ECSE-415 Introduction to Computer Vision
Final Project: Classification and Segmentation

Due: 9th April 2020, 11:59PM

1 Introduction

For this project, you will form teams of 4-5 people. Each team will participate and compete in an in-class Kaggle Challenge to segment (32 points) and classify (68 points) flowers from images. For this project, your team will use the 102 Flowers dataset [1]. The datasets for both tasks are provided with this project description. The objective of the project is to permit your team to explore the workings of the features and methods described in class. The classification task will require your team to extract features to be able to classify the flower type within a given image. Your team will then write code to segment the flowers from the images. The terms with asterisks* are defined in Section 6.

Projects should be submitted electronically via myCourses. All submitted material is expected to be the students' own work or appropriately cited (See Academic Integrity guidelines in the Syllabus). Software packages (e.g. scikit-learn, MATLAB toolboxes, Keras, PyTorch, etc.) which implement machine learning-related algorithms (SVM, RF, k-fold cross-validation, etc.) are allowed. Appropriate citations are expected. The use of external code without citation will be treated as plagiarism. Projects received after the deadline up to 24 hours late will be penalized by 30%. Projects received up to 24 hours late will not be graded. The project will be graded out of a total 100 points.

2 Submission Instructions

1. Submit (i) report in pdf format, (ii) all code.
2. Comment your code appropriately.
3. Make sure that the submitted code is running without error. Add a `README` file if required.

[1] [1] Nilsback, M.E. and Zisserman, A., 2008, December. Automated flower classification over a large number of classes. In 2008 Sixth Indian Conference on Computer Vision, Graphics Image Processing (pp. 722-729). IEEE.

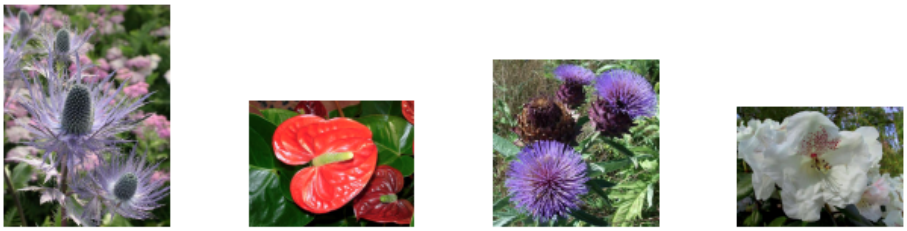

Figure 1: Example of training images for the task of flower classification.

4. Participate in the in-class Kaggle Challenge. Report your team name and performance. The submission deadline for the Kaggle challenge is the **7th April 2020, 11:59 PM**.

5. The report should be comprehensive but concise. It should not exceed 10 pages.

6. If external libraries were used in your code, please specify them by name and version in the `requirement.txt` file.

7. Submissions that do not follow the format will be penalized 10%.

3 Classification

The classification training dataset contains 6000 images with 102 categories (Fig. 1). A csv file with training image IDs and its corresponding classification labels is given along with this project submission. A separate testing dataset with 2189 images is hosted on in-class Kaggle Challenge.

The goal is to classify the images into the specified categories. To this end, your team is required to extract features from each image. You are free to use any of the computer vision features discussed in class or others you have found online (e.g. SIFT, SURF, HoG, Bag of Words, etc.). Keep in mind that multiple feature extractors can be combined. That being said, the use of Deep Learning, including CNNs, is **strictly prohibited for this task**. Once features are extracted, train a Support Vector Machine (SVM) classifier using your computed features and the ground truth labels. Optimize the hyperparameters (e.g., number of features, thresholds, SVM kernel, etc.) for the feature extraction stage and for the SVM classifier. Repeat using a different (non-deep learning) classifier of your choice.

3.1 Classifier evaluation

Evaluate your classifiers using K-fold cross-validation* (with K=5) on the training set you downloaded. If you aren't familiar with cross-validation see Section

6.2. Report the following metrics, as appropriate for your method of cross-validation:

1. Average classification accuracy* across validations, with the standard deviation. **(5/25 points)**
2. Average precision* and recall* across validations. Are these values consistent with the accuracy? Are they more representative of the dataset? In what situations would you expect precision and recall to be a better reflection of model performance than accuracy? **(5/25 points)**
3. A confusion matrix * on a validation set. Plot the matrix as an image. Are any of the classes difficult for your classifier? Discuss. **(5/25 points)**
4. Create your team on Kaggle in-class challenge. Report your team-name and performance of your team on the test dataset. Do you observe same performance as your training cross-validation performance? If not, what could be the reason for this? **(10/25 points)**

3.2 Classification Grading

For the report, describe your approach to the problem and elaborate on the design decisions. For example, discuss which features were extracted, how were your hyperparameters selected, etc. Include the metrics listed in Section 3.1. How was the data divided, and how did you perform cross-validation? Discuss the methods in detail; your goal is to convince the reader that your approach is performing the way you claim it does and that it will generalize to similar data. The grading for the submission is divided as follows:

1. Description of the contents of the dataset (image size across dataset, number of samples for each label, etc.). **(5 points)**
2. Explanation of feature extraction method including any pre-processing done on the images (ex. resizing). **(5 points)**
3. Explanation of how the feature extraction parameters were selected. **(5 points)**
4. Description of cross-validation method. **(5 points)**
5. Evaluation of performance and interpretation of results (from Section 3.1). **(25 points)**
6. Display 25 example images with their groundtruth class labels and predicted classs from your training images in the report. **(5 points)**
7. Inclusion of well-documented code (separate from report). **(8 points)**
8. **(10 points)** Your team will get points based on their ranking on the Kaggle Challenge. 1st ranked team will get 10 points, 2nd ranked team will get 9 points, 3rd ranked team will get 8 points and so on. Any team with ranking lower than 10 will get 1 point.

Figure 2: Example of (left) flower image and its corresponding (right) segmentation.

4 Segmentation

The second part of the project is to segment the foreground flowers from the background. The segmentation dataset provided contains 100 images. For each image there is a corresponding binary ground truth set of labels (Fig. 2). Your task is to generate a binary mask for the given flower images.

For this task, your team can make use of any of segmentation algorithms covered in-class (Ex. GMM [2], EM, K-means, Normalized graph-cut, Mean-Shift segmentation, etc.). The use of Deep Learning algorithms including CNNs is **strictly prohibited for this task**. Use **at least two** different segmentation methods. Optimize the hyperparameters of your segmentation algorithms.

4.1 Segmentation evaluation

Evaluate your segmentation algorithm using cross-validation* on the training set you downloaded with k=5 (and see Section 5.2). Use this to optimize hyperparameters for your segmentation algorithm. Report the following metrics, as appropriate for your method of cross-validation:

1. Compute the average DICE coefficient for the predicted vs. true segmentation mask across validations, with the standard deviation. **(5/10 points)**
2. Report the distribution of DICE coefficients over your validation sets. **(5/10 points)**

4.2 Segmentation Grading

For the report, describe your approach to the problem. Describe the method from the input images to the binary labels. Include the metrics listed in Section 5.1, and interpret the results. The grading for the report is divided as follows:

[2]Note that you can use GMM in a supervised manner. Refer to this link

1. Description of segmentation methods along with how the hyperparameters were selected. **(5 points)**
2. Description of cross-validation method. **(5 points)**
3. Evaluation of performance and interpretation of results (from Section 4.1). **(10 points)**
4. Display 5 examples training images and its corresponding predicted and groundtruth binary segmentation masks. **(5 points)** (Ex. Fig. 2)
5. Inclusion of well-documented code (separate from report). **(7 points)**

5 Bonus Points

1. Top team for Classification in-class Kaggle Challenge **(5 points)**
2. In addition to the non-deep learning framework developed, a bonus of **10 points** will be given for deep learning implementations which complete the classification task. The details of the implementation are left to the groups, but the goals are the same as in Sections 3. The submission should include following:
   (a) Schematic of architecture **(2 point)**
   (b) Description of training **(2 points)**
   (c) Evaluation of performance (as described in the relevant task's section) **(2 point)**
   (d) Comparison with the methods from Sections 3 **(2 point)**
   (e) Code with a description of the environment. **(2 points)**

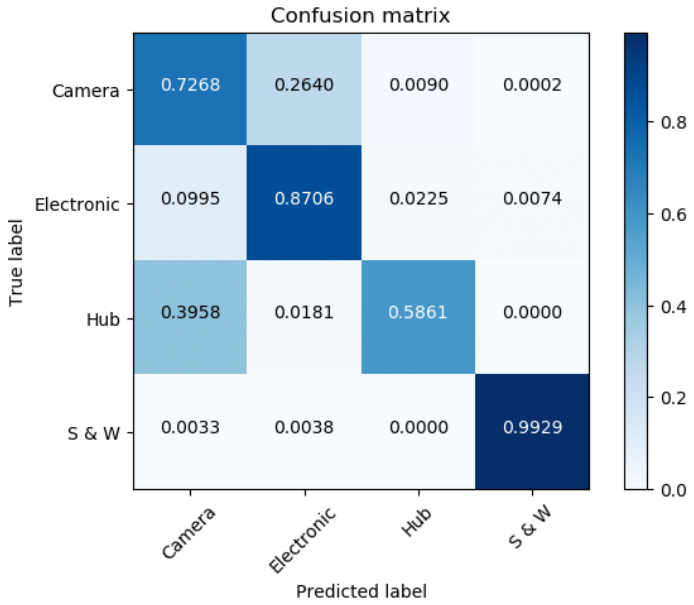

Figure 3: Confusion matrix for a 4-class labelling task. Entries along the diagonal are correctly classified. Off-diagonal terms indicate that the classifier is confusing one class for another.

6 Appendix

6.1 Definitions

1. **Accuracy:** The number of correct predictions divided by the total predictions. $\frac{TP+TN}{TP+TN+FN+FP}$

2. **Confusion matrix:** A visual representation of mis-classification. A 2D histogram of true vs. predicted labels. See Fig. 3.

3. **Cross-Validation:** Method of evaluating how well a model will generalize; evaluates how sensitive a model is to the training data. See Section 6.2.

4. **DICE Coefficent:** Intersection over union for predicted and true labels; $\frac{2TP}{2TP+FN+FP}$. See Fig. 4.

5. **Precision:** True positives divided by true positives and false positives: $\frac{TP}{TP+FP}$. "Of the predictions made for class C, what fraction was correct?"

6. **Recall:** True positives divided by true positives and false negatives; $\frac{TP}{TP+FN}$ "Of the samples for class C, how many were correctly predicted?"

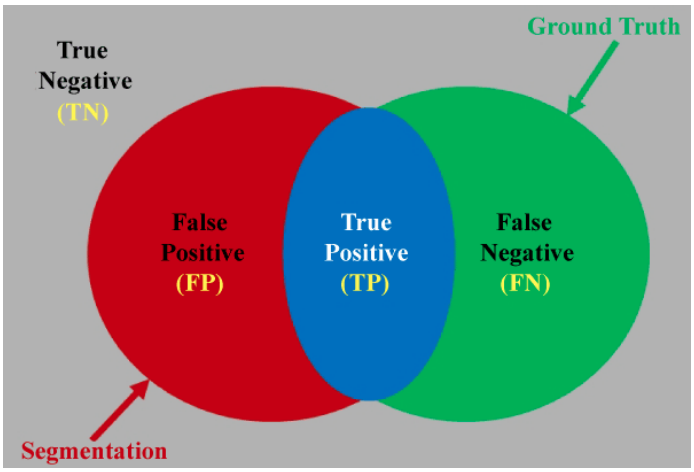

Figure 4: Schematic illustration of the measuring the segmentation errors for calculating the Dice similarity coefficient (DSC).

6.2 Cross-Validation

Cross-validation is the partitioning of the available dataset into two sets called training set and validation set. A model is trained on the training set and evaluated on validation set. The process is repeated on several, different partitions of the data, and the model's performance across the partitions indicate the model's ability to generalize to new datasets.

A widely used method for cross-validation is k-fold cross-validation. First, the available dataset is randomly partitioned into k equal subsets. One subset is selected for validation, and the remaining k-1 subsets are used to train the model. Each subset serves as validation set exactly once. The performance on the k validation subsets form a distribution which is used to evaluate the model's ability to generalize.