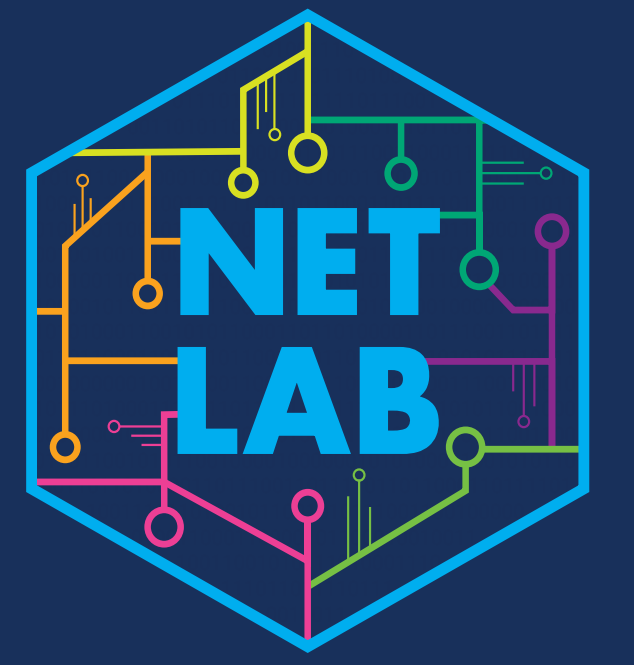


T.R.A.C. Air: Traffic-camera Real-time Analysis for Clean Air

BYU
Electrical & Computer
Engineering

Jason Rao, Christopher Kitras, Philip Lundrigan
Electrical and Computer Engineering Department



Challenge

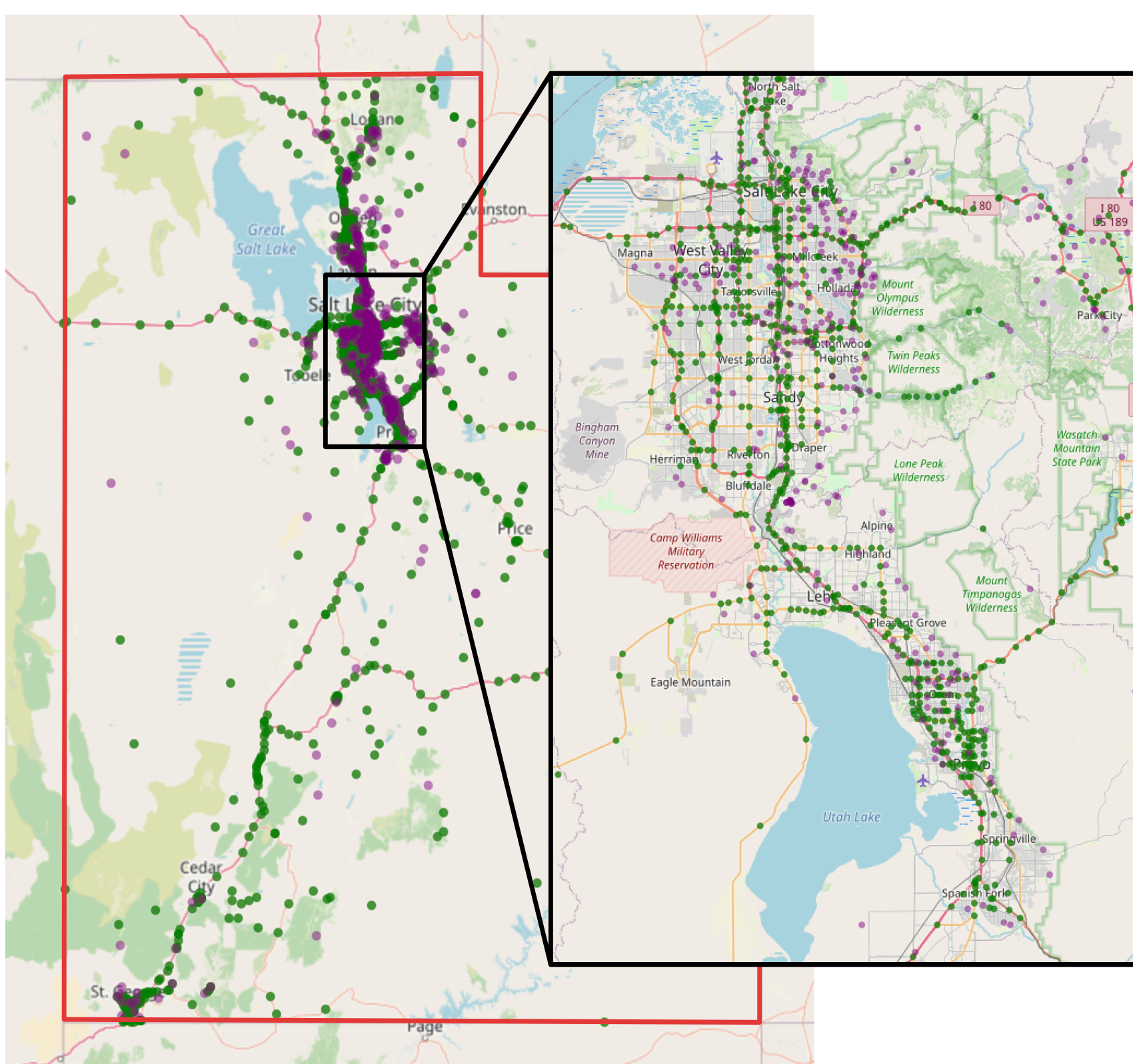
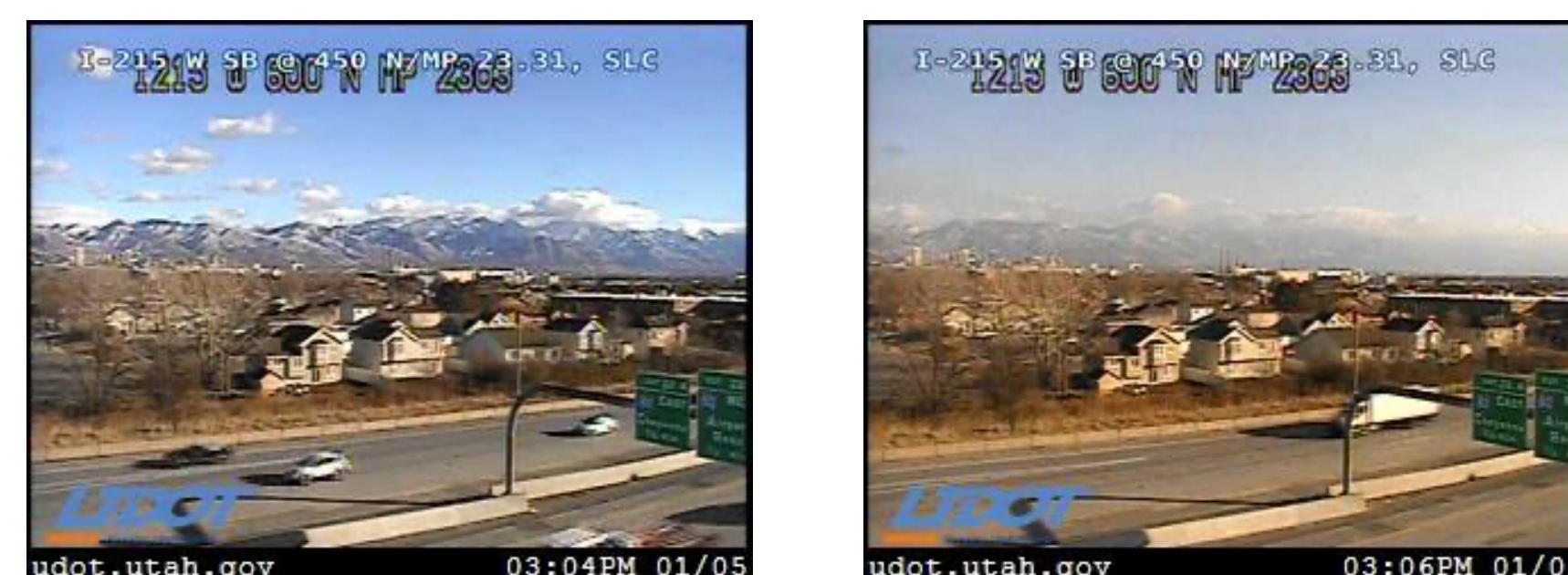
In 2021, air pollution was the **second leading risk factor** for early death worldwide. **To help improve air quality, we must monitor and quantify the air quality problem.**

Monitoring air quality traditionally relies on expensive, government-maintained stations, but these stations are geographically sparse. There are only **two stations in all of Utah County!**

Citizens have started to deploy their own privately owned sensors through a network called **PurpleAir**. These sensors provide fine-grain, real-time air quality data. However, these sensors **cost money and require stable infrastructure, leaving lower-income and developing areas with less coverage.**



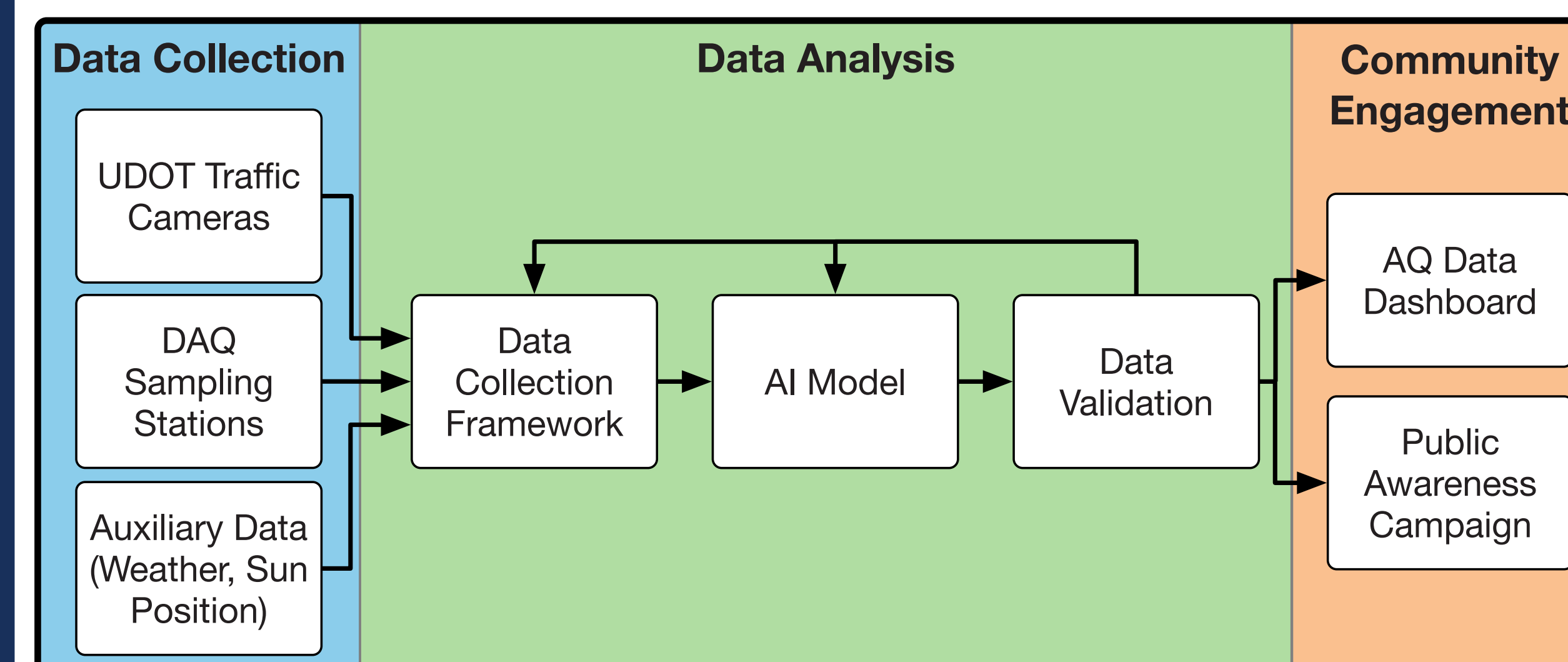
Can we reuse existing infrastructure to measure air quality?



There are **1,395 publicly available traffic cameras** (green dots) in Utah and **778 PurpleAir sensors** (purple dots). With hundreds of deployed traffic cameras, we gain far more data points than DAQ stations and private sensors alone.

Methodology

To create a system that collects, analyzes, and disseminates air quality data, we break the process down into three stages: **data collection, data analysis, and community engagement.** Our work currently focuses on the first two stages.



Data Collection

We collect data by deploying a series of Python scripts that scrape data from their appropriate sources. For the traffic camera air quality monitoring, we have collected over **235 million images from 1,395 traffic cameras across Utah from 2021–2025.** We have also collected corresponding hourly PM2.5 values from DAQ stations as training data.

Data Analysis

Using Google's open-source **multimodal LLM** (Gemma 3) with 4 billion parameters, we have created a prompt that can take in a traffic camera image and weather conditions and output an air quality estimate.

```
You are an air quality estimation model. Your task is to predict the PM2.5 value (µg/m³) based on a detailed weather description and a daytime camera image.

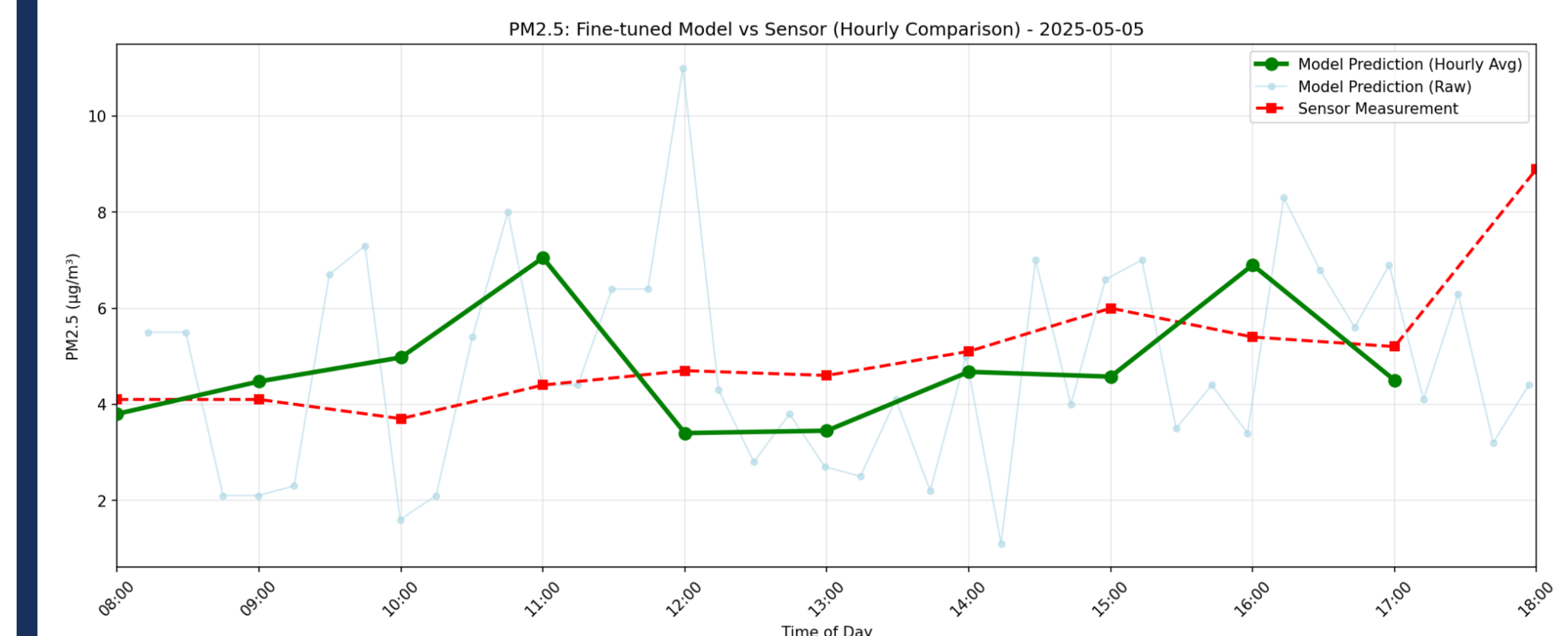
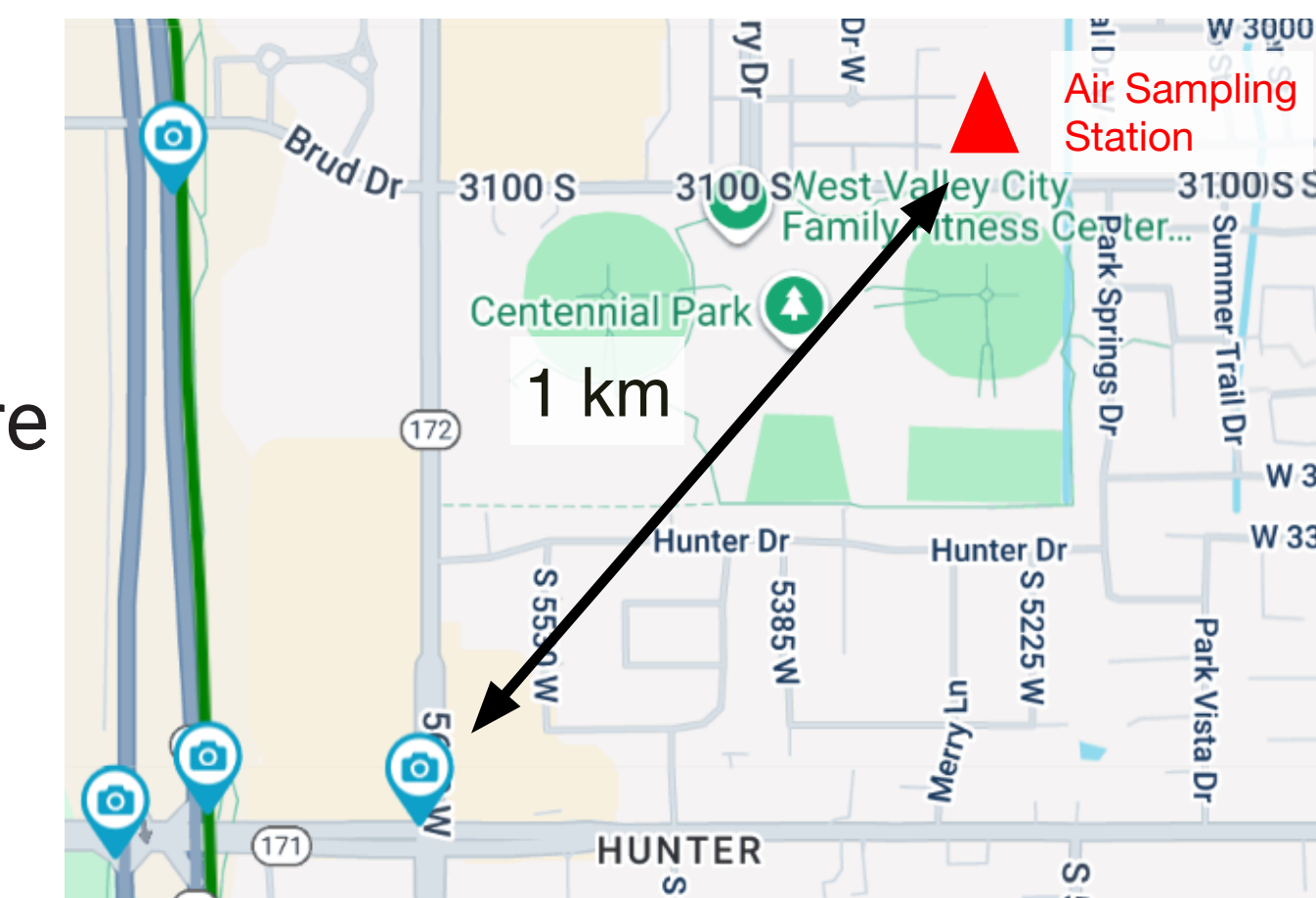
Weather at the time of the image was taken:
- Visibility: {weather.get('visibility', 'Unknown')} miles
- Cloudcover: {weather.get('cloudcover', 'Unknown')}%
- Weather: {weather.get('description', 'Unknown')} When this shows overcast condition, lower the pm2.5 prediction.

Ignore the traffic, and focus on specific features in the environment that may affect PM2.5 levels. Do not overestimate haze unless the image shows distinct gray smog or loss of distant detail beyond what clouds would cause. Now analyze the attached image and estimate the PM2.5 level in µg/m³.
```

To improve performance, we have customized the model, called instruction tuning using **low-rank adaptation (LoRA)**, without changing the entire model. We customize the model with 10,000 traffic camera images at different times of day and seasons.

Results

We have tested with multiple cameras in Utah **using nearby government-run sampling stations as ground truth.** We are interested in root mean square error (RMSE) of the estimated value produced by our model.



We generate estimates for all images of a camera during an hour and average the estimates together. Early results are promising—using one camera and estimating for four days, we are able to achieve an **RMSE of 0.76 µg/m³**. This performs better than prior work that used traditional machine learning approaches.

Future Work

There are many traffic cameras surrounding air quality sampling stations. We would like to **use multiple cameras at once** to improve the quality of our results.

We need to evaluate the **transferability** of our model: how well does it work at other locations? How well does it perform during different seasons? How well does it perform at night?

We have fine-tuned our model with 10,000 traffic camera images, but we have access to **235 million images**. We can do more fine-tuning of our model, at the cost of long training times.