EE 58J | Data Mining for Visual Media | Spring 2020
Assignment I

Product Image Recognition Challenge 2020 Part I

Here is the first part of the Product Image Recognition Challenge 2019 organized in the context of the EE 58J course. The challenge will continue during the whole semester and consists of experimenting with some standard visual descriptors and statistical classifiers for the task of automatically recognizing the label of a product package from its image. Typical examples of such images are shown below.

In this assignment, you will use the Vispera-SKU101-2019 dataset (©Vispera Information Technologies), which contains 101 different product SKUs (stock keeping unit as called in retail business, these stand for object classes in visual recognition terminology). The number of classes and the associated number of image instance are tabulated below. In the dataset folder structure, each class instances are placed under their corresponding folder. The dataset is not separated into training and test sets. In your subsequent experiment in this assignment, use 80% of the whole dataset as training instances (roughly 80 instances/class) and the remaining 20% as test instances (roughly 20 instances/class). Note that you should randomly sample each class to split the set into training and test subsets.

Product Category	Nr. Classes	Nr. Instances
Soft drinks-I	20	2030
Household	20	2081
Confectionery	20	2065
Ice cream	21	2140
Soft drinks-II	20	2045

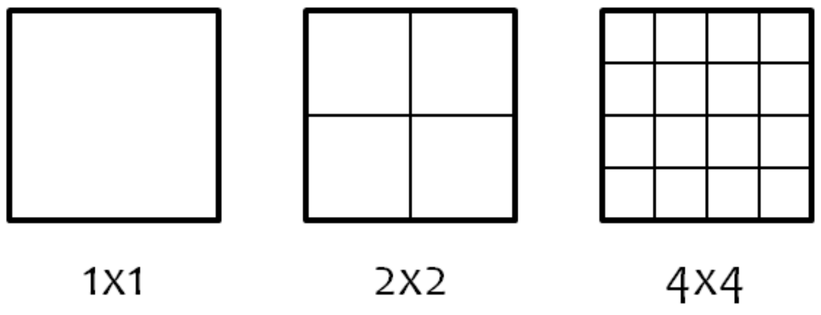
You will experiment with visual patch descriptors and the nearest-neighbor classifier and you will report the recognition performance across a set of algorithmic parameters. The implementation steps of the assignment are described as follows.

(1) Size Normalization. In subsequent steps, you will use size-normalized images. As a preprocessing step, normalize the size of each image in the dataset into 128-by-128 pixels. (Don't worry about losing the aspect ratio information)

(2) Image Description. In this step, you will implement two types of patch descriptors and aggregate them as global image descriptors. *A patch descriptor in the present context is the histogram of a certain visual characteristic (feature) computed over a rectangular image window.* The methods you will implement are as follows:

- **Color Histogram:** Convert the RGB values of the pixels in the window into HSV (Hue-Saturation-Value) and aggregate each feature into a histogram to obtain three histograms (one for Hue, one for Saturation and one for Value) for the window. Normalize each histogram to have unit L1-norm (i.e., sum of absolute values adds up to 1 for a L1-normalized histogram). The concatenation of the three histograms yields an overall color histogram descriptor for the window.
- **Gradient Orientation Histograms:** Convert the image into grayscale. Compute the horizontal and vertical components of the gradient at each pixel (by filtering the image with appropriate gradient filters), then calculate the gradient orientation angle. Aggregate the gradient orientation angles within the image window into a histogram. Again, normalize the result to have unit L1-norm.

Note that bin size and binning method in these histograms are design and experimentation parameters. In your implementation, these should be parametric (one should be able to provide them to your routine as input). These descriptors can be computed at arbitrarily sized rectangular windows. In this assignment, you will experiment with the spatial image gridding options shown below.



Each cell in each grid pattern above corresponds to an image window on which a patch descriptor should be computed as described. For instance, in the 1x1 grid, the whole 128-by-128 image is taken as a cell and all three descriptors are computed on that single cell; in the 2x2 grid, the descriptors are computed on 4 cells of size 32-by-32; in the 4x4 grid, likewise. The final descriptor of the image for a given descriptor type is thus the concatenation of the cell descriptors taken column-wise. The final outputs of this step should be a concatenated color histogram descriptor and a concatenated gradient orientation histogram descriptor for each image.

(3) Nearest Neighbor (NN) Classification. The NN classifier is conceptually the simplest statistical classifier. It memorizes all the descriptors in the training set along with their class labels. Given a query image (a testing image with unknown class label), we compute its descriptor and compare it to all the descriptors in the training set using a suitable distance metric, e.g., the L1-distance (sum of the absolute differences between descriptor dimensions). The class label of the closest descriptor in the training set is assigned as an estimate for the unknown class label of the query image. A natural variant of this classifier is one where we take a majority vote across K closest descriptors. This variant is called the K-NN classifier, which is known to have less variance than the plain NN classifier, but it might have a higher bias in terms of statistical accuracy. Optimal setting of K is performed by cross-validation (These aspects are not in scope of the present assignment and will be covered later in the course).

(4) Classifier Combination. Since we have three different types of descriptors encoding different sorts of visual information, it's natural to combine them to get better results. One possible option for this (that *you're NOT asked* to implement in this assignment) is to concatenate all three descriptors into a big one and apply the NN classifier on top of this combined descriptor. The other option (that *you're asked* to implement in this assignment) is to do the combination at a later stage. That is, the distances coming from different descriptor types are averaged to yield a single distance that will be used to retrieve training set instances by the NN procedure.

Reporting. After implementing the above steps, you're ready to report the performance of the above descriptors across a selection of experimental parameters. The associated tasks are:

- Experiment with different histogram binning and bin size options, these will affect the descriptor dimension. You're free to experiment with whatever options you like in order to optimize the performance.
- Experiment with different image gridding options: 1x1, 2x2 and 4x4 are must, finer decompositions are optional.
- Experiment with different distance measures: L1 is must, other options such as L2, Chi-Square, etc. are optional.

You will report the performance of all options above in terms of classification accuracy measured on the testing set. The classification accuracy is simply defined as the ratio of correctly classified testing instances to all available testing instances. Note that you have to report performance for (1) color histograms, (2) gradient orientation histograms, and (3) their combination as described above.

Report your results for each product category separately and provide the aggregate performance statistics as well. Make a visual summary of the most confused class to see the limitations of the implemented methods.