Problem Statement-4

Overview:

Designing a Chain-of-Thought-Based LLM System for Solving Complex Spatial Analysis Tasks Through Intelligent Geoprocessing Orchestration

Geospatial challenges such as flood risk assessment, site suitability analysis, or land cover monitoring often demand sophisticated workflows involving various GIS tools and data sources. Traditionally, constructing these workflows requires expert knowledge and manual processes. This challenge invites participants to build a system that uses reasoning capabilities of Large Language Models (LLMs) to automatically plan and execute geospatial workflows—step-by-step—much like a human expert.

Objective:

- Develop a reasoning-enabled framework combining LLMs and geoprocessing APIs to auto-generate multi-step geospatial workflows from natural language queries.
- Enable integration of heterogeneous spatial datasets and libraries for tool/resource selection.
- Build an interface translating user queries into sequential geoprocessing tasks with transparent Chain-of-Thought reasoning.
- Demonstrate with benchmark tasks like flood mapping and site selection including input/output, metrics, and visualizations.

Expected Outcomes:

- Web/desktop application that generates runnable GIS workflows from natural language queries.
- Outputs include: JSON/YAML workflow definition, Chain-of-Thought reasoning logs, and GIS outputs.
- Demonstrations for tasks like flood-risk mapping and site suitability analysis.
- Comparative metrics for accuracy, runtime, resource use, and error management versus baseline/manual methods.

Dataset Required:

Open data from Bhoonidhi, OpenStreetMap (OSM), and other public geospatial sources.

Suggested Tools/Technologies:

- Open-source geoprocessing tools like QGIS, GRASS GIS, or GDAL.
- LLMs such as Mistral-7B-Instruct, LLaMA-3-8B, or Phi-2 integrated using frameworks like LangChain or Transformers.

Expected Solution / Steps to be followed to achieve the objectives:

- Select efficient pre-trained LLMs like Mistral or LLaMA-3 compatible with mid-tier GPUs.
- Implement Retrieval-Augmented Generation (RAG) using documentation from QGIS/GRASS/GDAL.
- Create a prompt library for Chain-of-Thought reasoning and in-context learning.
- Define common GIS functions in JSON schema; enable model to select operations and parameters.
- Use tools like GeoPandas, Rasterio, WhiteboxTools, or headless PyQGIS to execute workflows.
- Incorporate a reasoning-acting-observation loop to improve task outcomes iteratively.
- Develop a Streamlit-based UI for input, output visualization, Chain-of-Thought tracking, and layer downloads.
- Continuously enrich the RAG DB and prompt sets with validated successful workflows.

Evaluation Parameters:

- Evaluate logical and syntactic validity of generated workflow steps.
- Assess clarity and traceability of Chain-of-Thought reasoning explanations.
- Measure robustness against errors like CRS mismatches and geometry issues.
- Benchmark efficiency and usability across a wide range of spatial queries.

Image Representing Problem Statement: