ECSE-415 Introduction to Computer Vision
Final Project: Detecting, Localizing, and Tracking People

Due: April 11, 2022, 11:59PM

1 Introduction

The goal of this project is to implement a complete pipeline to detect, localize, and track vehicles. The pipeline will consist of a number of steps: (1) a dataset setup (10 points) (2) a classifier (35 points), (3) a detector and localizer (25 points) and (4) an object tracker (30 points). A partial training and validation set sourced from the KITTI vision benchmark [1] is provided.

For this project, you will form a team of 4-5 people. The objective of the project is to permit your team to explore a sequence of concepts described in class. The classification task will require your team to extract features in order to discriminate between 2 classes: vehicle vs. non-vehicle. This classifier can be used to localize all instances of vehicle in an image. Instances of vehicles are then tracked from frame to frame in a video sequence.

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.

4. The report should be comprehensive but concise. It should not exceed 10 pages excluding references.

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

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

(a) Raw Frame

(b) Labeled Frame

(c) Extracted patch

Figure 1: Example: Detection and localization pipeline

3 Dataset (10 Points)

The training dataset (link) provided contains three training video sequences (A) 0000, (B) 0001, and (C) 0002. All videos are provided at fixed resolution of 1242x375. Bounding box annotations are provided for the above mentioned three difference video sequences. Students are required to parse a ground truth *0000.txt*, *0001.txt*, and *0002.txt* files associated with each video sequence in order to extract bounding box information for a specific sequence on a frame by frame basis. In order to accelerate label extraction, a colab notebook is provided. The *.txt* files contain frame number, bounding box id, bounding box coordinates, some other additional values which should be ignored.

Teams may expand their dataset as they see fit. Additional video sequences are made available with bounding box annotations. Additional external data may be incorporated at the teams discretion.

Build a classification dataset by extracting patches from the datasets. Extract vehicle information using bounding box annotations and select patches on non-vehicle with minimal intersection with vehicle bounding boxes. Report class distributions (e.g. how many vehicle vs. non-vehicle are in the images) as well as dataset statistics (e.g. aspect ratios, image sizes). **(10 points)**

Colab Resource: https://colab.research.google.com/drive/16L8jkVTCnapbY1DiPKAzmabnf3cvo984?usp=sharing

4 Classification (35 Points)

The final goal of the classifier will be to classify vehicle and non-vehicle in validation and test images. The system will first learn to differentiate these two classes during training/validation in a 3-fold validation **(see K fold validation section 8.2)**. To this end, your team is required to extract features from within 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.

4.1 Classifier Evaluation

Train and evaluate your classifiers using K-fold cross-validation (with K=3)as explained in Section 8.2. Report all metrics as specified below:

1. Average classification accuracy [1] across validations, with the standard deviation. **(5 points)**

2. Average precision [4] and recall [5] 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 points)**

4.2 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 4.1. 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. Explanation of feature extraction method including any pre-processing done on the images (ex. resizing). **(5 points)**

2. Explanation of how the feature extraction parameters were selected. **(5 points)**

3. Evaluation of performance and interpretation of results for SVM classifier. (from Section 4.1). **(10 points)**

4. Evaluation of performance and interpretation of results for a different (non-deep learning) classifier of your choice (from Section 4.1). **(10 points)**

5. Display 5 examples per class with their ground truth class labels and predicted class label from your training images in the report. **(5 points)**

5 Detection and Localization (30 Points)

The classification dataset was created using image patches of the localization set. The goal for this section of the project is to detect and generate bounding boxes for all instances of foreground objects of interest, in this case the vehicles in a frame. This is accomplished through the use of a localizer. Implement a non-deep learning localizer. You can use sliding window based approach for localizing vehicle by using the classifier trained in the previous section. You should apply non-maximum supression for removing redundant bounding boxes.

5.1 Detector and Localizer Evaluation

Evaluate your detector and localizer on 3-fold validation (same split as the classification section). Compute the IoU [3] for the predicted vs. true bounding boxes (for multiple boxes in one image, match the boxes to maximize the mean IoU). Report the distribution of IoU coefficients over your validation sets.**(5 points)**

5.2 Detection and Localization Grading

Describe your approach to the problem and the method from the input images to the set of output bounding boxes. Include the metrics listed in Section 5.1, and interpret the results. The grading for the report is divided as follows:

1. Description of the contents of the dataset (e.g., number of samples and bounding box size for each label, contents, etc.) **(5 points)**

2. Description of detection and localization method. **(10 points)**

3. One of the important part of sliding window based approach is the detection of redundant bounding boxes. There are mainly two remedies for this (1) Non-maximum suppression, (2) merging of bounding boxes.

    - Write your own code for non-maximum suppression in numpy to remove redundant bounding boxes. Evaluate the detection and localization performance (Section 5.1). **(5 points)**
    - Use openCV function groupRectangle (reference) to group bounding boxes.Evaluate the detection and localization performance (Section 5.1). **(5 points)**

4. Display ground truth and predicted bounding boxes for 5 different images from the validation set. **(5 points)**

6 Tracking (25 Points)

Once localized, it is possible to track people frame by frame. Your team will implement an object tracker to update a localization estimate using optical flow (refer Tutorial-11). Repeat the process with any other method of choice (refer Tutorial-12).

6.1 Tracker Evaluation

Select any one of the provided video sequences without labels. Localize instances of a vehicle in the initial frames of a video sequence. Track the instances for as many frames as possible or until they have left the frame. Feel free to localize new instances appearing in the frames.

Evaluate your tracker. Compute the mean IoU [3] for the predicted vs. true bounding boxes over the video sequence. Report the distribution of IoU over your validation sets. **(5 points)**

6.2 Tracking Grading

1. Description of tracking methods **(2.5 points * 2 = 5 points)**
2. Evaluation of tracking performance and interpretation of results for both type of tracking methods. (Section 5.1). **(5 points * 2 = 10 points)**
3. Run the algorithm on the provided tracking video and annotate bounding boxes for both implementations of the tracker. Submit the videos as additional files separate from the report. **(10 points)**

7 Bonus Points (10 Points)

In addition to the non-deep learning framework developed, a bonus of **10 points** will be given for deep learning implementations which complete either classification or localization tasks. The details of the implementation are left to the groups, but the goals are the same as in Sections 4.1, 5.1. The submission should include following:

1. Schematic of architecture. **(3 point)**
2. Description of training. **(2 points)**
3. Evaluation of performance (as described in the relevant task's section). **(3 point)**
4. Performance comparison to non deep learning method chosen. **(2 point)**

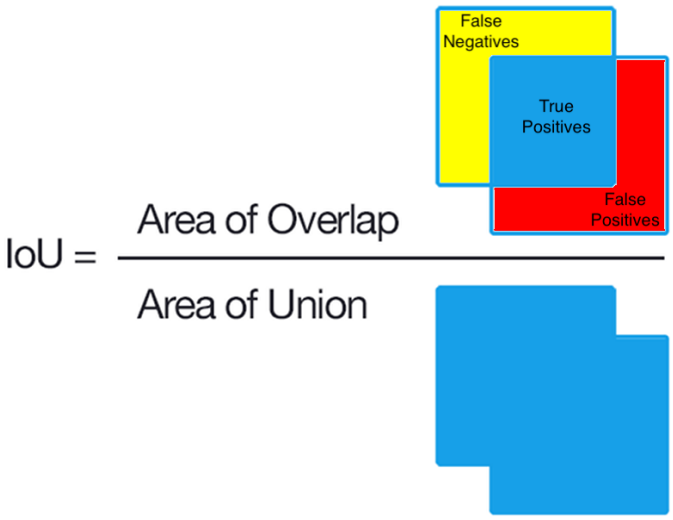

Figure 2: Schematic illustration of the measuring the intersection errors for calculating the intersection over union similarity coefficient (IoU).

8 Appendix

8.1 Definitions

1. **Accuracy:** The number of correct predictions divided by the total predictions. $\frac{TP+TN}{TP+TN+FN+FP}$
2. **Cross-Validation:** Method of evaluating how well a model will generalize; evaluates how sensitive a model is to the training data. See Section 8.2.
3. **IoU:** Intersection over union for predicted and true labels; $\frac{TP}{TP+FN+FP}$. See Fig. 2.
4. **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?"
5. **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?"

8.2 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.

In our case, we will use 3-fold validation method as the total number of provided sequences are three (0000, 0001, 0002). You will divide the whole dataset into following three different partitions for training and validation set.

- (Training Sequences) - (Validation Sequence): (0000, 0001) - (0002)
- (Training Sequences) - (Validation Sequence): (0000, 0002) - (0001)
- (Training Sequences) - (Validation Sequence): (0002, 0001) - (0000)