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

Product Image Recognition Challenge 2020
Part IV

Here is the fourth part of the Product Image Recognition Challenge 2020 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 experiment with the concept of data augmentation. You should use the *imgaug* image augmentation library, which is available here https://imgaug.readthedocs.io/en/latest/index.html

Data augmentation is a methodology that comes handy in data-scarce situations. The basic idea is to augment the training set by applying a number of parametrized transformations to each of the instances so that you obtain a bigger set for training (albeit artificially generated). The *imgaug* library comes with a very good documentation with examples.

The number of classes and the associated number of image instances in your standard data set are tabulated below. In the dataset folder structure, instances from each class are placed under their corresponding folder. You will run your experiments only on the **Confectionery** and **Ice cream** categories separately (shown in **red bold** in the table below), that is, you have **two** distinct classification problems according to the table below.

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

Experimental Setup. For each of the product categories in the scope of this assignment (**Confectionery** and **Ice Cream**), you will experiment with the following configurations.

Configuration	Original Training set	Augmented Training Set[1]	Test Set[4]
Baseline10	~10 instances per class	No augmentation[2]	~90 instances per class
Augmented10	~10 instances per class	Free to choose[3]	~90 instances per class
Baseline20	~20 instances per class	No augmentation[2]	~80 instances per class
Augmented20	~20 instances per class	Free to choose[3]	~80 instances per class
Baseline50	~50 instances per class	No augmentation[2]	~50 instances per class
Augmented50	~50 instances per class	Free to choose[3]	~50 instances per class

(1) Make sure you include the original training set into the augmented set. If you need (cross)-validation, this will also serve as such.
(2) Don't apply any augmentation here. This will serve as your baseline case that you are trying to improve.
(3) You are free to choose the type and size of the augmentation. You are only limited by your imagination and your computational resources ☺
(4) Do not augment the test set whatsoever, do not include any augmented set into it. Don't make any validation on the test set. Use this strictly to report final performance for a given experimental configuration. Any violation to this rule will only fool yourself.

Feature Extraction. You may use your favorite feature extractor.

Classifier. You may use your favorite classifier.

Bonus scheme. You may also follow a CNN-based approach for the above steps. In fact, you are encouraged to do so (bonus points will be provided if appropriately implemented and reported). You are free to choose whatever architecture you like, or you are comfortable with (AlexNet, Inception, ResNet, etc.).

Cautionary remark: If you follow this path, do not try to train the network from scratch, use it in either of the following approaches:

(i) Use the standard model pretrained on a different data set (e.g., ImageNet) to extract features and choose your favorite classical classifier (e.g., SVM) on top of these features (similarly to what you did in Assignment III, with the exception that here you are not provided with precomputed logits, you will have to compute them yourself).
(ii) Fine-tune the pretrained network with your data to obtain an end-to-end classifier.

Specify which approach you have taken.

Reporting. Report the test performance for each product category and configuration separately and using descriptive graphs. If you use any form of validation, specify the type and details. If you use cross-validation, show the cross-validation performance for different parameters visually, describe the method with which you find the best parameters and clearly tabulate them.