MACSS 30112 Winter 2025
Project Report 2
Group Chick-Fil-A
Charlotte Li, Baihui Wang & Anqi Wei

Group Project: Investigating the Global Spatial Distribution of Open-Source Artificial Intelligence Development

1. Project Description

The project has successfully expanded its scope from a U.S.-centric focus to a global analysis, as evidenced by the updated code's ability to scrape and geocode contributor locations worldwide. The GitHub scraping script now collects data on AI repositories across multiple geographic regions, and the data cleaning pipeline standardizes locations using both OpenStreetMap and Google Maps APIs. Initial findings confirm geographic clustering in tech hubs like the U.S., China, and Europe. A key change was the addition of token rotation in the scraping script to circumvent GitHub API rate limits, ensuring scalability.

In the next stage, we will finalize hypotheses testing by integrating socioeconomic variables (e.g., GDP, university density) and refine the definition of "AI hubs" by correlating scraped data with external indicators like venture capital investment.

2. Data Sources

GitHub data scraping is fully operational, with 10,000 repositories targeted and contributor locations extracted. The code now handles pagination and token rotation, addressing reliability concerns about API limits. Next, we'll download and merge external datasets (World Bank GDP, WHED, university density, internet penetration, regional venture capital investment) to enable regression analysis.

3. Data Cleaning and Wrangling

The current *data_clean.py* script standardizes country names, fills missing cities using mode imputation per country, and geocodes "Unknown" locations via API fallbacks. For location data, since the precision of GitHub locations varies—from specific campus addresses to general country names—we use the **OpenStreetMap** model to geocode each location and determine the corresponding city. This process helps us accurately identify each contributor's location while also standardizing entries that may be written in different languages. Additionally, this step prepares the data for geographic visualization, ensuring consistency and accuracy in our analysis.

While this approach improves data completeness, it may introduce biases toward frequently reported cities, as discussed in our team meetings. We will refine this methodology in the next stage. Changes include adding IP address filtering to exclude invalid entries and improving error handling for API timeouts. Aggregated outputs (city/country-level contributor counts, stars) are saved to CSV files. Furthermore, we used Google Translator API to standardize location names across different languages. However, Google Translate occasionally introduces errors, particularly when users use nicknames for cities. Thus, manual validation is necessary.

Since location names often contain spelling variations, abbreviations, and inconsistent formats, making direct string matching ineffective. To enhance accuracy, we used ***pycountry*** to clean and validate country names. Since *pycountry* primarily focuses on country-level data, we implemented custom cleaning and high-threshold fuzzy matching for cities and detailed location names using **recordlinkage** and **TF-IDF-based PolyFuzz**.

After the cleaning and merge phase, we identified inconsistencies such as city-country mismatches and misclassified locations, leading to data loss and inaccuracies. Some errors stem from the models we use, even those that are relatively advanced. In the next steps, we will resolve edge cases in geocoding (e.g., mismatched city-country pairs), cross-validate cleaned locations, reconsider our data structure, and conduct manual adjustments for high-risk cases.

Additionally, merging with population data for per capita metrics remains incomplete. Despite contributing to high-starred repositories, some contributors may be from countries with relatively few contributors, leading to abnormally high average stars per contributor. To mitigate this, we plan to integrate UN population data for AI Open-Source Performance Index (AIOSPI) calculations.

4. Data Analysis and Visualization

Aggregation by city and country is complete, with summary statistics (total contributors, stars) generated. Initial visualizations (e.g., CSV exports) are functional, but advanced spatial visualizations (heatmaps, network graphs) and regression models are pending. The current code calculates raw activity metrics but not the AIOSPI indices outlined in the proposal.

Next Steps:

- Implement AIOSPI formulas (geographic and per capita) using merged population data.
- Use GeoPandas to create global heatmaps of contributor density.
- Conduct OLS regression in Python to test H1 (GDP/university density vs. AIOSPI).
- Build a NetworkX collaboration graph for top repositories to visualize cross-regional ties.

- Using Folium and Geopandas to create interactive map to show the clustering of repos in different regions, specifically, at city and country levels.

5. Additional Notes

- Challenges: Geocoding APIs occasionally return inconsistent results (e.g., “San Francisco” mapped to Argentina). Manual validation of top cities is needed.
- Collaboration: Roles remain aligned with the initial plan, with Baihui leading data cleaning, Charlotte focusing on visualization, and Anqi finalizing scraping.
- Timeline: Aim to complete regression and visualization by Week 8, leaving Week 9 for refining presentation materials and rehearsing findings.