EE 58*J* Data Mining Project 1

Oğuzhan Sevim
April 5, 2020

I. Introduction

In this project, we will perform image recognition on the Vispera-SKU101 – 2019 dataset, where the recognition process will be mainly composed of obtaining descriptors and using a statistical classifier called Nearest Neighbor on these descriptors. Python programming language is preferred for this assignment.

II. Data

Vispera-SKU101 – 2019 data is composed of approximately 10.000 arbitrary–sized images in jpeg format. Each image belongs to one of 101 classes, and is kept in the folder named with SKU (stock keeping unit) it belongs to. Each of these class folders contains approximately 100 instances/images. An example image is given in Figure 1 in which the image belongs to class of "33*cl* Fanta in can" with the class SKU of 8736.

Fig. 1. An example image with 75×224 pixels. The image belongs to "33*cl* Fanta in can" class with SKU of 8736

The data is first downloaded to the local computer, and then retrieved one by one for further processes. In order to navigate easily between the local folders, "os" library is used.

III. Image Descriptors

After resizing all the images to 128×128 pixels, two types of descriptors are extracted from each normalized image. After normalization, each image is partitioned by $n \times m$ sized grids. The final descriptor of an image for a given descriptor type is then obtained by concatenation of these window descriptors taken column-wise.

For the descriptor extraction processes of the project, a widely used image processing and computer vision library called OpenCV is used. Along with the descriptors, some details of how to use OpenCV for our project's purposes is given in Sections III-A and III-B.

A. Color Histogram

The first descriptor type used is the color histograms of the images. In order to find the color histogram of a given image, we first read the image in BGR (Blue, Green, Red) format by using imread() function of openCV library. After converting BGR image into HSV format by cvtColor() function, the color histogram for each H,S, and V matrices are found separately by using calcHist() function of OpenCV. One of the best ways of this function is that it allow using masks on the image so that it only finds the histogram of certain part of the image. By patching the histograms of H,S, and V frames together, we obtain the color histogram of a given frame. Finally, the color histogram descriptor of an image is obtained by concatenating the histograms of all windows in a column-wise manner.

B. Gradient Direction Histogram

As the second type of descriptors, gradient direction histograms are used. First, each BGR image is converted into Gray scale images. Then, by using sobel() function of OpenCV, vertical and horizontal edges of the image are filtered (with different filter options) separately for each window. Consequently, the gradient directions are calculated by using the lengths of horizontal and vertical change vectors with the help of arctan2() function. Finally, the overall gradient direction descriptor is obtained by concatenating the histograms of all windows in a column-wise manner.

IV. Nearest Neighbor Classifier

After obtaining the descriptors from images, Nearest Neighbor (NN) method is used for classification of the test images. By using NN algorithm, we simply assign each test image to the class of its nearest neighbor in the descriptor space. This algorithm is implemented by using Scikit Learn library's K-Nearest Neighbor classifier function KNeighborsClassifier() by choosing number of neighbors $K = 1$.

V. Tuning the Hyperparameters

In this section, we investigate the effects of histogram bin number and the size of grids for different types of descriptors. In Tables I and II, accuracy of Color histogram descriptors and Gradient direction descriptors are respectively shown for different hyperparameters. In these tables, each row corresponds a different grid scheme from 1×1 to 8×8, where each column corresponds to different histogram bin number from 10 to 40. The performances of different experiments with combined descriptors are shown in Table III. In this experiments $L1$ norm is used as the distance metric of NN algorithm.

TABLE I
ACCURACIES FOR COLOR HISTOGRAM DESCRIPTOR

	10	**20**	**30**
1x1	51,47	56,39	58,08
2x2	58,66	62,18	62,42
4x4	64,06	65,61	65,12
8x8	66,04	65,89	65,80

TABLE II
ACCURACIES FOR GRADIENT DIRECTION HISTOGRAM DESCRIPTOR

	10	**20**	**30**
1x1	10,37	13,99	17,51
2x2	27,01	30,73	33,38
4x4	40,89	42,06	43,61
8x8	42,64	43,27	43,80

TABLE III
ACCURACIES FOR COMBINED DESCRIPTOR

	10	**20**	**30**
1x1	54,70	60,15	60,49
2x2	62,33	65,27	65,17
4x4	67,20	67,68	67,53
8x8	68,06	69,66	68,93

From Tables I, II, and III we can observe that increasing the grid size generally increases the performance. The same relation also seems to hold for histogram width (bin number) but with diminishing marginal return. From these tables, also by taking run time into account, we may choose to continue with a combined descriptor with 8×8 grid size and histogram width of 20.

Next, we will try to optimize our classifier further by changing the norm used in NN algorithm and the size of gradient filter. Please note that the above results are obtained by using $L1$ norm and 3×3 sobel filter. When we experiment with different filters norms we get the results shown in Table IV. Most obvious interpretation of Table IV seems to be the superiority of $L1$ norm over $L2$. Additionally, there seems to exist no dramatic performance difference between Sobel and Scharr filters.

TABLE IV
ACCURACIES OF CLASSIFIER WITH DIFFERENT NORMS AND FILTER TYPES

	sobel 3x3	**sobel 5x5**	**scharr 3x3**
L1	69,66	69,51	69,56
L2	56,15	55,91	56,01

VI. CATEGORY–WISE ACCURACIES

From the experimental results given in previous section, we design our classifier with the following choices:

- Combination of color histogram and gradient direction histogram descriptors
- 8×8 grid size,
- Histogram with 20 bins
- $L1$ norm used in NN algorithm
- 3×3 sobel filter in gradient search

When we apply the above combination for the classifier, classification accuracies of each category is found as below:

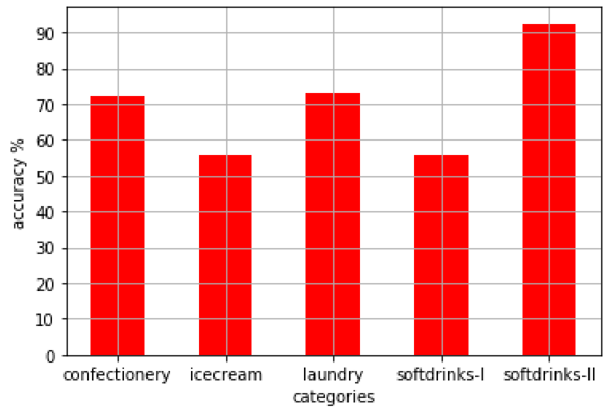

Fig. 2. Accuracies of each category

From Figure 2, we see that the classification accuracy of softdrink-II category is considerable higher than the others, whereas icecream category has the lowest accuracy. When we investigate the images in softdrinks-II category, we observe some kind of concurrency in terms of rotational orientation and contrast. An almost opposite observation can be made by looking at the icecream category images.

VII. CLASS WITH THE LOWEST ACCURACY

Class with the lowest accuracy is found to be *sku*.70 (Carte D'or Mini Çikolata Karnavalı) with accuracy of 20%. When we look at its images, we see that contrast deviate a lot through the images. This may make color histograms very dissimilar from image to image. Also, we observe the images were taken with different rotational orientations. This again may cause dissimilarities between the gradient direction histograms of different images of the same class. Rotational disorientation and contrast differences are illustrated in Figure 3.

As a final note, I believe that taking the histogram of gradient directions should eliminate the problem of rotational disorientation. By only rotating an image, we should probably end up with the same gradient direction histogram. So, the reason for this low accuracy could be contrast issue only. For time related reasons, I couldn't check that idea for now.

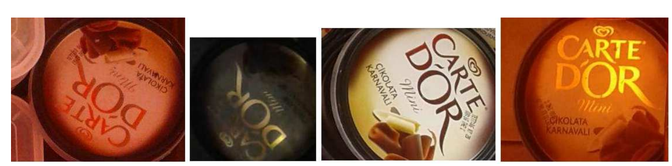

Fig. 3. Some instances from worst accuracy