



TECHNICAL REPORT

GeoGPT



Table of Contents

1. Executive Summary.....	3
2. Introduction	4
Challenges in Traditional GIS Workflows.....	4
How GeoGPT Addresses These Challenges.....	4
3. Key Features and Functionalities	5
1. Automated Code Generation.....	5
2. Intelligent Debugging.....	5
3. Advanced Data Handling and Visualization	5
4. Datasets and Case Study Overview.....	6
Overview of the Pesaro GIS Dataset.....	6
Case Study: GIS Visualization Tasks in Pesaro	8
Data Files Provided	8
Execution Process	9
Results.....	10
Code.....	11
5. Implementation Details	12
1. Overview of the Graphical User Interface (GUI).....	12
2. Key Functionalities of the GUI	12
a. Natural Language Task Input	12
b. File Browsing and Column Configuration	12
c. Model Selection and Execution	12
3. Technical Stack Used in Development.....	13
CustomTkinter:	13
Pillow:	13
Python Libraries:.....	13
Backend Integration:	13
6. Execution Workflow.....	13
1. Data Loading and Cleaning	13
2. Data Integration and Alignment	14
3. Visualization and Analysis Execution	14
4. Demonstration of Task Automation through Generated Python Scripts	14
5. Results.....	15
Building Height Map & Shapefiles	15
Year of Construction Map & Shapefiles.....	16

8. Conclusion.....	17
Summary of GeoGPT’s Impact on GIS Workflows	17
Future Potential for AI-Driven Geospatial Solutions	17
9. Appendix	17
Python Code Snippet	17
GUI Screenshots.....	18

1. Executive Summary

GeoGPT is a transformative framework designed to revolutionize Geographic Information Systems (GIS) by leveraging the power of Large Language Models (LLMs). Its primary purpose is to simplify and enhance geospatial data processing, analysis, and visualization through natural language instructions. GeoGPT serves as an intelligent assistant, enabling users—both experts and non-experts—to perform complex GIS tasks without requiring extensive technical expertise.

The core capabilities of GeoGPT include:

- **Natural Language Processing:** Understanding user-defined geospatial tasks and generating relevant workflows seamlessly.
- **Automated Code Generation:** Translating natural language queries into executable Python scripts tailored for GIS operations.
- **Interactive GUI:** A user-friendly interface for task configuration, file selection, and result visualization.
- **Dynamic Model Integration:** Supporting advanced LLMs like GPT-4 for enhanced reasoning and task execution.

GeoGPT's impact on GIS workflows is significant, addressing inefficiencies in traditional processes. By automating workflow creation and reducing the dependency on specialized knowledge, GeoGPT enables faster, more accurate analysis. This innovation democratizes GIS technologies, unlocking their potential for urban planning, environmental management, and more, thus paving the way for next-generation geospatial solutions.

2. Introduction

GeoGPT represents a groundbreaking advancement in the field of Geographic Information Systems (GIS), integrating cutting-edge Large Language Models (LLMs) to streamline geospatial data processing and analysis. Designed to bridge the gap between complex GIS operations and user-friendly solutions, GeoGPT empowers professionals and non-experts alike to efficiently handle geospatial tasks using natural language inputs. This framework combines advanced semantic understanding with GIS tools to offer a seamless, automated, and interactive experience for geospatial problem-solving.

Challenges in Traditional GIS Workflows

Traditional GIS workflows are often plagued by several limitations:

- **Complexity:** GIS operations typically require combining multiple tools and algorithms in a sequential workflow, which can be cumbersome and time-consuming.
- **Limited Adaptability:** Predefined workflows cannot accommodate even slight variations in user demands without manual modifications.
- **Steep Learning Curve:** Many GIS tools demand extensive technical expertise, making them inaccessible to non-specialized users.
- **Inefficiency:** Developing custom workflows for diverse tasks is labor-intensive and prone to human error.

How GeoGPT Addresses These Challenges

GeoGPT overcomes these barriers through its innovative design:

- **Natural Language Processing:** Users can describe tasks in plain language, which GeoGPT translates into executable workflows, eliminating the need for technical know-how.
- **Dynamic Workflow Creation:** GeoGPT generates customized workflows based on user input, allowing it to adapt to diverse and complex geospatial tasks.
- **Automation and Integration:** By leveraging LLMs to think, plan, and execute GIS operations step-by-step, GeoGPT ensures accuracy and efficiency.
- **Accessibility:** Its intuitive Graphical User Interface (GUI) provides a streamlined and user-friendly platform, democratizing GIS technologies.

By addressing these challenges, GeoGPT transforms traditional GIS workflows into a more flexible, efficient, and accessible system, opening new possibilities for professionals across various industries.

3. Key Features and Functionalities

GeoGPT is a robust framework that combines the capabilities of Large Language Models (LLMs) with Geographic Information Systems (GIS) tools to redefine how geospatial tasks are approached. Its core functionalities are designed to enhance productivity, reduce errors, and make GIS tasks accessible to a broader audience.

1. Automated Code Generation

- **Dynamic Workflow Creation:** GeoGPT translates natural language inputs into Python scripts, enabling users to execute complex GIS tasks without writing a single line of code.
- **Task Adaptability:** The platform generates customized code tailored to user requirements, accommodating a wide range of geospatial tasks.
- **Efficiency:** Automated code generation reduces the time and effort required to build workflows, allowing users to focus on analysis and decision-making.
- **Use Case Example:** A user can specify a task like "Generate a map of building heights and categorize them by year of construction," and GeoGPT produces executable code to achieve this seamlessly.

2. Intelligent Debugging

- **Error Identification:** GeoGPT's backend debugging engine identifies errors in generated code and provides suggestions for corrections.
- **Iterative Debugging:** In cases where initial execution fails, GeoGPT attempts multiple iterations of debugging, ensuring task completion without user intervention.
- **Feedback Mechanism:** The system alerts users with meaningful error messages and guides them to refine inputs if needed, fostering an interactive and user-friendly environment.
- **Reliability:** This feature ensures robustness in execution, even for highly complex tasks.

3. Advanced Data Handling and Visualization

- **Comprehensive Data Management:** GeoGPT seamlessly integrates CSV files and shapefiles, allowing users to specify columns of interest and set preferences for data attributes.
- **Interactive Mapping:** The system supports advanced geospatial visualizations, such as color-coded maps and chronological urban development maps, to help users derive actionable insights.
- **Dynamic Visualization Options:** Outputs are generated in visually appealing formats and saved in task-specific directories for easy retrieval and sharing.
- **Enhanced Accessibility:** Non-technical users can process and visualize geospatial data with minimal effort, leveraging GeoGPT's intuitive GUI.

GeoGPT's combination of automated code generation, intelligent debugging, and advanced data handling redefines the standards of GIS workflows, making it an indispensable tool for professionals and researchers in various fields.

4. Datasets and Case Study Overview

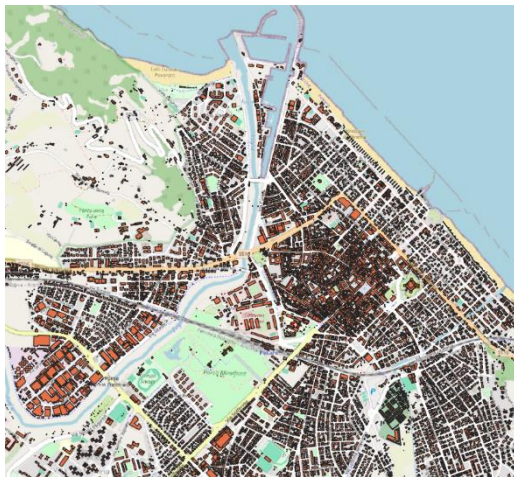
Overview of the Pesaro GIS Dataset

The GeoGPT project leverages a rich and diverse dataset that captures the intricate urban landscape of Pesaro, Italy. This dataset integrates both spatial (shapefiles) and non-spatial (CSV files) data to support comprehensive geospatial analysis and visualization. Below are the key components of the dataset and their significance:

- **Addresses:** Provides precise geolocations for navigation and urban planning, forming a foundational layer for municipal operations and decision-making.



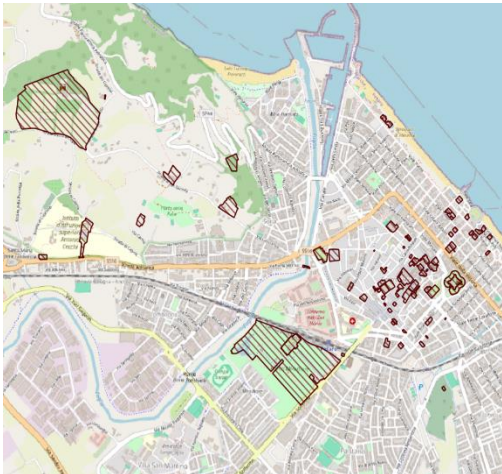
- **Buildings2005Pesaro:** Contains vital data on building heights and construction dates, enabling urban morphology studies, real estate assessments, and historical development analysis.



- **Districts2019:** Delivers demographic information and administrative boundary delineations at the district level, essential for effective policy-making, resource allocation, and governance.



- **MonumentalBond:** Highlights protected cultural and historical zones, aiding in cultural heritage conservation, tourism management, and preservation efforts.



- **Neighborhoods2019:** Offers insights into socio-economic layouts and community structures, supporting targeted interventions for local governance and public services.



- RoadMap: Captures the intricate roadway network, indispensable for infrastructure development, transportation management, and route optimization.



- SectionsISTAT: Provides granular statistical data on population and housing, structured by statistical units, essential for socio-economic analysis and reporting.



Case Study: GIS Visualization Tasks in Pesaro

The goal was to comprehensively visualize the GIS data related to building structures within the city of Pesaro. The specific objectives for GeoGPT were:

- To generate a visualization map that represents building heights across the city.
- To produce a chronological display of buildings based on their year of construction to identify historical patterns and development trends.

Data Files Provided

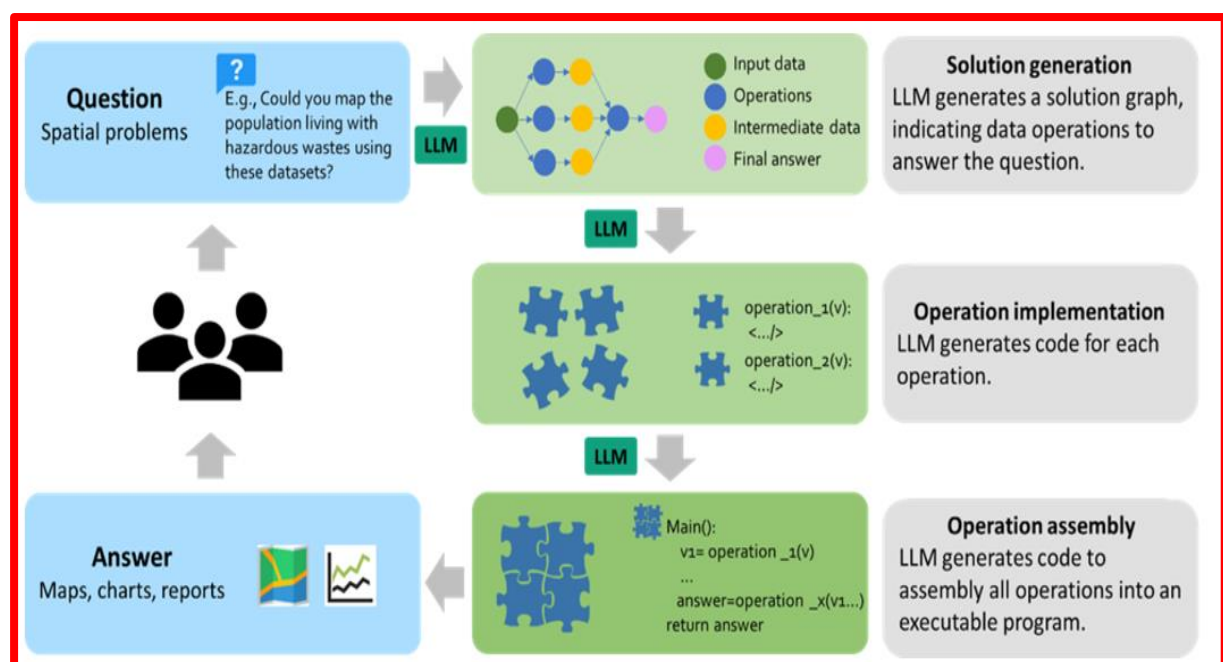
- Buildings2005Pesaro.csv: Contains detailed attributes of buildings including their heights and construction years, stored at 'CSV GIS Pesaro/Buildings2005Pesaro.csv'.
- edifici2005.shp: A shapefile that provides the spatial geometry of buildings, stored at 'Dati_Pesaro/edifici2005.shp'.

Both datasets include CSV and shapefile formats to ensure comprehensive spatial and attribute data integration.

Execution Process

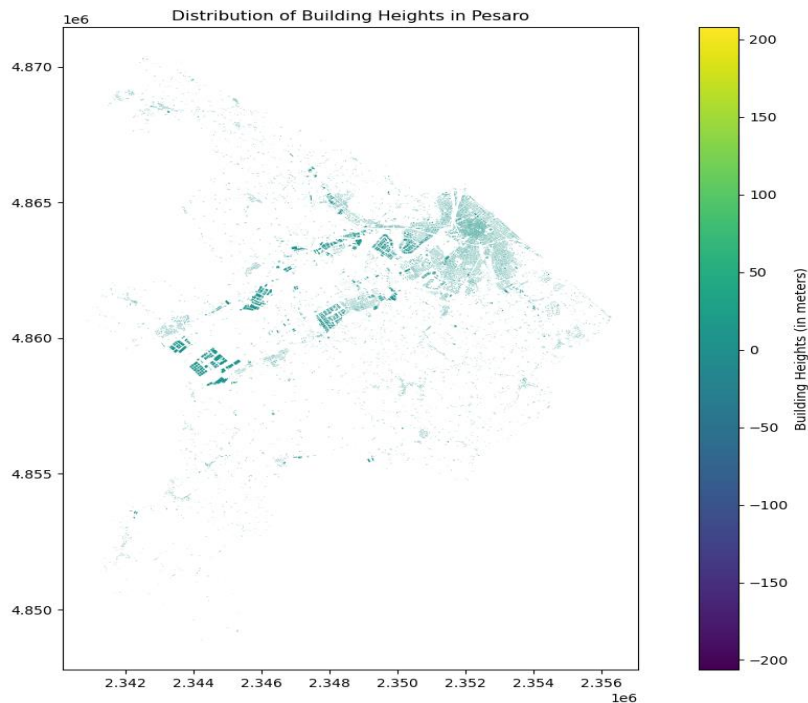
The LLM-GPT-4 model, utilizing the GeoGPT framework, generated a script to process and integrate both the CSV and shapefile data. Key steps included:

1. **Loading and Cleaning Data:** Both the CSV and shapefile data were loaded, with a cleaning process to eliminate records with missing values, ensuring high-quality spatial analysis.
2. **Data Integration:** The datasets were joined based on common attributes, such as building height (height in CSV and altezza in the shapefile) and year of construction (year_ctr in CSV and annoctr in the shapefile).
3. **Geospatial Processing:** The script adjusted map projections to ensure that both datasets aligned spatially for accurate geographic representation.
4. **Visualization Execution:** Utilizing matplotlib and geopandas, two maps were created:
 - A map color-coded by building heights to visualize vertical development across Pesaro.
 - A chronological map of buildings highlighting different construction epochs.

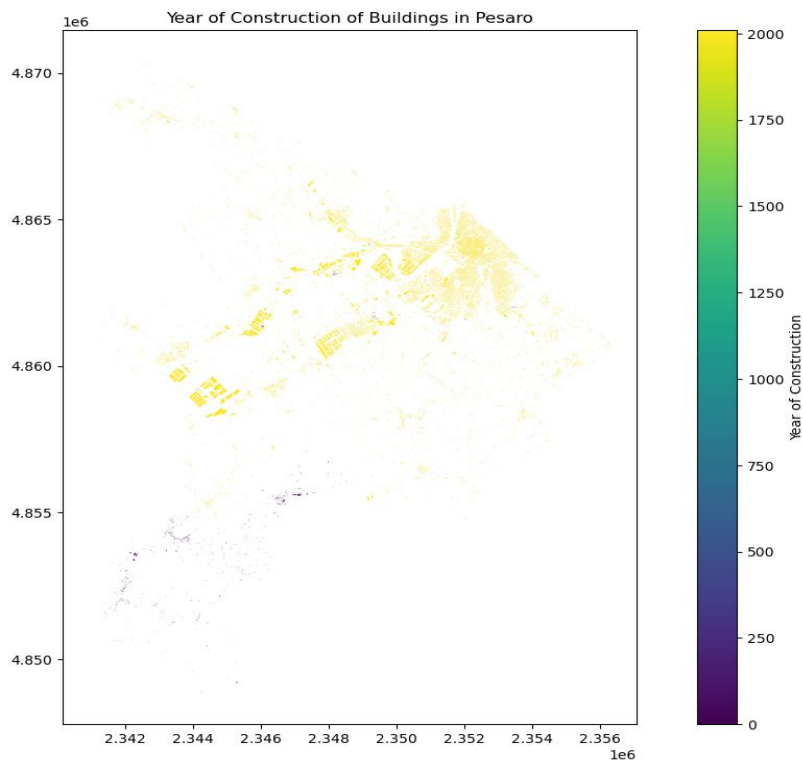


Results

Building Height Map: This map displays the distribution of building heights, using a gradient color scheme to easily identify areas with taller structures versus lower ones. It is instrumental in urban planning and zoning.



Year of Construction Map: Reveals the temporal distribution of building activities over decades, which is crucial for historical analysis and future urban planning initiatives.



Code

```
GeoGPT.py
C:\Users\Utente\GeoGPT\LLM-Geo\GeoGPT.py
1 import os
2 import openai
3 from LLM_Geo_kernel import Solution
4 import LLM_Geo_Constants as constants
5 import helper
6
7 # Case 3: Visualization of Building Data in Pesaro
8 task_name = 'Pesaro_Building_Visualization'
9
10 TASK = """
11 1) Generate a map to show the distribution of building heights in Pesaro.
12 Color-code the buildings by height to indicate different ranges of building heights.
13 Note that the building height column is 'height' in the CSV and 'altezza' in the shapefile.
14
15 2) Generate another map to show the distribution of buildings based on the year of construction,
16 highlighting historical and more recent buildings.
17 Note that the year of construction column is 'year_ctr' in the CSV and 'annoctr' in the shapefile.
18 """
19
20 DATA_LOCATIONS = [
21     "Building data from 'Buildings2005Pesaro.csv' which contains information such as building heights (column: 'height') ;",
22     "Shapefile for the buildings in Pesaro from 'edifici2005.shp', which provides the spatial geometry for each building ;",
23 ]
24
25 save_dir = os.path.join(os.getcwd(), task_name)
26 os.makedirs(save_dir, exist_ok=True)
27
28 # Assuming Solution is a class capable of handling tasks with GIS data
29 model = "gpt-4o"
30 solution = Solution(
31     task=TASK,
32     task_name=task_name,
33     save_dir=save_dir,
34     data_locations=DATA_LOCATIONS,
35     model=model,
36 )
37
38 print("Prompt to get solution graph:\n")
39 print(solution.direct_request_prompt)
40
41 try:
42     direct_request_LLM_response = solution.get_direct_request_LLM_response(review=True)
43     code = solution.execute_complete_program(code=solution.direct_request_code, try_cnt=10)
44     print(code)
45 except Exception as e:
46     print(f"An error occurred: {e}")
```

Prompt

5. Implementation Details

1. Overview of the Graphical User Interface (GUI)

The Graphical User Interface (GUI) of GeoGPT is designed to simplify interaction with geospatial data by enabling users to perform complex GIS tasks using a user-friendly interface. Its intuitive layout reduces the technical barrier for users, ensuring accessibility for both GIS professionals and non-experts.

The GUI provides a seamless experience by integrating natural language task input with backend automation, allowing users to efficiently process geospatial data without requiring detailed coding expertise.

2. Key Functionalities of the GUI

a. Natural Language Task Input

- Users can input tasks using natural language, describing the desired GIS operation or analysis.
- GeoGPT translates these queries into executable code dynamically, leveraging the capabilities of Large Language Models (LLMs).
- This feature ensures adaptability to a wide range of geospatial workflows without requiring predefined scripts or configurations.

b. File Browsing and Column Configuration

- The GUI includes file dialog boxes for seamless browsing and selection of CSV files and shapefiles.
- Users can specify the columns of interest in each file through dedicated input fields.
- Language preference (English or Italian) for column names can be selected using integrated checkboxes, ensuring flexibility for multilingual datasets.
- An optional "None" button allows users to skip specifying columns, streamlining workflows where such inputs are unnecessary.

c. Model Selection and Execution

- Users can select from available LLMs, such as GPT-4o, using a dropdown menu.
- A single-click "Execute" button automates the task execution process, leveraging the backend system to handle data processing, visualization, and output generation.
- A "Refresh" button is provided to reset the interface for a new task, ensuring a clean workspace for each session.

3. Technical Stack Used in Development

The GeoGPT GUI leverages modern technologies and libraries to deliver a seamless user experience:

CustomTkinter:

- Provides advanced theming and UI components for building modern, aesthetically pleasing interfaces.
- Supports features such as dark mode and custom color themes for enhanced usability.

Pillow:

- Handles image manipulation, including resizing and transparency, to integrate a branded background logo into the GUI.

Python Libraries:

- Geopandas: For geospatial data manipulation and analysis.
- Matplotlib: For creating geospatial visualizations, such as thematic maps.
- OpenAI API: For integrating LLMs to dynamically generate code and automate workflows.

Backend Integration:

- The GUI seamlessly communicates with backend modules, such as the Solution class, to execute tasks and generate outputs.
- Outputs, including processed data and visualizations, are saved in dynamically created folders named after the task for easy organization and access.

The combination of an intuitive GUI, natural language input, and advanced backend processing makes GeoGPT a powerful and user-friendly tool for geospatial data analysis. The technical stack ensures scalability, adaptability, and robustness, catering to the diverse needs of GIS professionals and researchers.

6. Execution Workflow

Step-by-Step Process

1. Data Loading and Cleaning

The execution process begins with users uploading their geospatial datasets through the GUI. Supported formats include:

- CSV files for tabular data.
- Shapefiles for spatial geometries.

GeoGPT validates the uploaded files to ensure compatibility and consistency:

- Checks for missing or invalid data entries.
- Verifies the integrity of shapefile geometries.

If required, GeoGPT prompts users to specify the columns of interest or language preferences for data interpretation, ensuring alignment with the analysis requirements.

2. Data Integration and Alignment

GeoGPT integrates datasets from different formats (e.g., CSV and shapefiles) to align them for analysis.

This step involves:

- Spatial joins to merge shapefiles with tabular data.
- Ensuring consistent Coordinate Reference Systems (CRS) across all spatial datasets.
- Resolving discrepancies in field naming or formatting between datasets.

The platform automates these tasks, saving time and reducing potential errors compared to manual data preparation.

3. Visualization and Analysis Execution

Once the datasets are processed, GeoGPT translates the user's natural language query into Python scripts tailored for the specific task.

The system supports a range of geospatial operations, such as:

- Mapping Building Heights: Generates a color-coded map highlighting height distributions.
- Urban Growth Analysis: Creates temporal visualizations based on construction years.
- Buffer and Intersect Operations: Identifies areas of interest based on proximity criteria.

These visualizations and results are saved in an output folder named after the task, ensuring easy identification and retrieval.

4. Demonstration of Task Automation through Generated Python Scripts

GeoGPT automates the entire workflow by dynamically generating Python scripts that correspond to the user's query.

Key features of the scripts:

- Task-Specific Functionality: Scripts are generated based on user-defined tasks, ensuring tailored solutions.
- Debugging Capability: Scripts include error-handling mechanisms to manage execution issues.
- Scalable Workflows: Modular design allows scripts to be reused or extended for similar tasks.

Once the scripts are executed, GeoGPT evaluates their outputs and, if necessary, retries with refined code, ensuring robust task completion.

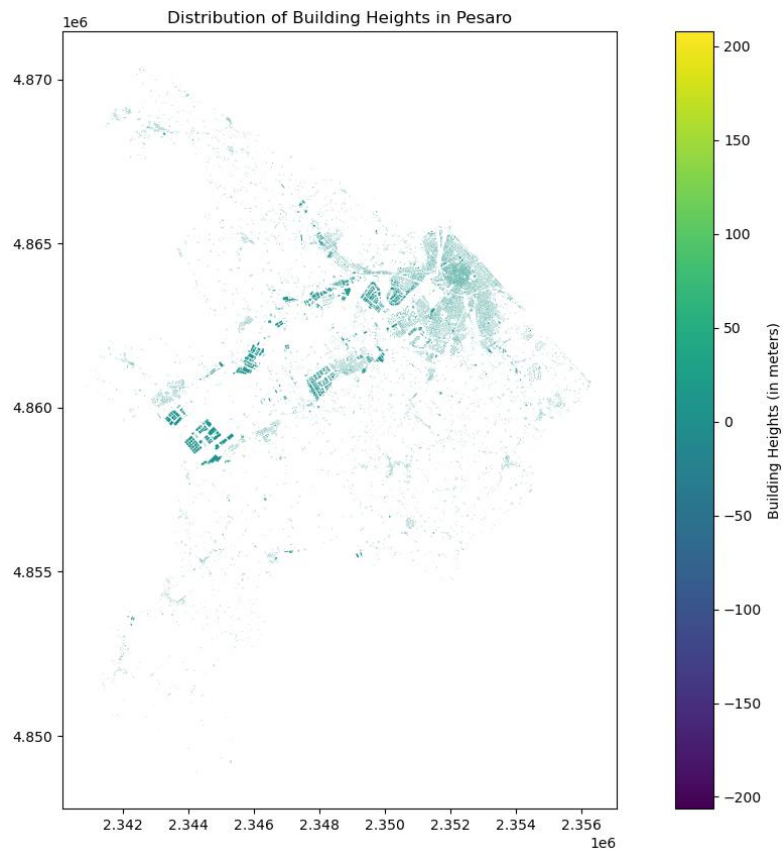
This workflow exemplifies GeoGPT's ability to bridge the gap between complex GIS operations and user-friendly automation. By combining data preparation, task-specific code generation, and seamless visualization, GeoGPT streamlines the entire geospatial analysis pipeline for professionals and non-experts alike.

5. Results

Maps and ShapeFiles Generated

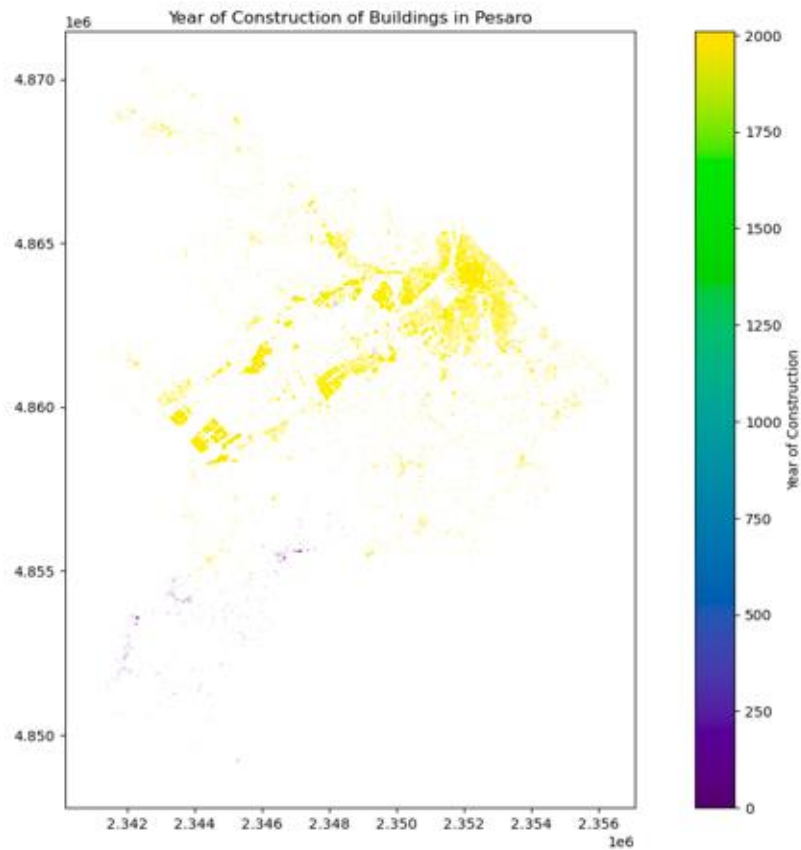
Building Height Map & Shapefiles

- A color-coded visualization depicting the distribution of building heights in Pesaro.
- Provides insights into urban density and architectural trends.
- Highlights areas with concentrated high-rise structures versus low-density residential zones.



Year of Construction Map & Shapefiles

- A chronological representation of urban development over time.
- Segregates buildings into temporal categories to showcase historical and recent construction trends.
- Highlights areas of rapid urbanization or redevelopment.



8. Conclusion

Summary of GeoGPT's Impact on GIS Workflows

GeoGPT has revolutionized GIS workflows by seamlessly integrating natural language processing capabilities with geospatial tools. Its ability to interpret user queries, automate complex geospatial tasks, and generate meaningful outputs has made GIS more accessible and efficient. By bridging the gap between expert GIS users and non-technical stakeholders, GeoGPT has redefined how geospatial data is analyzed, visualized, and utilized for decision-making.

Future Potential for AI-Driven Geospatial Solutions

GeoGPT paves the way for AI-driven advancements in geospatial technology. Its modular and extensible architecture allows for the inclusion of advanced GIS functionalities and domain-specific enhancements. Future iterations could integrate real-time data analysis, multi-modal inputs, and enhanced visualization techniques. GeoGPT exemplifies the transformative potential of AI in addressing complex geospatial challenges, empowering industries to achieve more informed and impactful decisions.

9. Appendix

Sample Python Code and GUI Screenshots

Python Code Snippet

A sample of the dynamically generated Python script used for processing spatial data is included to demonstrate the automation capabilities of GeoGPT.

```
... ```python
import matplotlib.pyplot as plt
import pandas as pd
import geopandas as gpd

# Function to process the building data and generate maps
def direct_solution():

    # File paths
    csv_file = 'FB_world/CSV GIS Pesaro/Buildings2005Pesaro.csv'
    shp_file = 'FB_world/Dati_Pesaro/edificio2005.shp'

    # Load the csv file into a dataframe and drop records with NaN cells
    df = pd.read_csv(csv_file)
    df.dropna(subset=['height', 'year_ctr'], inplace=True)

    # Load the shapefile into a geodataframe and drop records with NaN cells
    gdf = gpd.read_file(shp_file)
    gdf.dropna(subset=['altezzaa', 'annoctr'], inplace=True)

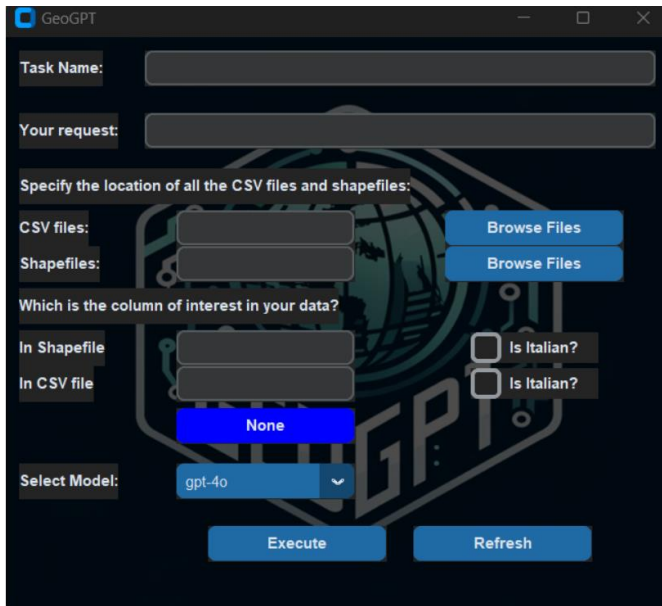
    # Join dataframe and geodataframe using common columns ('height' and 'year_ctr')
    merged = gdf.set_index('altezzaa').join(df.set_index('height'))

    # First Map - Building Heights
    fig, ax = plt.subplots(1, 1, figsize=(15, 10))
    ...

----- Running code (trial # 3/10) -----
```

GUI Screenshots

Screenshots of the intuitive GUI interface highlighting key features, including task input fields, file browsing, and execution controls.



The screenshot displays the GeoGPT application window with a dark theme. The interface includes the following elements:

- Task Name:** A text input field.
- Your request:** A text input field.
- Specify the location of all the CSV files and shapefiles:**
 - CSV files:** A text input field with a **Browse Files** button.
 - Shapefiles:** A text input field with a **Browse Files** button.
- Which is the column of interest in your data?**
 - In Shapefile:** A text input field with a **None** button.
 - In CSV file:** A text input field.
 - Is Italian?** Two checkboxes, one for each input field.
- Select Model:** A dropdown menu currently showing **gpt-4o**.
- Execute** and **Refresh** buttons at the bottom.