SFU

Practices in Visual Computing II (Spring 2025)

Assignment 4

3D Point Cloud Classification

Total Marks: 90

Due: Friday, 4th April, 23:55 PM

Overview

In this assignment, you will be working on 3D Point Cloud Classification. Unlike previous assignments, *no template code will be provided—you will need to implement everything from scratch on your own.* However, we have included a few helpful Python functions to assist you. Refer to the following sections for more details.

1 Dataloader (20 marks)

1. Start with the ModelNet-10 (download link) dataset, which consists of 3D meshes from 10 different categories, split into *train* and *test* sets.

2. Sample a point cloud from each mesh and use it for training/testing. The number of points to sample is a hyper-parameter, you can start with 2048 points but adjust if needed. You can use libraries like `trimesh` or `open3d` for this.
   Submit: Images of 5 meshes with their sampled point clouds at two different densities, displayed side-by-side and clearly visible.

3. Apply at least five augmentations to the point cloud before passing it to the model. Implement these augmentations from scratch (don't use any 3D libraries).

Submit: For each augmentation submit an image of the mesh, originally sampled point cloud, and two variations generated using that augmentation. In the end, include one final example with all augmentations applied at once.

4. Remember to create a validation split to ensure proper model evaluation!

5. Implement a function to test the dataloader and verify its correctness.

2 Model Design (20 marks)

1. Design your own model for point cloud classification (see PointNet for inspiration).
   Submit: An image of your model architecture (similar to Fig. 3 and 4 in Assignment 1 or Fig. 1 and 3 in Assignment 2). It doesn't need to be highly aesthetic but should clearly convey all the necessary details.
   Submit: A short note explaining what makes your model unique and how it is differs from PointNet.

2. Implement a function to test your model using dummy data. This is helpful while designing your model.

Note: You should not directly copy the architecture from prior works, but you are encouraged to take inspiration from them.
Submit: List all the works (papers, blogs, etc.) that influenced your model design.

3 Training and Testing (20 marks)

1. Implement the code to train and evaluate your model using the sampled point cloud data.

4 Code Submission (10 marks)

Your code should be well-written, commented, and organized. Code presentation is just as important as functionality—if your code is difficult to read or understand, nobody (including yourself) will use it in the future. Follow these guidelines for submission:

1. `README.md`: Specify the necessary details of your project, how is the code organized, how to install the requirements, how to run the code, etc. It should be a markdown file with proper formatting.
2. `requirements.txt`: The python libraries (with their version, if required) to be installed to run the code.
3. `model.py`: Contains the model definition.
4. `dataloader.py`: Implements the data loader.
5. `run.sh`: A shell script to execute the code. It should have commands for both, training followed by testing, and just testing with a pre-trained model.
6. `main.py`: The main driver file.
7. `utils.py`: A utility file for helper functions.
8. Treat hyperparameters (e.g., learning rate, number of epochs, batch size) as user inputs—use `argparse` to handle them.
9. Avoid hard-coded values—define variables and use them consistently.
10. Do not submit the pre-trained model. Instead, upload it to a cloud platform and provide a downloadable link in `README.md`.

Note: These are the minimum requirements—feel free to *improve* upon them!

5 Additional Questions (20 marks)

You will be submitting a report along with the code for this assignment. Answer the following questions in your report:

1. **Accuracy Metrics** – Report both *overall accuracy* (you should achieve at least 85% here) and *per-class accuracy*.
2. **Visualization** – Include images of 10 test meshes (one from each category), their sampled point clouds, and the top three predicted categories with probability scores.

3. **Model Strengths** – Identify the examples/classes where your model performed well and explain why. Include a few example visualizations as described above.

4. **Model Weaknesses** – Similarly, analyze the examples/classes where your model did not perform well and explain why. Provide corresponding visualizations.

Note: Ensure that all required materials from previous sections are included in your submission.