GEO5017 A2
Urban Objects Classification from AirBorne LiDAR Data

1 Introduction

We live in a 3 Dimensional (3D) world that is composed of urban entities, such as buildings, lampposts, mailboxes, and vegetation. During the last decades, there has been an ever-increasing demand, both by academia and industry, for analyzing and interpreting 3D urban geoinformation. A variety of research works have been devoted to the problem of 3D scene understanding. Successful interpretation of urban scenes allows for various modern applications such as urban modeling, autonomous driving, and environment monitoring.

Understanding the urban environment from 3D airborne LiDAR data has been extensively studied in recent years. LiDAR point clouds are vastly available in many countries. Additionally, compared to 2D imagery, point clouds provide more accurate 3D measurements with higher spatial resolution. Classification of point clouds is a fundamental task in 3D scene understanding, which aims to assign each point cloud a meaningful semantic label (e.g., building, tree, car). Semantic segmentation is a more complex task, assigning per point a label. In other words, segmentation can be viewed as dense classification on the point level.

In this assignment, instead of solving the semantic segmentation problem, we will focus on classification only. You will be working on grouping a set of pre-segmented individual urban objects into groups using supervised learning techniques (i.e., SVM, random forest).

2 The tasks

2.1 Feature engineering

You will need to design a set (≥ 6) of features for the task of classification. There are no standard features you can directly use. It could be helpful to visualize and observe the point clouds of different types of objects to get some intuition. It can also be helpful to select good features by looking into the distributions of potential features. Be creative! Any quantities that can separate different types of objects can be good features and are worth looking into. You are also welcome to refer to the features introduced in previous works [1, 2, 3][1].

[1] Pay attention: many features in these works cannot be directly used in this assignment because they were designed for semantic segmentation, which is different from classification

2.2 Classification

Carry out point cloud classification using the two classifiers introduced in the course, i.e., SVM and RF. In this assignment, we specifically ask you to use the implementation of the classifiers from Scikit-Learn. For the SVM classifier, please try different Kernel functions, select the most promising one (and justify your choice in the report). For both classifiers (i.e., SVM, RF), try different combinations of hyper-parameters and find the best model with the highest performance.

Randomly split the dataset into a training set and a test set (i.e., 6:4). Then for each classifier, train, test, and evaluate the result. Finally, compare the performance of the two classifiers.

For the evaluation, use the metrics described in Section 3.1.

3 Evaluation and analysis

3.1 Evaluation metrics

To evaluate the performance of a classifier over the test set, the following metrics are commonly used: overall accuracy (OA), mean per-class accuracy (mA), and confusion matrix.

- **Overall accuracy**

$$OA = \frac{1}{N}\sum_{i=1}^{C} n_i, \tag{1}$$

  where C is the total number of classes. n_i is the number of objects correctly classified in class i. N is the number of objects in total. OA measures the overall performance of the classifier.

- **Mean per-class accuracy**

$$mA = \frac{1}{C}\sum_{i=1}^{C} \frac{n_i}{N_i}, \tag{2}$$

  where N_i is the total number of objects in the i-th class. mA measures the per-category classification accuracy. It helps to relieve the class imbalance issue by averaging the accuracy of all classes.

- **Confusion matrix**. It has a specific table layout that allows the visualization of the performance of a classifier. Each row of the matrix represents the instances in an actual class while each column represents the instances in a predicted class[2]. You can compute the confusion matrix by yourself or use Scikit-Learn Metrics.

3.2 Analysis

Your report should also include the following analysis:

- Which features are more distinctive, and which are less distinctive? Use the metric(s) introduced in our lecture to quantitatively analyze the effectiveness of your proposed features, and select the 4 most effective ones.

[2]Confusion matrix. https://en.wikipedia.org/wiki/Confusion_matrix

- From each chosen model of the classifiers (i.e., SVM, RF), tune its train-test split ratio (i.e., gradually enlarge the training set while decreasing the test set), compute OA for each split (use cross validation) and make a plot of OA with respect to the number of samples of the training set. The plotted curve is also known as the "learning curve" in machine learning. How does the curve behave? Provide your insights on the learning curve.
- How do different configurations (i.e., hyperparameters) of a model affect its performance? Based on this analysis, explicitly make a recommendation of your final model for each classifier.
- Provide the error analysis of your final model for both classifiers. Use a confusion matrix to demonstrate your analysis. In what categories does your model do a good job? In what categories does your model make the most prediction mistakes? Why are these happening?

4 Dataset

We will use 500 point clouds of urban objects (The dataset is distributed with this assignment). For each point cloud, its ground truth label is encoded as follows (based on the base names of the point cloud files):

1) 000 - 099: building;
2) 100 - 199: car;
3) 200 - 299: fence;
4) 300 - 399: pole;
5) 400 - 499: tree.

5 Submission (Due: March. 25th)

Please compress all the following into a single archive titled **GEO5017_A2_Group_XX.zip** (where 'XX' is your group ID and can be found here) and submit it to BrightSpace:

- **A report (max 3 pages excluding figures and tables)**
  - **Introduction**
    Describe the motivation and goal of this assignment. (5%)
  - **Methodology**
    - Describe and define your initial features and explain how you select the 4 most effective features. (10%)
    - Describe your model configuration (i.e., hyperparameters) for SVM and RF. (10%)
  - **Experiments and evaluation**
    - Describe your experiments and classification results. Analyze the effect of model configuration (i.e., hyperparameters) (15%)

    - Provide error analysis of the model of both classifiers. (10%)
    - Visualize the learning curves of both classifiers and provide your analysis of the curve. (10%)
    - You must implement your own functions for grid search and learning curve visualization. You are recommended to use Matplotlib to plot the curves. 20% deduction (10% for hyperparameter tuning and 10% for learning curve visualization) applies if you simply visualize the learning curves using any existing library (e.g., Scikit-Learn).
  - **Conclusion**
    - What conclusion can we draw from this assignment? (5%)
    - What could be done to achieve a better result?(5%)
  - **A short description of who did what**.

- **Source code**
  - The source code, archived in a 'code' subfolder. The code should build, run, and reproduce your results without changes. (30%)
    - For Python code, **only** '*.py' files are accepted. If you use Jupyter Notebook for development, make sure you submit '*.py' files ('*.ipynb' files will not be accepted). For C++ code, **only** '*.cpp' files are accepted.
    - If you have multiple source code files that should run in a specific order, **do** name them in a way such that they are well ordered (e.g., '1_features.py', '2_train.py', '3_test.py', '4_visualization.py'). If no such information is provided, we will **expect a 'main.py' file** as the only entry point.
    - Provide a 'ReadMe.txt' file to briefly explain how to run the code and reproduce the results, e.g., dependence on external libraries/packages, the path to data, and where to find the results in case you save results or figures into files.
    - Please **do NOT include data** in your submission.
  - [*optional*] Provide a link to the GitHub repository (if you use GitHub) in the 'Experiment' section of your report. You are encouraged to collaborate with your teammates on GitHub.

Note: The report should be as concise as possible (within 3 pages excluding figures, tables, and references), but it should provide sufficient information to re-implement your methods and reproduce your results. Try to use the mathematical language (i.e., equations) as much as possible. You will have to omit less important details and discussions. Irrelevant descriptions and discussion may lead to the deduction of points.

References

[1] Chao-Hung Lin, Jyun-Yuan Chen, Po-Lin Su, and Chung-Hao Chen. Eigen-feature analysis of weighted covariance matrices for lidar point cloud classification. *ISPRS Journal of Photogrammetry and Remote Sensing*, 94:70–79, 2014.

[2] Hugues Thomas, François Goulette, Jean-Emmanuel Deschaud, Beatriz Marcotegui, and Yann LeGall. Semantic classification of 3d point clouds with multiscale spherical neighborhoods. In *2018 International conference on 3D vision (3DV)*, pages 390–398. IEEE, 2018.

[3] Weixiao Gao, Liangliang Nan, Bas Boom, and Hugo Ledoux. Sum: A benchmark dataset of semantic urban meshes. *ISPRS Journal of Photogrammetry and Remote Sensing*, 179:108–120, 2021.