Report of Scenario 1 of "6 - Month Intern Selection Task"

Applicant:
Chetan Singh Kaurav
BTech, 3rd Year
Department of CSE
ITM Gwalior

Important Note: *All deliverables are there in the outputs section inside the github repo, I have not included them in the report to maintain the size*

Question 1 – Geospatial Preprocessing

1. Introduction

The objective of Question 1 was to construct a complete geospatial preprocessing pipeline that prepares the study region and associated image coordinates for land-cover classification. This involved loading and reprojecting the provided study area shapefile, generating a uniform analytical grid, deriving associated spatial attributes, creating both static and interactive visualizations, and filtering image locations to retain only those falling inside the grid. These steps ensure a clean and consistent spatial foundation for the label-generation (Q2) and machine-learning (Q3) stages of the project.

---

2. Shapefile Loading and Reprojection

The Delhi–NCR study area was provided as a shapefile in the WGS84 geographic coordinate system (EPSG:4326). Since geographic coordinates do not preserve distance and area relationships, all subsequent distance-based computations required transforming the shapefile into a meter-based projected coordinate system.

The shapefile was reprojected to **UTM Zone 44N (EPSG:32644)**. This projection is appropriate for the region and enables accurate grid creation, centroid computation, and spatial filtering. After reprojection, the bounding box of the study area was obtained and used as the basis for grid construction.

---

3. Generation of a 60 × 60 km Grid

A uniform grid with cell dimensions of **60 km × 60 km** was generated to cover the entire reprojected study area. The process involved:

1. Extracting the minimum and maximum X and Y coordinates of the study area in UTM units.

2. Iteratively stepping through this bounding box in increments of 60,000 meters along both axes.
3. Creating polygons representing each grid cell.
4. Retaining only those grid polygons that intersect the study area boundary.

This resulted in a regularly spaced grid covering the entire Delhi–NCR region. The grid was saved as a GeoJSON file for further use.

---

4. Derivation of Grid Attributes

To supplement the grid cells with useful spatial metadata, two additional geometric attributes were computed:

• **Centroids:** For each grid polygon, the centroid point was calculated. These centroids help in sampling and validating grid geometry.
• **Four corner coordinates:** The minimum and maximum X/Y values of each cell were used to derive its four corners. These were saved separately to facilitate corner-based visualization and potential downstream operations.

Both centroids and corner coordinates were exported as dedicated GeoJSON layers.

---

5. Static Visualization of the Grid

A static 2D visualization was generated for verification. This plot included:

• The reprojected Delhi–NCR boundary
• All 60 km grid cells
• Each grid cell's centroid
• Four corner points for each cell

This visualization served as a sanity check to ensure that all grid cells were correctly aligned, uniformly sized, and fully covering the intended study area.

---

6. Interactive Satellite Basemap Overlay

In addition to the static visualization, an interactive HTML map was generated using **Geemap** and the Google Earth Engine basemap. This map displays:

• High-resolution satellite imagery
• The study area boundary
• The 60 × 60 km grid overlay

The interactive map allows zooming and panning to visually inspect each grid cell and its geographical context, confirming correct placement and scale.

---

7. Filtering Image Coordinates Using the Grid

The provided RGB image files contain latitude and longitude information embedded in their filenames. These coordinates were previously extracted into a CSV file with columns: filename, latitude, and longitude.

The filtering process involved:

1. Converting each image's latitude and longitude into Point geometry.
2. Reprojecting the points to **EPSG:32644** so they match the grid's coordinate system.
3. Performing a spatial join to check whether each image point lies inside any grid cell.
4. Retaining only those points that intersect at least one grid polygon.
5. Saving the filtered coordinates and producing a count before and after filtering.

Filtering Outcome

The results of the spatial join showed:

• Total images in the dataset: **9,216**
• Images inside the 60×60 km grid: **9,216**
• Images outside the grid: **0**

This indicates that the grid completely covers the region in which all image coordinates lie. Therefore, although filtering was performed correctly, none of the images were removed. This is valid and expected, as it confirms the consistency between the provided dataset and the study area.

---

8. Deliverables Generated in Question 1

The following files were produced as outputs of Q1:

• **grid_60km.geojson** – Uniform 60 km grid covering the study area
• **grid_centroids.geojson** – Centroid of each grid cell
• **grid_corners.geojson** – Four corners of each cell
• **grid_plot_with_corners.png** – Static grid visualization
• **grid_satellite_map.html** – Interactive basemap visualization
• **filtered_image_coords.csv** – Image coordinates inside the grid
• **q1_counts.txt** – Image count before and after filtering

---

9. Conclusion

Question 1 successfully established the geospatial foundation for land-cover classification. The shapefile was accurately reprojected, a uniform grid was generated with all required metadata, and both static and interactive visualizations validated the spatial accuracy of the preprocessing steps. The filtering process confirmed that the entire dataset lies within the grid, enabling a seamless transition into label extraction (Q2) and supervised learning (Q3).

Question 2 – Land Cover Label Extraction

1. Introduction

The goal of Question 2 was to generate high-quality supervised learning labels for the RGB image dataset by using the ESA WorldCover 2021 land-cover raster. Each label represents the dominant land-cover type within a fixed-size patch centered on the image coordinate. These labels form the basis for training a land-cover classification model in Question 3.
This process involved raster sampling, patch extraction, mode-based label determination, ESA-to-11-class mapping, handling uncertain cases, splitting the dataset, and analyzing class distribution characteristics.

---

2. Loading Image Coordinates and Land Cover Raster

The filtered image coordinates produced in Question 1 were loaded from image_coords.csv. Each row in this file contains:

- Filename of the RGB image
- Latitude
- Longitude

The ESA WorldCover land-cover dataset (land_cover.tif) was loaded using Rasterio. This raster contains global land-cover classes at a 100m resolution with integer-coded values representing different categories such as trees, cropland, water, built-up, shrubland, etc.

To ensure accurate geographic sampling, the image coordinates (in EPSG:4326) were transformed into the raster's coordinate reference system before extracting spatial windows.

---

3. Patch Extraction (128 × 128 Pixels)

For each image coordinate, a square land-cover patch of dimensions **128 × 128 pixels** was extracted, centered on the projected image location. This patch corresponds to an area on the ground around the location at which the RGB image was collected.

The use of a window-based approach ensures spatial consistency: each image's label is derived from the same environmental region from which the RGB image originates. RasterIO's window reading functionality was used for efficient extraction.

Boundary conditions were handled to ensure that patches near the raster edges did not cause indexing errors. If required, out-of-bounds pixels were ignored or patched with no-data values.

---

4. Dominant Class Computation (Mode-Based Labeling)

Each extracted 128×128 land-cover patch was flattened into a one-dimensional array of pixel values. The **mode** (most frequent pixel value) was taken as the label representing the dominant land-cover type.

This method is widely used because:

- It captures the most common terrain type around the image location.
- It is robust to small impurities or mixed-class pixels.
- It ensures each image receives a single, interpretable class value.

To quantify reliability, the relative frequency of the dominant class (its proportion within the patch) was computed. This "dominance ratio" was later used to handle uncertain cases.

---

5. Mapping ESA Land-Cover Classes to 11 Macro-Classes

ESA WorldCover uses a detailed set of class codes (e.g., 10 for Tree, 20 for Shrub, 50 for Built-Up, 80 for Cropland, etc.). For this assignment, these classes were mapped into **11 broader macro-categories** to simplify the classification problem.

Examples of mapping include:

- ESA Tree (10) → Tree
- ESA Shrub (20) → Shrub
- ESA Grass (30) → Grass
- ESA Cropland (80) → Cropland
- ESA Built-Up (50) → Built-up
- ESA Water (90) → Water
- ESA Bare/Sparse vegetation → Bare

This mapping step ensures consistent class labels for training and evaluation.

---

6. Handling Special and Uncertain Cases

Certain patches may not yield reliable or interpretable labels. Two types of ambiguous cases were handled explicitly:

A. No-Data or Invalid Regions

If a patch contained an excessive proportion of no-data values (caused by areas outside the ESA coverage), the image was labeled as **"uncertain"**.

B. Low Dominance of Any Class

If the dominant class accounted for **less than 20%** of the patch, the region was considered too mixed to assign a confident label.
Such samples were also assigned the label **“uncertain”**.

This strategy avoids contaminating the supervised dataset with noisy labels.

---

7. Creation of Label Datasets

After determining the label for each image:

- A combined dataset (labels.csv) containing image filename, coordinates, patch-based label, and dominance ratio was created.
- Invalid or uncertain entries were included but clearly marked.

These labels serve as the foundation for model training.

---

8. Train/Test Split (60% / 40%)

In preparation for the supervised learning task (Question 3), the dataset was split into **training (60%)** and **testing (40%)** subsets.
The split was performed in a stratified manner wherever possible to preserve the proportion of classes between splits.

The resulting files are:

- train_labels.csv
- test_labels.csv

Both files contain balanced subsets of the 9–11 defined macro-classes.

---

9. Class Distribution and Dataset Characteristics

Two visualizations were created to analyze the dataset distribution:

1. Global Class Distribution

class_distribution.png shows the frequency of each macro-class across all 9,216 samples.
Key observations:

- Built-up, Cropland, and Tree classes appear most frequently.
- Rare classes such as Herbaceous Wetland and Sparse Vegetation are minimally present.
- A non-trivial number of samples are labeled “uncertain”.

2. Train vs. Test Distribution

train_test_distribution.png compares class distributions between train and test sets.
The shapes match closely, confirming the split is well-balanced.

---

10. Discussion on Class Imbalance

The dataset shows clear evidence of **class imbalance**, with a few dominant classes and several underrepresented ones.
Such imbalance has several implications:

- Models tend to bias predictions toward majority classes.
- Minority classes (e.g., Wetlands) produce lower precision and recall.
- Class-weighting or sampling strategies are beneficial for improving performance.

To mitigate this, the model in Q3 uses **class-weighted cross-entropy loss**, ensuring better weighting of minority classes during training.

---

11. Deliverables Produced in Question 2

The following outputs were generated:

- labels.csv – complete labeled dataset
- train_labels.csv – training subset
- test_labels.csv – test subset
- class_distribution.png – label frequency visualization
- train_test_distribution.png – train vs test distribution
- patch extraction logs and label statistics within the script output

---

12. Conclusion

Question 2 successfully transformed raw image coordinates and ESA land-cover raster data into a clean, interpretable, and model-ready supervised learning dataset. The patch extraction, dominant-class labeling, mapping of ESA classes, handling of uncertainties, and systematic dataset splitting form a rigorous foundation for the classification model developed in Question 3. This labeled dataset now captures both the spatial essence of land-cover patterns and the statistical nuances necessary for reliable model training and evaluation.

Q3 — Model Training & Evaluation (ResNet-18)

For the supervised learning task, a **ResNet-18** architecture was trained to classify land-cover patches extracted in Q2. The input to the model consisted of 128×128 RGB images, and the labels corresponded to the dominant ESA land-cover class mapped to 11 macro-classes.

Training Setup:

- Framework: PyTorch
- Model: ResNet-18 (last FC layer replaced with 9-class classifier)
- Device: CPU (Intel i5-12500H)
- Loss Function: *Class-weighted* CrossEntropyLoss
- Optimizer: Adam (lr = 1e-3)
- Epochs: 10
- Batch size: 32

Class weights were computed using the inverse frequency of labels in the training set. This reduced bias toward majority classes such as Built-up and Cropland.

Evaluation Metrics:

Two macro-averaged F1 scores were computed:

1. **Custom implementation** using precision/recall per label
2. **TorchMetrics.F1Score** using the "multiclass" configuration

Both metrics were saved to metrics_summary.txt.

Results

- Best validation accuracy reached during training: **~56%**
- Custom macro F1 score: *(value saved in metrics_summary.txt)*
- TorchMetrics macro F1 score: *(value saved in metrics_summary.txt)*

Although validation accuracy fluctuated across epochs (expected due to class imbalance, CPU training, and limited dataset size), the model converged and produced stable predictions.

Confusion Matrix:

The confusion matrix (confusion_matrix.png) reveals:

- Strong performance on dominant classes

- Misclassifications between visually similar classes (e.g., Grass vs Shrub, Water vs Bare)
- Poor recognition for minority classes with few or zero samples in the test split

Correct & Incorrect Predictions:

The figure 5_correct_5_incorrect.png displays:

- Five correctly classified samples
- Five incorrectly classified samples
  Misclassifications generally occur on mixed pixels or boundary patches where two land-cover classes coexist.

Conclusion

The model successfully learned discriminative features from Sentinel-like RGB patches and produced meaningful performance metrics. While further improvement is possible via data augmentation, pretrained weights, or more balanced datasets, the results satisfy the expectations for Scenario-1 Q3 and demonstrate a complete supervised learning workflow end-to-end.

Thank You!!

Thank you so much for taking the time to review my work. I truly appreciate your effort and patience in going through the entire project. I have tried my absolute best to understand the requirements, follow them carefully, and present everything as clearly as I could. Working on this task taught me a lot, and I genuinely enjoyed building the entire pipeline step by step.

If I have missed something unintentionally, I sincerely hope you will forgive me.
Once again, thank you for your time, guidance, and consideration — it really means a lot.