Earth Observation Land Cover Pipeline: Project Report

Final Submission-

By Akhilesh Y Ragate

1. Introduction

This report documents the development and implementation of an automated land cover classification pipeline using Sentinel-2 imagery and ESA WorldCover data over the Delhi-NCR region. The workflow enables reproducible extraction, labeling, splitting, and modeling for robust environmental analysis.

2. Data Preparation

2.1. Spatial Grid Creation

- 60×60 km regular grid generated across the Delhi-NCR boundary using GeoPandas and shapely.
- Each satellite image is mapped by its center coordinates to the appropriate grid cell.
- Grid exported as `delhi_ncr_grid.geojson` for downstream reference.

2.2. Image Metadata Extraction

- Image filenames parsed to obtain center latitude and longitude, producing `image_coords.csv`.
- Spatial join ensures only images falling within the grid are retained, resulting in `imgs_within_grid.csv`.

3. Land Cover Patch Extraction and Label Assignment

- For every valid image:
    - Extracted a 128×128 pixel patch from the ESA WorldCover raster.
    - Assigned the most frequent (mode) land cover class within the patch as the label, mapping ESA codes to readable class names.
    - Applied quality control: Only patches with ≥50% valid (non-NoData) pixels were included.
- The final, clean dataset was saved as `labelled_images_clean.csv`.

4. Dataset Splitting and Class Analysis

- Stratified train-test split: 60% training, 40% test, preserving land cover proportions.
- Classes with only one example were excluded from splitting to ensure stratification worked correctly.
- Generated split files: `train_split.csv` and `test_split.csv`.

Class Distribution Example

Class Name	**Training Examples**
Cropland	5,100
Built-up	2,000
Other	...

- Visualization of class balance highlighted Cropland and Built-up as dominant, with minor representation of classes like Grassland and Wetland.

5. CNN Model Training and Evaluation

5.1. Model Setup

- Used a ResNet-18 CNN with ImageNet weights, final layer adjusted for the number of land cover classes.
- Training on GPUs when available.

5.2. Performance Metrics

- Macro F1 Score: Computed using both scikit-learn and torchmetrics for cross-validation.
- Confusion Matrix: Visualizes classification accuracy and characteristic confusions.Example Results: Shows both correct and incorrect model predictions for qualitative review.

6. Key Findings

- Total Valid Images: 9,216 across the study area.
- Dominant Classes: Cropland and Built-up areas; class imbalance observed, indicating opportunity for weighted or augmented learning.
- Macro F1 Score: Model demonstrates high accuracy on dominant classes, with lower performance on rare classes.
- Model Insights: Most misclassifications occur between visually-similar classes (e.g., Cropland vs. Grassland).

7. Recommendations

- For Class Imbalance: Introduce class weighting or sample augmentation to improve rare class performance.
- For Reproducibility: Maintain all preprocessing, label assignment, and split scripts with clear file and path management.
- Next Steps: Scale up to additional regions, test with images from different seasons/years, and consider alternative model architectures for further performance gains.

8. Usage Notes

- All intermediate outputs are stored in the `data/` folder for traceability.
- All scripts are modular and reusable for new AOIs or label sets.
- Visualization scripts can be adapted to provide PDF summaries of model predictions or class unions for reporting.

9. Appendix: Key Scripts and Data Files

- `01_grid_visualization.py` — Grid creation and image mapping
- `02_label_extract_assignment.py` — Land cover patch extraction, labeling, and filtering
- `03_train_test_split.py` — Stratified splitting and class analysis
- `04_cnn_train_eval.py` — CNN model training, scoring, and visualization
- `data/labelled_images_clean.csv`, `data/train_split.csv`, `data/test_split.csv` — Key processed datasets