark the optimal working point and set the title of the plot to display values of the thresholds, F scores and AUC for both classifiers. You can use the following snippet as an example how to set the title.

```
1      title(sprintf('t = \%1.3g, F = \%1.3g,', 0.1, 0.4));
```

Make sure that you get the same results that you have calculated manually.



Assignment 2: Image retrieval using histograms and correlation

In this assignment you will implement several image retrival systems. The input to the system is a query image and the system should return the images in the database sorted by similarity to the query image. For each approach you will have to extract features from all the images, then compare your query image to all of the database images. You will also have to compare the performance of different features over all the images in your dataset by calcualting a $N \times N$ similarity matrix.

You will test your retrieval systems on the Caltech 101 dataset [1]. It consists of approximately 9000 color images from 101 classes. You can use the funtion `prepare_caltech` to extract the images and their corresponding classes. The function returns a list of images along with the corresponding class labels. You can modify it to only extract (or ignore) specific classes and/or to resize the images for faster processing if you wish. For developing the system you are advised to use a smaller subset of classes with a smaller number of samples to speed up the computation (or alternatively, a subset of easier classes, e.g. `stop_sign`, `faces`, `strawberry` etc.). If you encounter problems because of the high difficulty of the Caltech dataset, you can also use a much easier dataset that is included in the `images` directory. Note that you *must* test your systems on the Caltech dataset.

(a) Implement the classifier based on color histograms. Write a function that computes 3D color histograms for all images in your database[2].

To compute the distance between the reference histogram and every other histogram in the database you will use Hellinger distance that is defined as:

$$H(\mathbf{h}_1, \mathbf{h}_2) = \sqrt{\frac{1}{2} \sum_{i=0}^{N-1} \left(\sqrt{h_1(i)} - \sqrt{h_2(i)} \right)^2}. \tag{2}$$

[1] `http://www.vision.caltech.edu/Image_Datasets/Caltech101/`

[2] Use the function `myhist3` in the RGB color space (it is available in the exercise material), then reshape the 3D histograms to 1D histograms and stack them together in a 2D matrix (the matrix is composed in a way that the i-th row includes the histogram of the i-th image).

Note that low values of Hellinger distances signify high similarity and high values signify low similarity. This is exactly the opposite to what is expected by your ROC curve function that you will use in the following tasks. This can be fixed by redefining the histogram distance measure. If we assume that $H(\mathbf{h}_1, \mathbf{h}_2)$ denotes the Helliner distance between histograms $\mathbf{h}_1$ and $\mathbf{h}_2$, we can define the new distance simply as

$$\rho(\mathbf{h}_1, \mathbf{h}_2) = 1 - H(\mathbf{h}_1, \mathbf{h}_2). \tag{3}$$

Write a script that tests your system by loading the database (use 8 bins per color channel), using the fifth image in the database as a reference image. Compute the distances to all other images, sort the distances and display the first five matches and the reference image in a same figure (use the function `sort` to perform sorting; as noted in the documentation this function can also return the indices of sorted elements in the original vector).

(b) Implement a system that uses normalized cross-correlation of grayscale images. The normalized cross-correlation between two sequences of same size $\mathbf{X}$ and $\mathbf{Y}$, denoted as $NCC(\mathbf{X}, \mathbf{Y})$ is defined as scalar product between normalized sequences[3]:

$$NCC(\mathbf{X}, \mathbf{Y}) = \frac{1}{N} \frac{\sum (x_i - \bar{x})(y_i - \bar{y})}{\left(\sqrt{\frac{1}{N}\sum (x_i - \bar{x})^2}\right)\left(\sqrt{\frac{1}{N}\sum (y_i - \bar{y})^2}\right)}. \tag{4}$$

where $\bar{x}$ and $\bar{y}$ denote the mean values of the elements in the sequences. Sequences are more similar if the correlation is higher. We can compute normalized cross-correlation for grayscale images of same size if we reshape them into vectors of intensity values. Repeat the testing of the system in the same manner than in the previous task, load images, convert them to grayscale, select a reference image, compute the distances and display the first five matches. What is the system sensitive to?

(c) Implement a retrieval system that uses CNN features to calculate image similarity. You will need the MatConvNet library that can be found here: `http://www.vlfeat.org/matconvnet/`. Use MatConvNet and a pretrained neural network you can find online[4]. The choice of the neural network architecture is up to you. Note that architectures of the models are different and you need to check the model documentation to find out which layer contains useful features (but generally it is the penultimate layer). Its features should be one dimensional vectors that you can compare using the Hellinger distance. Plot the ROC curve for the system and comment on its performance related to the NCC and color histogram systems. You can help yourself using the following code snippet:

```
net = load('imagenet-vgg-f.mat');
im = list{i};
im_ = single(im);
im_ = imresize(im_, net.meta.normalization.imageSize(1:2));
im_ = im_ - net.meta.normalization.averageImage;
```

[3]For more information see `http://en.wikipedia.org/wiki/Cross-correlation`

[4]`http://www.vlfeat.org/matconvnet/pretrained/#imagenet-ilsvrc-classification`

```
res = vl_simplenn(net, im_);
feat = res(20).x(:);
```

(d) ★ (5 points) Extend the histogram method by including spatial information. Divide the input image into subregions, calculate the histogram for reach region, then concatenate the histograms into a final feature vector. Experiment with different numbers of subregions and comment on the performance of the new method.

(e) ★ (15 points) Implement an image retrieval system of your own design that will outperform both the histogram and the NCC systems. Create something unique with the methods of your choosing. You have to prove the system's performance using ROC curves.

(f) The image retrieval systems can currently return distances between each image in the database to the reference image. If we want to build a binary classifier we have to determine the optimal threshold that can be used to decide if a classifier belongs to the class of the reference image or not. We will use ROC analysis that you have implemented in the previous assignment. Use the ROC curves to determine optimal threshold. You can use an image in the database as a reference image and compute distances to every other image in the dataset. If we select the first image we can generate a ROC curve for that image, however, this curve only tells us the properties of the system for this input and may not be generalizable. If we select a different image we can get a completely different ROC curve with a different optimal point. Instead, use the following procedure to compute the ROC curve over all images in the database.

   (a) Load the database.

   (b) For each image in the database:

      i. Use the selected image as a reference image.

      ii. Compute the distance between the reference image and the remaining images in the database and store the distances to vector ϑ_{score} (the vector must not include distance to the reference image itself).

      iii. Use the provided label ground truth data to compose a vector of binary ground truth for the given reference image by comparing it to the class of the reference image. Save the binary ground truth to vector ϑ_{class}.

      iv. Extend the overall vectors for scores and ground truth by appending the new data: $\varphi_{\text{class}} = \varphi_{\text{class}} + \vartheta_{\text{class}}$ and $\varphi_{\text{score}} = \varphi_{\text{score}} + \vartheta_{\text{score}}$.

   (c) Use the composed overall vectors φ_{score} and φ_{class} to compute the ROC curve.

   Evaluate all three systems by plotting their ROC curves and compare them. Determine the optimal threshold value, plot it in the ROC space (on top of the ROC curve). Do not forget to label the axes of the plot.

(g) ★ (20 points) Implement a bag-of-words approach for image retrieval. First, extract local features from each image (you can use SIFT or something similar), then generate a codebook via a clustering method (as described here `https://en.wikipedia.org/wiki/Bag-of-words_model_in_computer_vision#Codebook_generation`). The image feature vectors are then histograms of the codewords and can be used for comparison. See the paper [2] for further info on implementation details.

References

[1] T. Fawcett. An introduction to ROC analysis. *Pattern Recognition Letters*, 27(8):861–874, 2006.

[2] Jialu Liu. Image retrieval based on bag-of-words model. *arXiv preprint arXiv:1304.5168*, 2013.