VayuBuddy: LLM-powered natural language interface for exploring and understanding air pollution data

Yash Bachwana*
yash.bachwana@iitgn.ac.in
Indian Institute of Technology
Gandhinagar, Gujarat, India

Khush Shah*
khush.shah@iitgn.ac.in
Indian Institute of Technology
Gandhinagar, Gujarat, India

Nitish Sharma
nitishsharma1295@gmail.com
India

Zeel B Patel
patel_zeel@iitgn.ac.in
Indian Institute of Technology
Gandhinagar, Gujarat, India

Nipun Batra
nipun.batra@iitgn.ac.in
Indian Institute of Technology
Gandhinagar, Gujarat, India

Sarath Guttikunda
sguttikunda@gmail.com
UrbanEmmissions.info
India

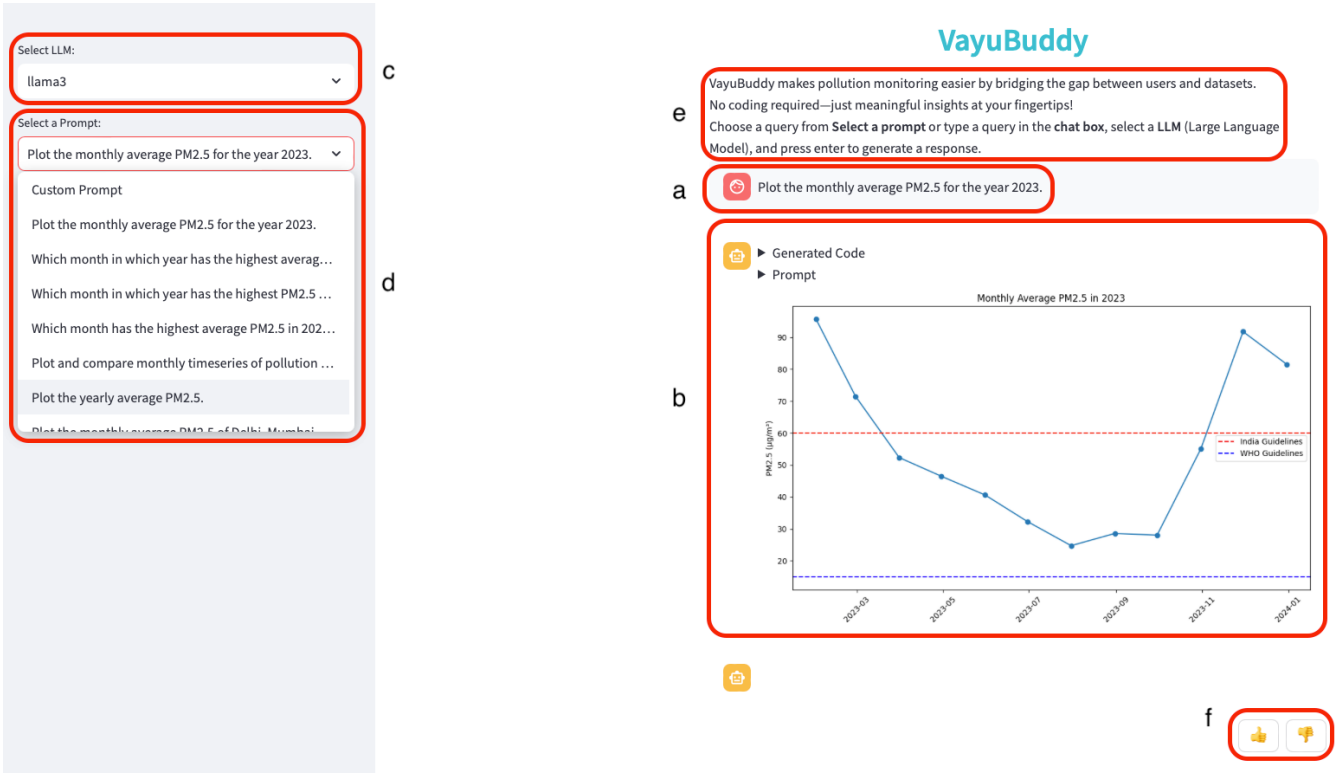

Figure 1: VayuBuddy's Interface. a) Prompt given by the user. b) VayuBuddy's response to the prompt. c) 3 LLMs to choose from. d) Dropdown menu to choose pre-specified prompts from. e) Description and instructions about VayuBuddy. f) Thumbs-Up and Thumbs-Down buttons are used to collect user feedback for each response from VayuBuddy.

*Authors contributed equally to this research.

ACM COMPASS '24, June 03–05, 2024, IIIT Delhi, Delhi, India

ABSTRACT

Almost 6.7 million lives are lost to air pollution each year, making it one of the biggest threats to environmental health. However, the difficulties in communicating about air pollution obstruct public awareness and action. The gaps in the current ecological health literacy further highlight this problem. In this work, we present VayuBuddy: a LLM-powered natural language interface to explore and understand air pollution

ACM ISBN 978-x-xxxx-xxxx-x/YY/MM…$15.00
https://doi.org/10.1145/nnnnnnn.nnnnnnn

data. VayuBuddy is a web application that can comprehend user requests and offer customised replies based on the data from the pollution board in India. We create a dataset of natural language prompts and find that LLMs are highly accurate in generating code that gets executed to answer the query from the dataset. Systems can such as VayuBuddy can lower the entry-barrier into understanding and (potentially mitigating) air pollution.

ACM Reference Format:
Yash Bachwana, Khush Shah, Nitish Sharma, Zeel B Patel, Nipun Batra, and Sarath Guttikunda. 2024. VayuBuddy: LLM-powered natural language interface for exploring and understanding air pollution data. In *Proceedings of Make sure to enter the correct conference title from your rights confirmation emai (ACM COMPASS '24).* ACM, New York, NY, USA, 8 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 INTRODUCTION

Air pollution stands as one of the most significant environmental health risks of our time, silently claiming over 6.7 million lives annually. In India alone, each person is annually exposed to an average of 83 µg/m^3, a value 16.6 times higher than the World Health Organization's guideline [1]. This alarming exposure contributes to approximately 70 deaths per 100,000 people, amounting to nearly a million fatalities in 2019. Fine particulate matter pollution ($PM_{2.5}$) is a primary culprit responsible for chronic obstructive pulmonary disease, lower respiratory infections, stroke, ischemic heart disease, and various cancers. Additionally, it plays a significant role in type 2 diabetes and neonatal disorders[2].

The challenges surrounding air pollution communication, however, hinder public understanding and action. Despite national guidelines and growing environmental health literacy literature, risk communication efforts are often ineffective. Moreover, a growing literature [8] on environmental health literacy suggests that communication about environmental risks must move beyond individual behaviour education to empower communities to mobilize to reduce environmental threats. Limitations in current environmental health literacy further highlight this issue. Often relied upon for information, media sources tend to present misperceptions and distortions regarding air quality risks, leading to a public disconnect from reality [2]. Moreover, entertainment media exaggerates risks, contributing to misinformation [4].

In recent years, advancements to transformer [14] based large language models (LLMs) have revolutionised information retrieval and processing. Models like GPT, BERT [3], and Llama [12] leverage the transformer architecture to capture long-range dependencies, providing sophisticated answers by processing vast contexts.

In this paper, we present our system called VayuBuddy[3] which aims to address questions about air pollution across India. VayuBuddy leverages LLMs to convert natural language user queries into Python code and then executing this code to generate answers. Our approach involves ingesting the Central Pollution Control Board (CPCB) air quality dataset, specifically focusing on daily average air quality data, such as pollutant concentrations. Additionally, the application is equipped with custom instructions to handle specific types of questions, enhancing its effectiveness in addressing diverse air pollution-related concerns.

We employed three different LLMs for code generation. We also added custom prompts (called system prompts) to ground the queries in the domain (and thus incorporate domain expertise). As an example, we mentioned the permissible pollution limits in India. We also added functions that map from pollution values to air quality index values (AQI).

We evaluate the three LLMs (LLama3, Mixtral and Gemma) on 20 curated prompts. We specifically curated these 20 prompts to capture variety of queries across: i) output type (plot/text); ii) query hardness; iii) query type (spatial, temporal, etc.). We find that LLama3 is able to generate the correct response most of the times.

We ensured code reproducibility to facilitate transparency and enable others to replicate our methodology and results. Our web application can be found https://huggingface.co/spaces/SustainabilityLabIITGN/VayuBuddy.

We believe that interfaces such as ours will lower the entry-barrier into understanding and hopefully mitigating air pollution.

2 RELATED WORK

2.1 Chatbots

Chatbots have gained significant attention in recent years. Various approaches have been explored to develop chatbots for different domains, including healthcare, customer service, education, and environmental monitoring [1]. In addition to offering real-time communication, chatbots have the potential to inform people about safety precautions, health dangers, and environmental data [6].

Our work currently treats each query as independent. In the future, we plan to develop the functionality for retaining the context across queries and thus be able to answer follow-ups.

2.2 Codegen LLMs

In numerous programming tasks, Codegen LLMs—such as OpenAI's Codex[4], a natural language machine learning model trained on billions of lines of code—perform brilliantly and frequently produce readable and accurate Python code [15]. These models have been utilized in various programming tasks, including code completion, summarisation, and translation [7].

Our current work does not presently use any codegen LLM. In the future, we believe that a sufficiently tuned codegen LLM will likely outperform a general purpose LLM.

[1] https://www.unep.org/topics/air
[2] https://www.unep.org/interactives/air-pollution-note/
[3] https://huggingface.co/spaces/SustainabilityLabIITGN/VayuBuddy
[4] https://openai.com/index/openai-codex

2.3 LLMs for tabular data

Language models (LLMs) tailored for tabular data processing have emerged as powerful tools for handling structured data in a natural language format. These models are designed to understand and generate text representations of tabular data, enabling tasks such as data summarisation, querying, and analysis. Prior research has demonstrated the effectiveness of LLMs for tabular data in domains such as finance, healthcare, and e-commerce [10, 16].

Our current work generates Python code that is executed to obtain insights from tabular data. In the future, we plan to look into tabular LLMs which bypass the code generation and execution step.

2.4 Air Quality Toolkits

Libraries, such as Vayu[5] (for Python) and OpenAir[6] (for R), help users visualise air-quality data and lower the barrier to producing meaningful insights. We believe our work is complementary to such tools.

3 DATASET

We now describe our two datasets used in this study.

3.1 Air Quality Dataset

Our air quality dataset comprises pollution measurements of $PM_{2.5}$ and PM_{10} concentrations measured in μgm^{-3}. The data is curated from the Central Pollution Control Board (CPCB) India. CPCB offers concentrations of several air pollutants, including $PM_{2.5}$, PM_{10}, SO_2, NO_2, NO, CO, and Ozone. However, $PM_{2.5}$ and PM_{10} exceed the established standards more frequently than other pollutants in India. Hence, our analysis primarily focuses on these two pollutants. We have extracted a subset of the dataset from 2017 to 2023 for analysis. The data is originally collected every 15 minutes. In total, there are 537 sensors across the country from which the data is collected. Not all the sensors collected data for this entire duration. To account for missing data within a day and to reduce the computation load, we take the average of all $PM_{2.5}$ values on a single day, across all sensors.

3.2 Natural Language Prompts Dataset

We now discuss the natural language prompts dataset that we created for evaluation. We created this prompt dataset using the following strategies:

- We included prompts *suggested by air quality experts* and those often used in various studies.
- We included prompts that *generate a specific kind of output*: text, plot.
- We included prompts based on *kind of query*: spatial plots v/s prompts to visualize a time series.
- We included prompts of *varying hardness* categorised as: easy, medium, or hard. This difficulty is decided by a poll taken among the authors regarding how much time it would take for a human to write the code for each question.

We show the 20 test prompts shown in Listing 1, of which 8 expect a response in text format, and the remaining 12 expect a plot. We explain the different categories of the prompts in Table 1. Our team then wrote the code corresponding to these queries to form the ground truth response.

Category (Count)	Example Prompt
Plot Output (12)	Plot the yearly average PM2.5.
Text Output (8)	Which city has the highest PM2.5 level in July 2022?
Spatial (11)	Which state has the highest average PM2.5?
Temporal (20)	Plot the monthly average PM2.5 of Delhi.
Raw Time (3)	Which city witnessed the lowest PM2.5?
Aggregated Time (17)	Plot the monthly average PM2.5 of Delhi.
Easy (2)	Plot the yearly average PM2.5.
Moderate (12)	Which month in which year has the highest PM2.5 overall?
Complex (6)	Plot and compare the monthly average PM2.5 of Delhi, Mumbai and Bengaluru for the year 2022.

Table 1: Example prompts. These are some examples of the test prompts for each of the categories, along with the count of the occurrence of that category.

Listing 1: Test prompts with categories

```
P1: 'Plot the monthly average PM2.5 for the year
2023.' (Plot Output, Temporal, Aggregated Time,
Easy)
P2: 'Which month in which year has the highest
average PM2.5 overall?' (Text Output, Temporal,
Aggregated Time, Moderate)
P3: 'Which month in which year has the highest PM2
.5 overall?' (Text Output, Temporal, Raw Time,
Moderate)
P4: 'Which month has the highest average PM2.5 in
2023 for Mumbai?' (Text Output, Spatial, Temporal,
Aggregated Time, Moderate)
P5: 'Plot and compare monthly time-series of
pollution for Mumbai and Bengaluru.' (Plot Output,
Spatial, Temporal, Aggregated Time, Complex)
P6: 'Plot the yearly average PM2.5.' ('Plot Output,
 Temporal, Aggregated Time, Easy)
```

[5] https://sustainability-lab.github.io/vayu/
[6] https://www.openair.com

```
P7: 'Plot the monthly average PM2.5 of Delhi,
Mumbai and Bengaluru for the year 2022.' (Plot
Output, Spatial, Temporal, Aggregated Time,
Moderate)
P8: 'Which month has the highest pollution?' (Text
Output, Temporal, Raw Time, Complex)
P9: 'Which city has the highest PM2.5 level in July
 2022?' (Text Output, Spatial, Temporal, Raw Time,
Moderate)
P10: 'Plot and compare monthly time-series of PM2.5
 for Mumbai and Bengaluru.' (Plot Output, Spatial,
Temporal, Aggregated Time, Moderate)
P11: 'Plot and compare the monthly average PM2.5 of
 Delhi, Mumbai and Bengaluru for the year 2022.' (
Plot Output, Spatial, Temporal, Aggregated Time,
Complex)
P12: 'Plot the monthly average PM2.5.' (Plot Output
, Temporal, Aggregated Time, Moderate)
P13: 'Plot the monthly average PM10 for the year
2023.' (Plot Output, Temporal, Aggregated Time,
Moderate)
P14: 'Which (month, year) has the highest PM2.5?' (
Text Output, Temporal, Aggregated Time, Complex)
P15: 'Plot the monthly average PM2.5 of Delhi for
the year 2022.' (Plot Output, Spatial, Temporal,
Aggregated Time, Moderate)
P16: 'Plot the monthly average PM2.5 of Bengaluru
for the year 2022.' (Plot Output, Spatial, Temporal
, Aggregated Time, Moderate)
P17: 'Plot the monthly average PM2.5 of Mumbai for
the year 2022.' (Plot Output, Spatial, Temporal,
Aggregated Time, Moderate)
P18: 'Which state has the highest average PM2.5?' (
Text Output, Spatial, Temporal, Aggregated Time,
Complex)
P19: 'Plot monthly PM2.5 in Gujarat for 2023.' (
Plot Output, Spatial, Temporal, Aggregated Time,
Moderate)
P20: 'What is the name of the month with the
highest average PM2.5 overall?' (Text Output,
Temporal, Aggregated Time, Complex)
```

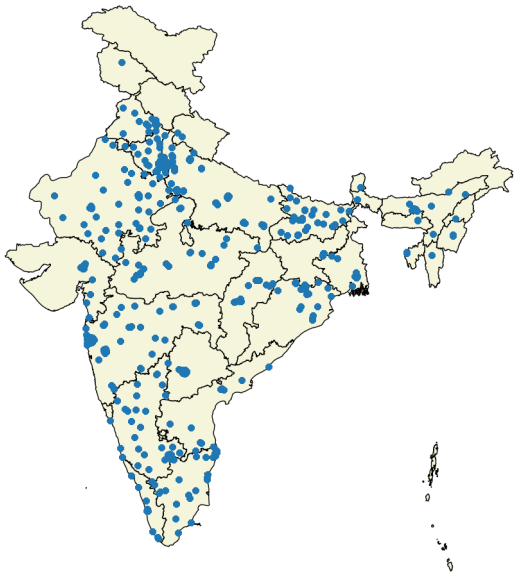

Figure 2: Location of Sensors: This image was generated by VayuBuddy with the following prompt: "Plot the locations of the stations on the India Map. Do not Annotate."

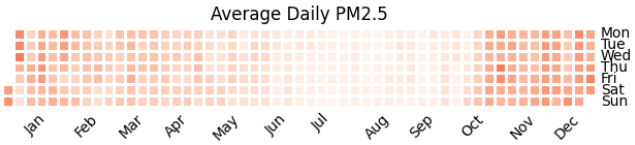

Figure 3: This image was generated by VayuBuddy with the following prompt: "Create a calendar map showing average PM2.5."

4 METHODOLOGY

Our application VayuBuddy is designed to return answers to natural language queries about air pollution. Figure 1 shows our main interface. We developed VayuBuddy's interface in Streamlit[7] and we use GroqCloud[8] for LLM support/assistance.

4.1 Overall flow

Once a user provides a prompt (either custom or from one of our suggested prompts) and chooses a LLM, our system generates Python code that can operate on the air quality dataset. A runner module then executes the generated code and provides the answer in the UI. However, occasionally VayuBuddy runs into an error while generating or executing the code. Therefore, to increase the robustness of the application, we try each prompt five times before confirming the failure.

We now showcase three non-trivial prompts (Figure 2, Figure 3, and Figure 4) and their corresponding plots generated correctly by VayuBuddy. These show the richness of the capabilities. Not all of our team members responsible for creating and evaluating prompts were able to timely implement such plots. Thus, we did not include them in our 20 plots for evaluation and would consider these to be in the "very-hard" category.

[7] https://streamlit.io
[8] https://console.groq.com

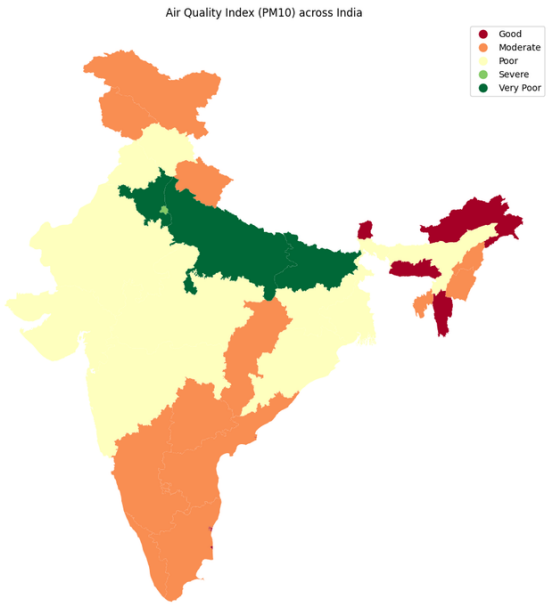

Figure 4: This image was generated by VayuBuddy with the following prompt: "Plot the choropleth map showing PM10 levels across India, with different colours representing different AQI categories (e.g., Good, Moderate, Unhealthy, etc.)"

4.2 LLM models

We compare three models: Llama3 [13], Mixtral [5], and Gemma [11]. The specific versions of these models are: llama3-70b-8192, mixtral-8x7b-32768 and gemma-7b-it, respectively. We use the LLama3 LLM as the default in our application.

- **llama3-70b-8192**: Llama3 features a model size of 70 billion parameters with a context window of 8192 tokens. Significant improvements in architecture include a tokenizer with a vocabulary of 128K tokens for more efficient language encoding and the adoption of grouped query attention (GQA) to enhance inference efficiency. The models are trained on sequences of 8192 tokens with masking to prevent self-attention from crossing document boundaries.
- **mixtral-8x7b-32768**: This is a sparse mixture-of-experts [9] network with a model size of 47 billion parameters and a context window of 32768 tokens. It utilizes only 13 billion active parameters during inference since it is a sparse mixture of networks. The model consists of experts, which are feed-forward neural networks, and a gate network or router that determines the routing of tokens to different experts. This router is pre-trained alongside the rest of the network.
- **gemma-7b-it**: A model with the size of 7 billion parameters and a context window of 8192 tokens adopts key improvements such as Multi-Query Attention, RoPE Embeddings, GeGLU Activations, and RMSNorm. The training process begins with supervised fine-tuning on a mix of text-only, English-only synthetic, and human-generated prompt-response pairs, followed by reinforcement learning from human feedback (RLHF). The reward model is trained on labelled English-only preference data, and the policy is based on a set of high-quality prompts.

4.3 Infusing domain expertise

It is important to note that the LLM models perform poorly out of the box. We incorporate various domain insights and experiences into our customisations to the system in two ways: i) system prompts; ii) predefined functions and additional metadata.

4.3.1 System Prompts. We develop custom instructions (henceforth called system prompts) that are prepended to the prompt chosen by the user. These include the guidelines for $PM_{2.5}$ emissions in India and by WHO. These also include specific instructions to help the LLM direct the response as a plot or text. The full set of system prompts are described in Listing 2.

4.3.2 Predefined Functions and Additional Metadata. We also include functions that are always included in the generated code (and thus can be leveraged by generated code). These include a function to create an air quality index (AQI) value given the different pollutant values. This is a country specific encoding. We found that including the pollutant ranges to AQI values mapping via function works better than by including these details in the system prompt. Similarly, we included some code and metadata corresponding to geographical shapefiles for India.

Listing 2: System Prompts. Additional information and direction given to the LLM.

```
- The columns are 'Timestamp', 'station', 'PM2.5',
'PM10', 'address', 'city', 'latitude', 'longitude',
and 'state'.
- Frequency of data is daily.
- `pollution` generally means `PM2.5`.
- You already have df, so don't read the csv file
- Don't print anything, but save result in a
variable `answer` and make it global.
- Unless explicitly mentioned, don't consider the
result as a plot.
- PM2.5 guidelines: India: 60, WHO: 15.
- PM10 guidelines: India: 100, WHO: 50.
- If result is a plot, show the India and WHO
guidelines in the plot.
- If result is a plot make it in tight layout, save
 it and save path in `answer`. Example: `answer='
plot.png'`
- If result is a plot, rotate x-axis tick labels by
 45 degrees.
- If result is not a plot, save it as a string in `
answer`. Example: `answer='The city is Mumbai'`
- I have a geopandas.geodataframe india
containining the coordinates required to plot
Indian Map with states.
```

```
- If the query asks you to plot on India Map, use
that geodataframe to plot and then add more points
as per the requirements using the similar code as
follows : v = ax.scatter(df['longitude'], df['
latitude']). If the colorbar is required, use the
following code : plt.colorbar(v)
- If the query asks you to plot on India Map plot
the India Map in Beige color
- Whenever you do any sort of aggregation, report
the corresponding standard deviation, standard
error and the number of data points for that
aggregation.
- Whenever you're reporting a floating point number
, round it to 2 decimal places.
- Always report the unit of the data. Example: The
average PM2.5 is 45.67 μgm^-3.
- Try to avoid mentioning time in the plot. If you
have to, use the 24-hour format.
```

5 EVALUATION

We now evaluate the efficacy of the different LLMs on the different prompts.

5.1 Experimental Setup

We ran each LLM model 5 times on each prompt. We measured the correctness in the following manner: A score of 1 is assigned if the response obtained is correct. Any other response is treated as incorrect and scored 0.

There are four possible ways for a response to be considered incorrect:

(1) LLM failing to generate the code.
(2) The LLM generates the code, but the code gives error while executing.
(3) The LLM generated code runs without errors but gives a wrong answer.
(4) The response from the LLM appears in a different format. Example: When VayuBuddy responds with a plot instead of a text-based answer.

The web application is hosted on HuggingFace. We use all the default configurations for the LLMs, including, but not limited to, cache, callbacks, timeout, and max_tokens, except temperature, which is set to 0.

5.2 Results and Analysis

Table 2 shows that VayuBuddy (with LLama3 as the default) can respond correctly to most prompts. We can also note that Llama3 outperforms the other models, Mixtral and Gemma, by a considerable margin. We believe we get superior results because Llama3 is trained on more recent data and has significantly more parameters. Next, we analyse the performance across the different prompts and different LLMs and the reasons for their efficacy. First, in Table 3, we compare two categories of prompts: "Plot Output" and "Text Output". We specifically check if a model is able to get the correct answer at least 4 out of 5 times. We can note that both Mixtral and Gemma struggle more with the plot-based prompts. Next, we note that all three models perform poorly regarding the test prompt P8: "Which month has the highest pollution?". We found that the LLama3 LLM would sometimes get the correct answer but other times provide a plot instead, whereas the other two LLMs generally return a plot.

Prompt	Llama3	Mixtral	Gemma	Average Across LLMs
P1	**5**	0	0	1.66
P2	3	**4**	0	2.33
P3	4	**5**	0	3.00
P4	**5**	0	**5**	3.33
P5	**5**	0	0	1.66
P6	**5**	0	4	3.00
P7	**4**	0	0	1.33
P8	**2**	0	0	0.66
P9	**5**	1	0	2.00
P10	**5**	0	0	1.66
P11	**5**	0	0	1.66
P12	**5**	**5**	0	3.33
P13	**5**	0	**5**	3.33
P14	**5**	**5**	0	3.33
P15	**5**	0	0	1.66
P16	**5**	0	0	1.66
P17	**5**	4	0	3.00
P18	**5**	4	**5**	**4.66**
P19	**5**	0	0	1.66
P20	**5**	1	**5**	3.66
Average Across Prompts	**4.65**	1.45	1.20	

Table 2: Performance of each model on each test prompt. We performed 5 trials for each prompt for each model and calculated the performance as the number of times we got the correct response. Among the three, Llama3 gives the best results.

Category	Llama3	Mixtral	Gemma
Plot Output (out of 12)	12	2	2
Text Output (out of 8)	6	4	3
Total	18	6	5

Table 3: Comparison between output categories when the model responds to each prompt correctly at least 4 times.

6 LIMITATIONS AND FUTURE WORK

- **Expanding Data Sources:** We aim to enhance VayuBuddy's capabilities to additionally answer queries based on text inputs. These could include various advisories from pollution control boards.
- **Extending Feature Categories:** We plan to include more categories of features such as other pollutants like SO_2, NO_2, NO, CO, and Ozone. Along with that, we plan to add important meteorological parameters such as wind (direction and speed).
- **Leveraging Advanced Language Models for Code Generation:** Exploring emerging LLMs tailored for code generation represents another avenue for advancing VayuBuddy's capabilities. Experimenting with specialized LLMs like GitHub Copilot or Starcoder could enhance VayuBuddy's proficiency in generating code snippets with reduced errors and improved efficiency.
- **Enhancing Prompting Methods:** Furthermore, delving into advanced prompting methods presents an opportunity to enhance VayuBuddy's capabilities. While the platform currently relies on Zero Shot Prompting, which involves prompting the model without specific examples, other strategies like Chain of Thought, Tree of Thought, and React Prompting merit exploration. These approaches encourage deeper consideration before generating responses, potentially refining VayuBuddy's ability to offer nuanced insights on air quality-related queries.
- **Implementing Active Learning Strategies:** Another innovative approach for enhancing VayuBuddy's capabilities is exploring active learning strategies. By integrating active learning methodologies, VayuBuddy can intelligently select a set of prompts, optimising the training process and improving model performance over time. Active learning techniques can enhance VayuBuddy's adaptability to new data and user interactions.
- **Automating Library Installation:** To streamline the user experience and enhance the autonomy of VayuBuddy, a methodology for automated library installation could be developed. Instead of requiring users/us to manually install libraries and provide them to the model, VayuBuddy could autonomously identify the necessary libraries based on the user's query and install them as needed. This solution would simplify the user interaction process and improve the efficiency and accessibility of VayuBuddy, allowing users to seamlessly access air quality insights without the burden of manual setup.

7 CONCLUSION

In this work, we explored our system VauBuddy to respond to natural language queries pertaining to air pollution based on data from CPCB. We believe that a system such as ours can be used by various stakeholders including but not limited to pollution control boards, researchers. Such systems put the focus back on the problem or the question being asked rather than the engineering efforts towards solving the query to extract the data. In our initial discussions with various air quality experts, the response has been positive. We plan to roll this to more stakeholders.

REFERENCES

[1] Lara Christoforakos, Nina Feicht, Simone Hinkofer, Annalena Löscher, Sonja F Schlegl, and Sarah Diefenbach. 2021. Connect with me. exploring influencing factors in a human-technology relationship based on regular chatbot use. *Frontiers in digital health* 3 (2021), 689999.

[2] Ricardo Cisneros, Paul Brown, Linda Cameron, Erin Gaab, Mariaelena Gonzalez, Steven Ramondt, David Veloz, Anna Song, and Don Schweizer. 2017. Understanding public views about air quality and air pollution sources in the San Joaquin Valley, California. *Journal of Environmental and Public Health* 2017 (2017).

[3] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. arXiv:1810.04805 [cs.CL]

[4] Christopher Frayling. 2013. *Mad, bad and dangerous?: the scientist and the cinema.* Reaktion books.

[5] Albert Q. Jiang, Alexandre Sablayrolles, Antoine Roux, Arthur Mensch, Blanche Savary, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Emma Bou Hanna, Florian Bressand, Gianna Lengyel, Guillaume Bour, Guillaume Lample, Lélio Renard Lavaud, Lucile Saulnier, Marie-Anne Lachaux, Pierre Stock, Sandeep Subramanian, Sophia Yang, Szymon Antoniak, Teven Le Scao, Théophile Gervet, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. 2024. Mixtral of Experts. arXiv:2401.04088 [cs.LG]

[6] M.N Nuha, G.K.N.P Wishvajith, Amitha Caldera, N.T Weerasinghe, G.S Weeratunga, and Shashika Lokuliyana. 2023. Predicting the Air Pollution Level and Creating an Awareness Platform. In *2023 5th International Conference on Advancements in Computing (ICAC)*. 137–142. https://doi.org/10.1109/ICAC60630.2023.10417387

[7] Saurabh Pujar, Luca Buratti, Xiaojie Guo, Nicolas Dupuis, Burn Lewis, Sahil Suneja, Atin Sood, Ganesh Nalawade, Matthew Jones, Alessandro Morari, and Ruchir Puri. 2023. Automated Code generation for Information Technology Tasks in YAML through Large Language Models. arXiv:2305.02783 [cs.SE]

[8] Van Bogart K Perez-Zuniga R. Ramírez AS, Ramondt S. [n. d.]. Public Awareness of Air Pollution and Health Threats: Challenges and Opportunities for Communication Strategies To Improve Environmental Health Literacy. ([n. d.]). https://doi.org/10.1080/10810730.2019.1574320

[9] Omar Sanseviero, Lewis Tunstall, Philipp Schmid, Sourab Mangrulkar, Younes Belkada, and Pedro Cuenca. 2023. Mixture of Experts Explained. https://huggingface.co/blog/moe

[10] Yuan Sui, Mengyu Zhou, Mingjie Zhou, Shi Han, and Dongmei Zhang. 2024. Table Meets LLM: Can Large Language Models Understand Structured Table Data? A Benchmark and Empirical Study. In *Proceedings of the 17th ACM International Conference on Web Search and Data Mining* (<conf-loc>, <city>Merida</city>, <country>Mexico</country>, </conf-loc>) *(WSDM '24)*. Association for Computing Machinery, New York, NY, USA, 645–654. https://doi.org/10.1145/3616855.3635752

[11] Gemma Team, Thomas Mesnard, Cassidy Hardin, Robert Dadashi, Surya Bhupatiraju, Shreya Pathak, Laurent Sifre, Morgane Rivière, Mihir Sanjay Kale, Juliette Love, Pouya Tafti, Léonard Hussenot, Pier Giuseppe Sessa, Aakanksha Chowdhery, Adam Roberts, Aditya Barua, Alex Botev, Alex Castro-Ros, Ambrose Slone, Amélie Héliou, Andrea Tacchetti, Anna Bulanova, Antonia Paterson, Beth Tsai, Bobak Shahriari, Charline Le Lan, Christopher A. Choquette-Choo, Clément Crepy, Daniel Cer, Daphne Ippolito, David Reid, Elena Buchatskaya, Eric Ni, Eric Noland, Geng Yan, George Tucker, George-Christian Muraru, Grigory Rozhdestvenskiy, Henryk Michalewski, Ian Tenney, Ivan Grishchenko, Jacob Austin, James Keeling, Jane Labanowski, Jean-Baptiste Lespiau, Jeff Stanway, Jenny Brennan, Jeremy Chen, Johan Ferret, Justin Chiu, Justin Mao-Jones, Katherine Lee, Kathy Yu, Katie Millican, Lars Lowe Sjoesund, Lisa Lee, Lucas Dixon, Machel Reid, Maciej Mikuła, Mateo Wirth, Michael Sharman, Nikolai Chinaev, Nithum Thain, Olivier Bachem, Oscar Chang, Oscar Wahltinez, Paige Bailey, Paul Michel, Petko Yotov, Rahma Chaabouni, Ramona Comanescu, Reena Jana, Rohan Anil, Ross McIlroy, Ruibo Liu, Ryan Mullins, Samuel L Smith, Sebastian

Borgeaud, Sertan Girgin, Sholto Douglas, Shree Pandya, Siamak Shakeri, Soham De, Ted Klimenko, Tom Hennigan, Vlad Feinberg, Wojciech Stokowiec, Yu hui Chen, Zafarali Ahmed, Zhitao Gong, Tris Warkentin, Ludovic Peran, Minh Giang, Clément Farabet, Oriol Vinyals, Jeff Dean, Koray Kavukcuoglu, Demis Hassabis, Zoubin Ghahramani, Douglas Eck, Joelle Barral, Fernando Pereira, Eli Collins, Armand Joulin, Noah Fiedel, Evan Senter, Alek Andreev, and Kathleen Kenealy. 2024. Gemma: Open Models Based on Gemini Research and Technology. arXiv:2403.08295 [cs.CL]

[12] Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, Aurelien Rodriguez, Armand Joulin, Edouard Grave, and Guillaume Lample. 2023. LLaMA: Open and Efficient Foundation Language Models. arXiv:2302.13971 [cs.CL]

[13] Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, Dan Bikel, Lukas Blecher, Cristian Canton Ferrer, Moya Chen, Guillem Cucurull, David Esiobu, Jude Fernandes, Jeremy Fu, Wenyin Fu, Brian Fuller, Cynthia Gao, Vedanuj Goswami, Naman Goyal, Anthony Hartshorn, Saghar Hosseini, Rui Hou, Hakan Inan, Marcin Kardas, Viktor Kerkez, Madian Khabsa, Isabel Kloumann, Artem Korenev, Punit Singh Koura, Marie-Anne Lachaux, Thibaut Lavril, Jenya Lee, Diana Liskovich, Yinghai Lu, Yuning Mao, Xavier Martinet, Todor Mihaylov, Pushkar Mishra, Igor Molybog, Yixin Nie, Andrew Poulton, Jeremy Reizenstein, Rashi Rungta, Kalyan Saladi, Alan Schelten, Ruan Silva, Eric Michael Smith, Ranjan Subramanian, Xiaoqing Ellen Tan, Binh Tang, Ross Taylor, Adina Williams, Jian Xiang Kuan, Puxin Xu, Zheng Yan, Iliyan Zarov, Yuchen Zhang, Angela Fan, Melanie Kambadur, Sharan Narang, Aurelien Rodriguez, Robert Stojnic, Sergey Edunov, and Thomas Scialom. 2023. Llama 2: Open Foundation and Fine-Tuned Chat Models. arXiv:2307.09288 [cs.CL]

[14] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2023. Attention Is All You Need. arXiv:1706.03762 [cs.CL]

[15] Michel Wermelinger. 2023. Using GitHub Copilot to Solve Simple Programming Problems. In *Proceedings of the 54th ACM Technical Symposium on Computer Science Education V. 1* (<conf-loc>, <city>Toronto ON</city>, <country>Canada</country>, </conf-loc>) *(SIGCSE 2023)*. Association for Computing Machinery, New York, NY, USA, 172–178. https://doi.org/10.1145/3545945.3569830

[16] Yazheng Yang, Yuqi Wang, Sankalok Sen, Lei Li, and Qi Liu. 2024. Unleashing the Potential of Large Language Models for Predictive Tabular Tasks in Data Science. arXiv:2403.20208 [cs.LG]