BIRZEIT UNIVERSITY

Electrical and Computer Engineering Department

ENCS5396 – Machine Learning

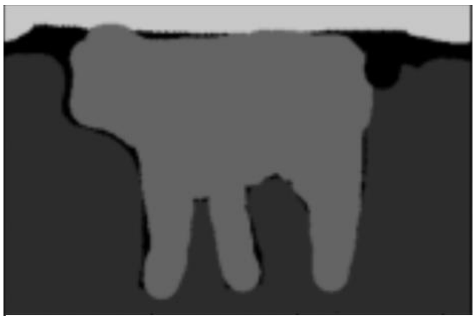
Project – ML applied on image segmentation

Second Semester - 2022

Submission Deadline: June 13, 2022
Team Work: Teams of two students.
Programming Language: You can use any programming language and any ML library. As a recommendation, implement your project in Python and use scikit-learn to test different ML models.

In this project, we will test machine learning models for image segmentation task. The problem of image segmentation can be formulated as a classification task, where every pixel in an image is mapped to an already known class C based on the information extracted from its N x N neighbors. The following figures illustrate an input example for the image segmentation task and its corresponding segmentation labels.

Input image


Ground-truth segmentation map

A set of images and their ground-truth segmentation maps is provided to train and evaluate your models. An image is in principle a three-dimensional array of size H x W x 3, where H is the image height, W is the image width, and 3 corresponds to the three RGB channels. Each location in the 2d grid is called a pixel and each of its RGB channels has a value between 0 and 255 that represent the intensity of the color. For example, a pixel with value [255, 0, 0] corresponds to a red pixel, whereas a grey pixel will have a value [127, 127, 127].
For our application, each pixel in an image is labeled with a unique class C in [Grass = 3, Water = 2, Cow = 1, Others = 0] as specified in its segmentation map. Separate *train_images.txt* and *test_images.txt* files are provided to specify training and testing images respectively.
For all tested model, the input will be an image patch of size 17 x 17 and the target will be the class label for the center pixel of the patch.

You need to complete the following tasks:

1. Training Data Generation

Create a balanced training set by randomly sampling image patches of size 17 x 17 from the training images and assigning the class label of the center pixel as a target for the image patch. Sample at

least 5000 patches for each class (i.e., you will have a training set that contains at least 20000 examples.)
Create another balanced validation set by sampling around 800 image patches per class from the training images. Note that the validation set and the training set are disjoint.
For the test set, we will use all the image patches in the testing images.

2. Features extraction

We will represent each image patch with a 192-dimensional features vector that corresponds to the intensity histograms for each of the RGB channels in the patch.
For each of the RGB channels, create a histogram with 64 bins, where bin *i* contains the number of pixels with intensity value in the range [*i**4, (*i*+1)*4), and $0 \leq i < 64$.
Normalize the histogram of each channel by dividing by the number of pixels in the patch.
The final features vector is obtained by concatenating the normalized histogram for all the channels.

3. Baseline Model

As a baseline model, evaluate a nearest neighbor baseline (with k=1). Report the pixel-wise accuracy on the test images.

4. Improvements

Try to achieve better performance by evaluating two additional models on the task. Discuss and motivate your model selection, and comment on why the performance has improved (or potentially did not improve). If the chosen models have hyper-parameters, make sure to tune at least one hyper-parameter for each model using the validation set. To tune a parameter, you need to test at least three different values.

For each model report the pixel-wise accuracy on the training, validation, and testing sets.

Visualize the predictions for both models on the test images (same as the figure above). Examine the visualized predictions and check the wrongly predicted labels. Discuss any interesting observation you notice on the prediction errors.

5. Report

Besides your code, you need to submit a report (max. 8 pages) to analyze and discuss your results. Your project will be mainly evaluated based on the quality of the report (60% of the project grade). The report should have the following sections:

- Introduction: Describe the task you are addressing in the project and the different models you have tried. Also, define the evaluation metric that you have used.
- Technical Details: for each model you have tried, list the values of all the hyper-parameters you have used and explain the meaning of each one.
- Experiments and Results: discuss all the experiments you have done (model evaluation, hyper-parameters selection, …. etc). Include both quantitative results (pixel-wise accuracy) and visualized predictions on the test images. Comment on all the results.
- Conclusions and Discussions: discuss your conclusions and also comment on the limitations of the used models and the evaluation metric.