ECSE 415 - Intro to Computer Vision - Project

Due: November 29th, 2018

1 Submission

For this project, you will form groups of 4-5 people. Submissions should be submitted electronically via myCourses. All submitted material is expected to be the students' own work or appropriately cited (academic integrity guidelines can be found here). Software packages (e.g. scikit-learn, MATLAB toolboxes) which implement machine learning-related algorithms (SVM, k-fold cross-validation) are allowed, along with bounding box drawing (but not generation). Appropriate citations are expected; 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**.

In this project you will address two problems: localization and classification. You will use the MIO-TCD [1] car dataset for this purpose. The dataset can be downloaded from the challenge website. Each problem is divided into sections to help you explore the computer vision task. The classification task will have you extract features to classify the type of car within an image. For the localization task, you will pick out objects of interest from a large image to classify them.

Terms with asterisks* are defined in Section 5.

2 Classification

The classification dataset contains over 600k images with 11 categories. The goal of this section is to classify the images into these categories. You will extract features from each image using a computer vision algorithm of your choice. Keep in mind that multiple feature extractors can be combined. 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) for your feature extraction and the SVM. Repeat using a (non-deep learning) classifier of your choice.

2.1 Classifier evaluation

Evaluate your classifiers using cross-validation* on the training set you downloaded. If you aren't familiar with cross-validation, use k-fold cross-validation with k=10 (and see Section 5.2). Report the following metrics, as appropriate for your method of cross-validation:

- Average classification accuracy across validations, with the standard deviation. **5/15 points**
- 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? **5/15 points**
- A *confusion matrix** on a validation set. Plot the matrix as an image. Are any of the classes difficult for your classifier? Discuss. **5/15 points**

2.2 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 2.1. 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:

- Description of the contents of the dataset (number of samples and image size, for each label). **5 points**
- Explanation of feature extraction method. **5 points**
- Explanation of how the feature extraction parameters were selected. **5 points**
- Description of cross-validation method. **10 points**
- Evaluation of performance and interpretation of results (from Section 2.1). **15 points**
- Inclusion of well-documented code (separate from report). **10 points**

3 Localization & Classification

The classification dataset was created using image crops of the localization set. The second part of the project is to generate bounding boxes for the foreground objects of interest; namely, for objects belonging to the localization classes. The bounds are then used to crop the large image and are passed to a classifier. The implementation of the localizer is left up to you. You may use the classifier you trained in the previous section. Note that the class labels for localization are different from the ones for classification; some labels will require pre-processing (i.e., more design decisions that you need to report).

3.1 Localizer evaluation

Evaluate your localizer as follows:

- Compute the DICE coefficient for the predicted vs. true bounding boxes (for multiple boxes in one image, match the boxes to maximize the mean DICE) **5/25 points**
- Report the distribution of DICE coefficients over your validation sets. **10/25 points**
- Use the localization predicted by your localizer to evaluate your classifier from Section 2. Compare the accuracy, prediction, and recall of using your localizer + classifier vs. using the classification data + classifier. Is there a difference, and why/why not? Should the 'background' label of the classifier be included when evaluating the performance of the localizer, and why/why not? **10/25 points**

3.2 Grading

For the report, describe your approach to the problem. Describe the method from the input images to the set of output bounding boxes. Include the metrics listed in Section 3.1, and interpret the results. The grading for the report is divided as follows:

- Description of the contents of the dataset (e.g., number of samples and bounding box size for each label, contents, etc.) **5 points**
- Description of localization method. **10 points**
- Description of your cross-validation approach. **5 points**
- Evaluation of localization performance and interpretation of results. (Section 3.1). **25 points**
- Inclusion of well-documented code (separate from report). **5 points**

4 Deep Learning Bonus

A bonus of **10 points** will be given for deep learning implementations which complete the localization or classifications tasks. The details of the implementation are left to the groups, but the goals are the same as in Sections 2 and 3. The submission should include:

- Schematic of architecture (**1 point**)
- Description of training (**2 points**)
- Evaluation of performance (as described in the relevant task's section) (**1 point**)
- Description of validation (**3 points**)
- Comparison with the methods from Sections 2 and 3 (**1 point**)
- Code with a description of the environment[1]. (**2 points**)

[1] In Python, use "pip3 freeze > requirements.txt" to generate the requirements file for the Python environment

5 Appendix

5.1 Definitions

Accuracy	The number of correct predictions divided by the total predictions. $\frac{TP+TN}{TP+TN+FP+FN}$
Confusion matrix	A visual representation of misclassification. A 2D histogram of true vs. predicted labels. See Figure 1.
Cross-validation	Method of evaluating how well a model will generalize; evaluates how sensitive a model is to the training data. See Section 5.2.
DICE Coefficent	Intersection over union for predicted and true labels; $\frac{2TP}{2TP+FN+FP}$. See Figure 2.
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?"
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?"

5.2 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.

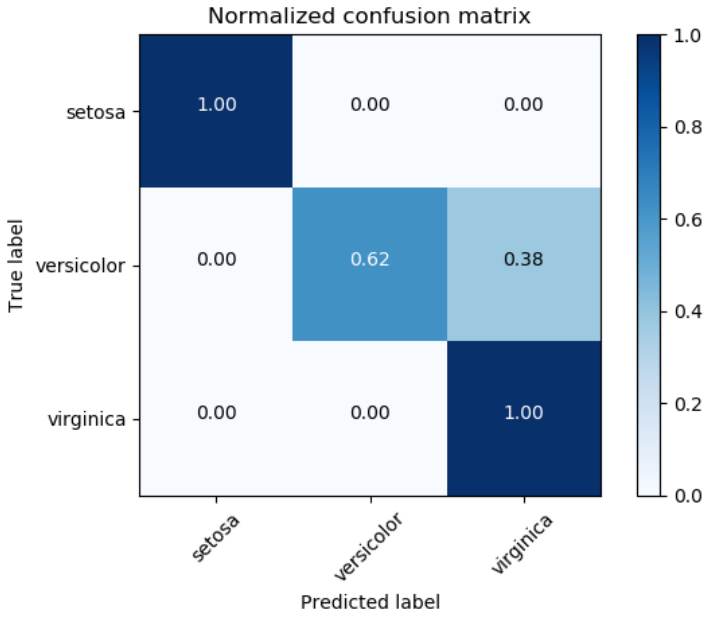

Figure 1: Confusion matrix for a 3-class labelling task, from [2]. Entries along the diagonal are correctly classified. Off-diagonal terms indicate that the classifier is confusing one class for another.

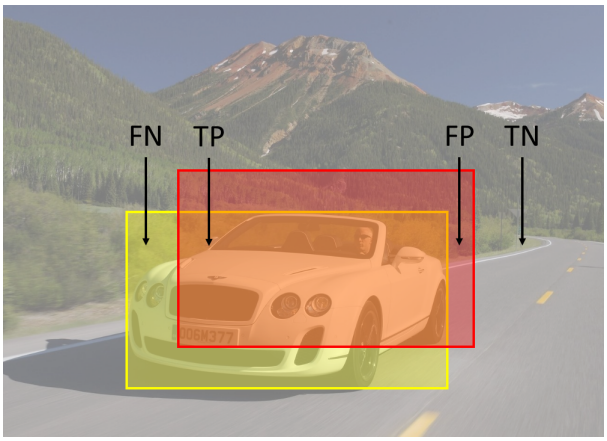

Figure 2: An image of a car taken from [3] and its associated bounding boxes. Let the yellow box contain the true labels for the ground truth, and let the red box contain the positive labels for the predictions. Where they overlap is the True Positive (TP). The positive predictions outside the ground truth are the False Positives (FP). The true ground truth labels outside the prediction box are the False Negatives (FN). The remaining labels represent the True Negative (TN). The DICE coefficient is the size of the overlap from each set (2*TN) normalized by the total area ((FN+TP) + (TP+FP)).

References

[1] Z. Luo, F. Branchaud-Charron, C. Lemaire, J. Konrad, S. Li, A. Mishra, A. Achkar, J. Eichel, and P. Jodoin, "Mio-tcd: A new benchmark dataset for vehicle classification and localization," *IEEE Transactions on Image Processing*, vol. 27, no. 10, pp. 5129–5141, Oct 2018.

[2] "Confusion matrix," http://scikit-learn.org/stable/auto_examples/model_selection/plot_confusion_matrix.html, accessed: 2018-10-30.

[3] J. Krause, M. Stark, J. Deng, and L. Fei-Fei, "3d object representations for fine-grained categorization," in *4th International IEEE Workshop on 3D Representation and Recognition (3dRR-13)*, Sydney, Australia, 2013.