Progress Report #2

DS6051

Kiana Dane

Project: Instruction-Tuned LLM for Construction and Utilities

March 20, 2025

Data Sourcing

San Francisco Utility Permits API: I used the Socrata API to download over 100,000 utility excavation permit records (smdf-6c45) from San Francisco's open data portal.

Water Quality Environmental Sensors: I queried the national Water Quality Portal to retrieve environmental monitoring station data across California (state code US:06), with a focus on station locations in San Francisco.

Geocoding Permits

To enable spatial joins between permit data and sensor locations, I geocoded all permit streetnames using OpenStreetMap's Nominatim service. Since Nominatim has rate limits and requires a proper user-agent header, I configured and tested a polite request loop on my Ubuntu EC2 instance. The geocoding process standardized street names, resolved missing latitude/longitude fields, and saved progress every 100 rows for fault tolerance.

With geocoded permits and a dictionary of four known sensor station coordinates in San Francisco, CA, I implemented a spatial filter using scipy.spatial.cKDTree to identify permits within a 2-km radius of each station. This significantly narrowed the search space and enabled further pairing logic for synthetic example generation.

Hugging Face Model Access

I gained API access to Mistral-7B-Instruct-v0.3 via Hugging Face's InferenceClient. After authentication and token configuration, I verified generation using the hosted endpoint and wrote a minimal client wrapper class using langchain's LLM interface for modularity and reusability.

Tool Development

The two tools I implemented to be invoked via structured output from the LLM are:

1. Permit Lookup Tool
    a. Accepts a location and date range and returns matching permits with metadata (e.g., contractor, activity type). Backed by the geocoded permit dataset.
2. Anomaly Retrieval Tool
    a. Accepts a sensor ID and time window, returning relevant environmental measurements (e.g., spikes in ammonia or nitrate).

Each tool was implemented with predictable input/output schemas to allow for LLM fine-tuning or function calling integration.

Synthetic Example Generation

To construct a high-quality instruction tuning dataset, I developed a Python pipeline that synthesizes realistic input-output pairs by joining environmental sensor readings with nearby construction permits and prompting a language model to generate explanatory hypotheses that I can use to train the model. To avoid overfitting the model to a fixed format, I defined multiple text templates to express sensor observations in diverse, natural-sounding ways.

Ex. 1: Could the increase in Phosphorus levels on WESTGATE DR on July 21, 2010, be related to the recent excavation work for the sewer line installation, potentially causing soil erosion and phosphorus runoff?

Ex. 2: Could the increase in Phosphorus levels on APPLEWOOD DR on September 1, 2008, be related to the recent excavation work for the sewer line installation, potentially causing soil erosion and phosphorus runoff?

Hugging Face Model Integration via LangChain

I created a lightweight wrapper class to connect to “mistralai/Mistral-7B-Instruct-v0.3” using langchain’s LLM interface and the Hugging Face Inference API. The model was set to generate up to 512 tokens per prompt with moderate temperature of 0.3 for creativity. I loaded sensor and permit records and performed basic cleaning. Dates were parsed, and street names were normalized to improve matchability. Since the datasets lacked explicit shared keys, I performed a manual join using substring matching on normalized street

names. Each matched sensor-permit pair was merged into a single dictionary for later. This resulting merged dataset represented plausible real-world scenarios where construction may influence environmental readings.

Next Steps

The next steps of the project will focus on fine-tuning and evaluation.

1. Fine-Tune the Base Model Using LoRA
2. Integrate tools with model
3. Evaluate model output & filtering
4. Error analysis