AI Project

Pollution Awareness in Brussels

2025-01-13

J. Hermans
ECAM

Contents

github repository : https://github.com/Jordan-92/langchain_project.git

I. Introduction

Air pollution poses significant health and environmental challenges in urban areas worldwide, and Brussels is no exception. This project leverages artificial intelligence (AI) to address the issue of air quality monitoring and prediction by combining advanced data collection, machine learning models, and a user-friendly interface. The main goal of the system is to provide actionable insights into pollution levels, enabling individuals, particularly those with respiratory vulnerabilities, to take informed precautions.

By predicting the levels of PM10 and PM2.5 particles based on meteorological data, the system empowers users to anticipate and mitigate the health risks associated with air pollution. Additionally, the integration of LangChain-based tools extends the system's utility by providing users with information on sustainable urban initiatives, promoting awareness and engagement in environmentally friendly practices.

II. Supporting the Sustainable Development Goals (SDG)

II.1. Introduction

This project aligns with SDG 3: Good Health and Well-being and SDG 11: Sustainable Cities and Communities. By providing insights into pollution levels in Brussels and enabling proactive measures for lung-sensitive individuals, it addresses public health concerns and contributes to sustainable urban living.

II.2. Sustainable Development Goal (SDG) Chosen

1. Within SDG 3: Good Health and Well-being, this project aligns specifically with Target 3.9:

   "Substantially reduce the number of deaths and illnesses from hazardous chemicals and air, water, and soil pollution and contamination by 2030."

   It supports this target by:

   - Providing Pollution Predictions: The prediction tool helps individuals, especially those with respiratory vulnerabilities, take precautionary measures to avoid exposure to high pollution levels.
   - Raising Awareness: By offering accessible information on environmental initiatives, the project educates communities on reducing pollution-related health risks.

2. This project would also be SDG 11: Sustainable Cities and Communities, specifically Target 11.6:

   "Reduce the adverse environmental impact of cities, including by paying special attention to air quality and municipal and other waste management."

   It directly addresses this target by:

   - Monitoring Air Quality: Providing predictions of pollution levels helps mitigate the impact of air quality issues on sensitive populations.
   - Promoting Sustainable Practices: By gathering and sharing information on eco-friendly initiatives, the system supports local efforts to improve environmental conditions.
   - This aligns closely with the urban focus of SDG 11, making it a precise and relevant choice for your project's impact framework.

II.3. Supporting Need and Impact

Urban air quality is an urgent concern. In Belgium, several cities have struggled to meet EU air quality standards. According to a 2020 report by the Belgian Interregional Environment Agency, concentrations of harmful pollutants like nitrogen dioxide and particulate matter frequently exceed safe levels in major urban areas. By analyzing pollution data alongside information on urban projects, authorities can make evidence-based decisions to mitigate pollution.

Source: https://www.eufje.org/images/docConf/visio2020/REPORT_BELGIUM.pdf

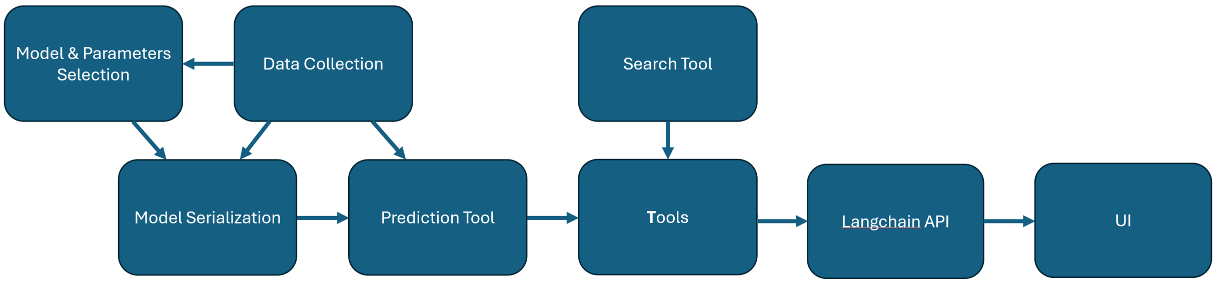
II.4. Discussion

The integration of AI and predictive analytics could further enhance Belgium's air quality management, especially by making the government and public aware of underrepresented high-risk zones.

III. Technical Description of the System

III.1. Overview of the System

This project can be separated into data collection processes, analysis for comparing and fine-tuning different machine learning models, model serialization process, tools for the langchain, api using langchain and UI.



There is also a utils.py script providing a set of utility functions for many of the above-mentioned parts of the project.

III.2. Data Collection Process

The process for gathering data for this LangChain-based project combines the retrieval of historical air quality data and weather data.

Air Quality & Weather Data Retrieval

The get_historical_pm10_pm25.py script focuses on fetching air quality data (PM10 and PM2.5) for specific stations and communes. And the get_weather_data.py script retrieves weather forecasts and historical data to complement air quality information, essential for making accurate pollution predictions.

Key Features:

1. API Integration: The system uses APIs (airnet.waqi.info & VisualCrossing Weather API) to gather historical data for air quality pollutants (PM10 and PM2.5) from monitoring stations and a variety of meteorological metrics.
2. Multi-Station Data Aggregation: (in Air Quality Data): Results are combined into a multi-index DataFrame, with each station's data labeled for distinction.
3. Output Storage: Processed data is saved as JSON files for each commune, enabling easy reuse and scalability. JSON files are in orient=split to avoid repetition and thus reduce file size.
4. Merge with existing data to obtain even more data (in Weather Data)
5. Error Handling

Configuration Management

A YAML configuration file enables flexibility and scalability:

- Enable parameter: It allows to avoid starting a process if you've forgotten to set the configuration parameters.
- Commune Selection: The communes list defines which areas are processed.
- Date Settings: first_day and last_day allow for dynamic date ranges for weather data.
- This modular approach ensures that the system is adaptable to new communes or extended analysis periods.

Discussion

These data retrieval processes form the foundation of the LangChain project. By leveraging structured APIs and robust handling of edge cases, the system ensures:

- Reliability: Continuous availability of clean, merged data for predictions.
- Scalability: Ability to add new communes or extend date ranges without major code modifications.
- Flexibility: Modular design using YAML configurations to adapt to varying requirements.

III.3. Model comparison and fine-tuning

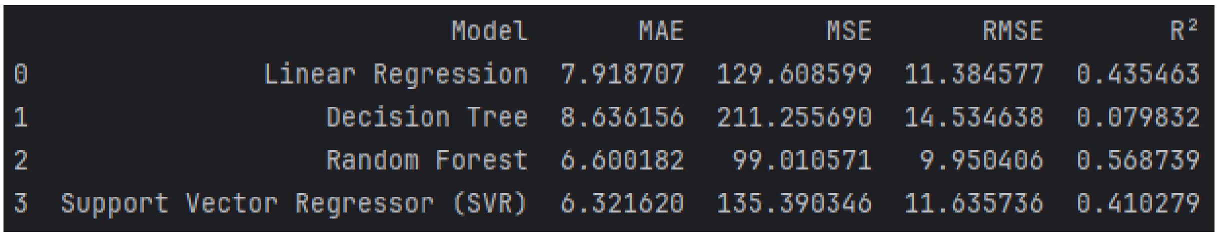
The project evaluates and compares multiple machine learning and deep learning models to predict air pollution levels (PM10 and PM2.5) in the Brussels region based on meteorological data. Below is an in-depth explanation of the methodology and results.

Model comparisons

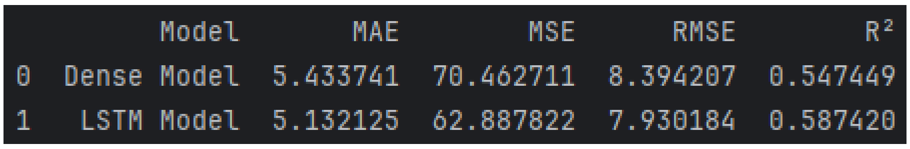
The machine_learning_comparison.py and deep_learning_comparison.py script evaluates classical models on their ability to predict PM10 levels.

1. Data Preprocessing:
   - Data for PM10 and PM2.5 is merged with weather data using utility functions.
   - Input features include weather parameters(temperature, humidity, precipitation, snow, wind speed, pressure, UV index, and moon phase) and temporal parameters(year, month, day of week).
   - Data is split into training (80%) and testing (20%) sets using train_test_split.
2. Model Selection: The following models are compared:
   - Linear Regression: A baseline linear model.
   - Decision Tree Regressor: A tree-based model for capturing non-linear relationships.
   - Random Forest Regressor: An ensemble of decision trees to improve robustness.
   - Support Vector Regressor (SVR): A kernel-based regression method.
   - Dense Model: A fully connected feed-forward neural network. Suitable for general-purpose predictions.
   - LSTM Model: A recurrent neural network designed for time-series or sequential data. Captures temporal dependencies better than a Dense Model.
3. Scaling:
   - StandardScaler is used to normalize features for the SVR, Dense and LSTM model to optimize performance.
4. Evaluation Metrics:
   - Mean Absolute Error (MAE): Measures the average absolute difference between true and predicted values.
   - Mean Squared Error (MSE): Penalizes larger errors more heavily, providing a quadratic perspective on prediction accuracy.
   - Root Mean Squared Error (RMSE): The square root of MSE, representing error in the same unit as the target variable.
   - R^2 (Coefficient of Determination): Indicates the proportion of variance in the target variable that the model explains. Values closer to 1 are better.

Results Summary



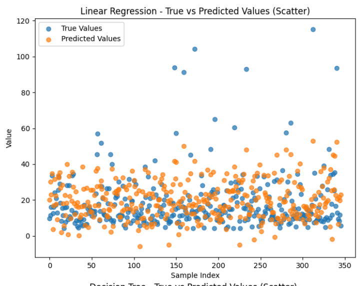
```
                           Model       MAE         MSE       RMSE        R²
0              Linear Regression  7.918707  129.608599  11.384577  0.435463
1                  Decision Tree  8.636156  211.255690  14.534638  0.079832
2                  Random Forest  6.600182   99.010571   9.950406  0.568739
3  Support Vector Regressor (SVR)  6.321620  135.390346  11.635736  0.410279
```



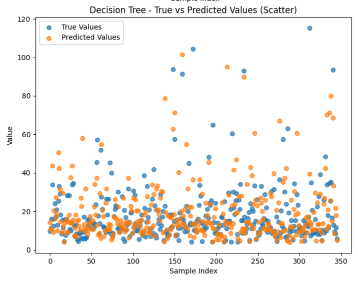
```
         Model       MAE        MSE      RMSE        R²
0  Dense Model  5.433741  70.462711  8.394207  0.547449
1   LSTM Model  5.132125  62.887822  7.930184  0.587420
```

Key Observations

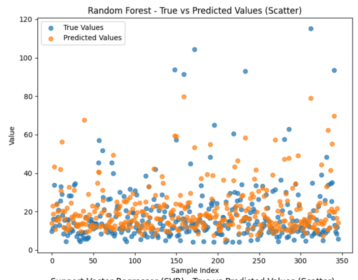
1. Linear Regression:
   - Moderate performance with an R^2 of 0.435.
   - Performs well on simpler relationships but may struggle with non-linearity in the dataset.
   - Predicted values follow the trend of true values but with noticeable deviations for higher PM10 values.



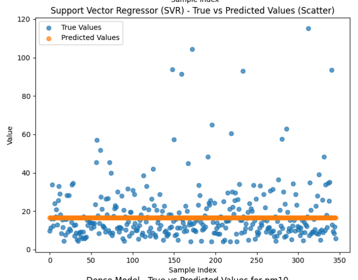
2. Decision Tree:
   - Poor performance with an $R^2 = 0.0798$.
   - Likely overfitted due to a lack of regularization, resulting in poor generalization.
   - Predictions show significant scatter, indicating poor generalization and overfitting.



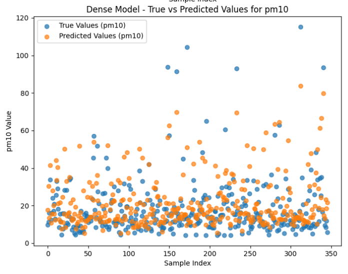
3. Random Forest:
   - Good performance with a R^2 of 0.569.
   - The ensemble approach reduces overfitting and captures complex relationships effectively.
   - Predictions align closely with true values, demonstrating its ability to capture patterns in the data.



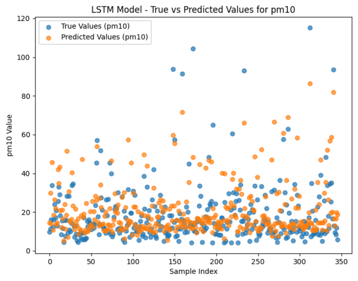
4. Support Vector Regressor:
   - Moderate performance with an R^2 of 0.410.
   - Suitable for smaller datasets, but its performance is slightly behind Random Forest due to the dataset's size and complexity.
   - Predictions appear to be consistent. Seems to just set all predictions to the mean value.



5. Support Vector Regressor:
   - Good performance with an R^2 of 0.547.
   - Robust for straightforward, non-sequential relationships.
   - Captures general trends but struggles with high PM10 values, evident from the wider scatter for higher ranges.



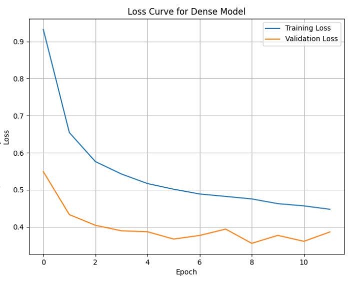
6. Support Vector Regressor:
   - Best performance with the lowest MSE (62.89) and the highest R^2 (0.587).
   - Handles sequential dependencies, making it better suited for time-series data or data with temporal patterns.
   - Captures general trends but struggles with high PM10 values, evident from the wider scatter for higher ranges.

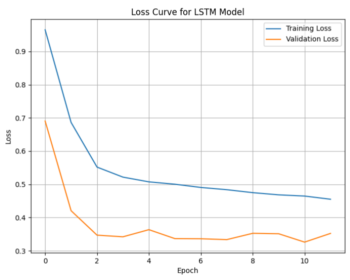


Training Dynamics

Loss Curves:

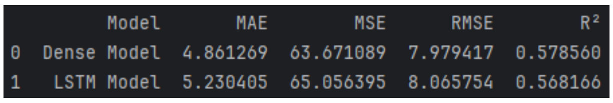
- Both models exhibit smooth convergence, with the training and validation losses decreasing steadily.





Test without time features

I retested my deep learning models without the time features (year, month, day_of_week).



```
        Model        MAE        MSE      RMSE        R²
0  Dense Model  4.861269  63.671089  7.979417  0.578560
1   LSTM Model  5.230405  65.056395  8.065754  0.568166
```

We can observe that the inclusion of time features considerably improves LSTM performance.The LSTM model leverages these sequential features more effectively due to its recurrent architecture.

Discussion

The LSTM model is the winner due to its ability to model temporal dependencies effectively, which is critical when predicting PM10 based on features like year, month, and day_of_week. However, the Random Forest remains a viable alternative for simpler, non-temporal tasks or when computational efficiency is a priority.

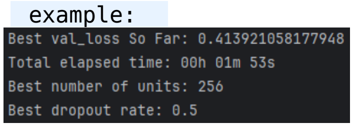
III.4. Creating the Pickle File for the Best Model

The model_tuning_and_serialization.py script is designed to train and tune an LSTM model for predicting PM10 and PM2.5 levels using hyperparameter optimization with Keras Tuner's RandomSearch.

Key Features

1. Dynamic Model Creation with Tuning:
   - The create_lstm_model function dynamically adjusts model architecture based on hyperparameters passed by the tuner:
   - Units in the LSTM layer: Tuned between 32 and 256 in steps of 32.
   - Dropout Rate: Tuned between 0.1 and 0.5 in steps of 0.1.

`example:`



```
Best val_loss So Far: 0.413921058177948
Total elapsed time: 00h 01m 53s
Best number of units: 256
Best dropout rate: 0.5
```

2. The architecture includes:
   - LSTM layer with relu activation.
   - Dropout for regularization.
   - Dense layers for regression output.
3. Hyperparameter Tuning:
   - The RandomSearch tuner explores combinations of the number of LSTM units and dropout rates.
   - Objective: Minimize validation loss.
   - Trials: 10 trials, each averaged over 2 executions.
4. Scalable Data Preprocessing:
   - Scales both input (X) and target (y) features using StandardScaler.
   - Reshapes input data for compatibility with LSTM models ((samples, time_steps, features)).
5. EarlyStopping Callback:
   - Stops training when validation loss stops improving for 5 consecutive epochs, saving computational resources and preventing overfitting.
6. Model Persistence:
   - The best model and associated scalers (scaler_X and scaler_y) are saved to a .pkl file for deployment and future predictions.
7. Flexible Configurations:
   - Leverages configuration files (config.yaml and communes.yaml) to dynamically adapt to different communes and stations.

III.5. LangChain-Based API with Toolchain Agent

The API_langchain.py and tools.py scripts implement a Flask-based API that integrates LangChain agents with custom tools for responding to user queries. This API is the backbone for answering questions about air quality using a toolchain approach.

Workflow of the API

I. Initialization

1. Environment Configuration
2. LangChain Agent Setup
   - The ChatOpenAI model (gpt-3.5-turbo) is initialized with a low temperature setting to ensure consistent, deterministic responses.
   - Tools are registered with the agent using LangChain's create_tool_calling_agent.
3. Flask Application:
   - A Flask app is initialized with CORS enabled to allow communication from the frontend.

II. Agent Execution

- The LangChain agent is wrapped in an AgentExecutor for structured invocation of tools based on user queries.

III. Handling User Queries

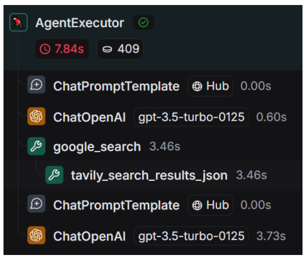
- The /ask endpoint processes GET requests:
  1. Validates the query parameter question.
  2. Passes the query to the agent executor via invoke.
  3. Returns the agent's structured response in JSON format.

Tools for LangChain Agent

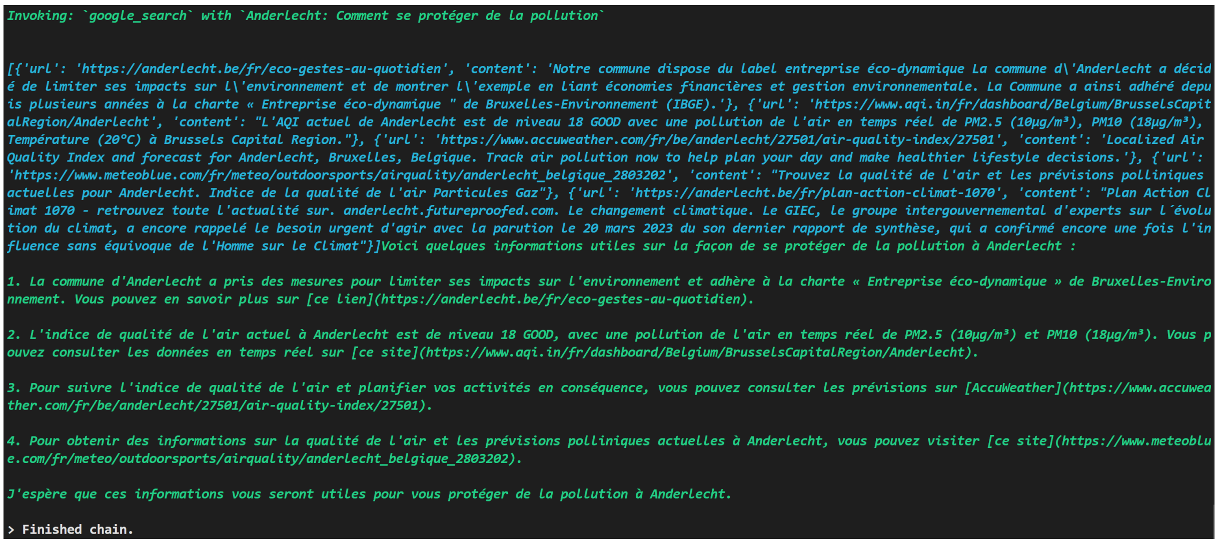
Two tools are defined in tools.py and registered with the LangChain agent. These tools enhance the agent's capabilities to provide targeted answers.

I. Google Search Tool `name="google_search"`

- Uses the TavilySearchResults integration to fetch information about pollution.
- Ideal for answering general questions or retrieving external data from the web.



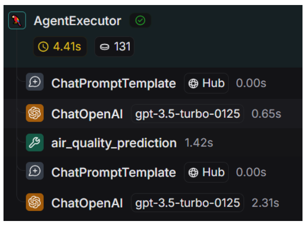
```
AgentExecutor
7.84s  409
ChatPromptTemplate  Hub  0.00s
ChatOpenAI  gpt-3.5-turbo-0125  0.60s
google_search  3.46s
    tavily_search_results_json  3.46s
ChatPromptTemplate  Hub  0.00s
ChatOpenAI  gpt-3.5-turbo-0125  3.73s
```



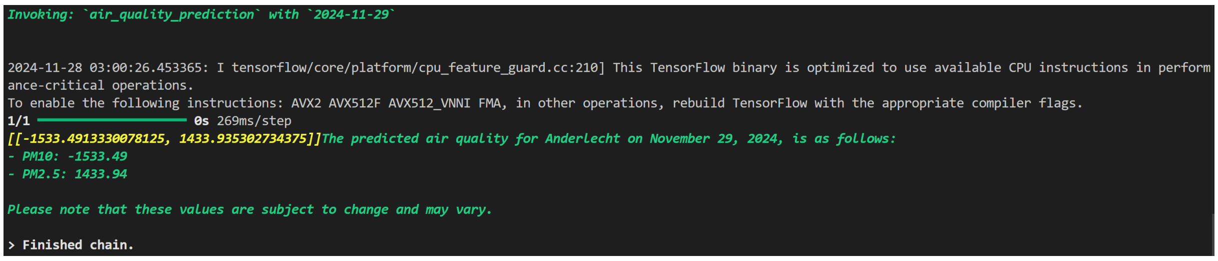
```
Invoking: `google_search` with `Anderlecht: Comment se protéger de la pollution`


[{'url': 'https://anderlecht.be/fr/eco-gestes-au-quotidien', 'content': 'Notre commune dispose du label entreprise éco-dynamique La commune d\'Anderlecht a décid
é de limiter ses impacts sur l\'environnement et de montrer l\'exemple en liant économies financières et gestion environnementale. La Commune a ainsi adhéré depu
is plusieurs années à la charte « Entreprise éco-dynamique " de Bruxelles-Environnement (IBGE).'}, {'url': 'https://www.aqi.in/fr/dashboard/Belgium/BrusselsCapit
alRegion/Anderlecht', 'content': "L'AQI actuel de Anderlecht est de niveau 18 GOOD avec une pollution de l'air en temps réel de PM2.5 (10µg/m³), PM10 (18µg/m³),
Température (20°C) à Brussels Capital Region."}, {'url': 'https://www.accuweather.com/fr/be/anderlecht/27501/air-quality-index/27501', 'content': 'Localized Air
Quality Index and forecast for Anderlecht, Bruxelles, Belgique. Track air pollution now to help plan your day and make healthier lifestyle decisions.'}, {'url':
'https://www.meteoblue.com/fr/meteo/outdoorsports/airquality/anderlecht_belgique_2803202', 'content': "Trouvez la qualité de l'air et les prévisions polliniques
actuelles pour Anderlecht. Indice de la qualité de l'air Particules Gaz"}, {'url': 'https://anderlecht.be/fr/plan-action-climat-1070', 'content': "Plan Action Cl
imat 1070 - retrouvez toute l'actualité sur. anderlecht.futureproofed.com. Le changement climatique. Le GIEC, le groupe intergouvernemental d'experts sur l´évolu
tion du climat, a encore rappelé le besoin urgent d'agir avec la parution le 20 mars 2023 du son dernier rapport de synthèse, qui a confirmé encore une fois l'in
fluence sans équivoque de l'Homme sur le Climat"}]Voici quelques informations utiles sur la façon de se protéger de la pollution à Anderlecht :

1. La commune d'Anderlecht a pris des mesures pour limiter ses impacts sur l'environnement et adhère à la charte « Entreprise éco-dynamique » de Bruxelles-Enviro
nnement. Vous pouvez en savoir plus sur [ce lien](https://anderlecht.be/fr/eco-gestes-au-quotidien).

2. L'indice de qualité de l'air actuel à Anderlecht est de niveau 18 GOOD, avec une pollution de l'air en temps réel de PM2.5 (10µg/m³) et PM10 (18µg/m³). Vous p
ouvez consulter les données en temps réel sur [ce site](https://www.aqi.in/fr/dashboard/Belgium/BrusselsCapitalRegion/Anderlecht).

3. Pour suivre l'indice de qualité de l'air et planifier vos activités en conséquence, vous pouvez consulter les prévisions sur [AccuWeather](https://www.accuwea
ther.com/fr/be/anderlecht/27501/air-quality-index/27501).

4. Pour obtenir des informations sur la qualité de l'air et les prévisions polliniques actuelles à Anderlecht, vous pouvez visiter [ce site](https://www.meteoblu
e.com/fr/meteo/outdoorsports/airquality/anderlecht_belgique_2803202).

J'espère que ces informations vous seront utiles pour vous protéger de la pollution à Anderlecht.

> Finished chain.
```

II. Air Quality Prediction Tool `name="air_quality_prediction"`

- Predicts air quality metrics (PM10 and PM2.5) for a specified date using pre-trained LSTM models serialized as .pkl files.
- Leverages local weather data (weather.json) for feature inputs.



```
AgentExecutor
4.41s  131
ChatPromptTemplate  Hub  0.00s
ChatOpenAI  gpt-3.5-turbo-0125  0.65s
air_quality_prediction  1.42s
ChatPromptTemplate  Hub  0.00s
ChatOpenAI  gpt-3.5-turbo-0125  2.31s
```



```
Invoking: `air_quality_prediction` with `2024-11-29`


2024-11-28 03:00:26.453365: I tensorflow/core/platform/cpu_feature_guard.cc:210] This TensorFlow binary is optimized to use available CPU instructions in perform
ance-critical operations.
To enable the following instructions: AVX2 AVX512F AVX512_VNNI FMA, in other operations, rebuild TensorFlow with the appropriate compiler flags.
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 269ms/step
[[-1533.4913330078125, 1433.935302734375]]The predicted air quality for Anderlecht on November 29, 2024, is as follows:
- PM10: -1533.49
- PM2.5: 1433.94

Please note that these values are subject to change and may vary.

> Finished chain.
```

Workflow of Air Quality Prediction Tool

1. Weather Data Loading:
   - Reads weather.json to retrieve weather metrics for the specified date.
   - Filters and preprocesses the data using functions from Utils/utils.
2. Model and Scalers:
   - Loads the pre-trained LSTM model and associated scalers from the .pkl file.
   - Scales the input data using the stored scaler_X.
3. Prediction:
   - Reshapes the scaled input data to match the LSTM model's expected format.
   - Uses the LSTM model to predict scaled air quality values.
   - Inversely transforms the predictions to their original scale using scaler_y.
4. Output:
   - Returns the predicted PM10 and PM2.5 values as a list.

Strengths of the System

1. Modular Design:
   - The separation of tools allows easy addition or modification of functionalities.
   - Each tool is independent and reusable, enhancing scalability.
2. Powerful AI Integration:
   - LangChain's toolchain agent intelligently routes queries to the right tool.
3. Scalable Predictions:
   - Predictions are based on pre-trained models, enabling fast and reliable responses.

Discussion

The LangChain-based API is a crucial component of the project, bridging the gap between advanced AI capabilities and user accessibility. By utilizing the LangChain framework, the system efficiently processes natural language queries and invokes the appropriate tools for predictive modeling or web-based research.

III.6. Web Application (UI)

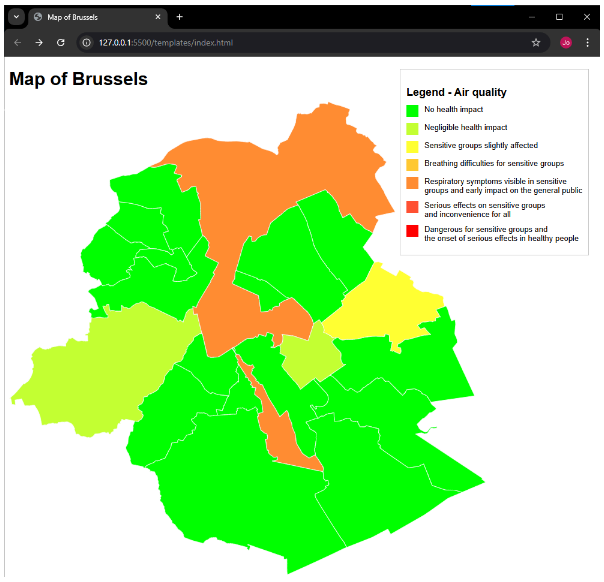
The web application serves as the user interface for the LangChain project, providing users with an interactive platform to visualize air quality data across Brussels and ask questions about specific communes. Below is an in-depth explanation of its components, functionality, and implementation. The web application consists of two primary pages: Index Page (index.html) and Details Page (details.html)

Index Page (index.html)

The index page provides a map-based visualization of air quality data for Brussels, enabling users to assess pollution levels for different communes at a glance.

Key Features:

1. SVG Map Integration: Each commune can be dynamically manipulated and has its pollution level represented by a color code.
2. Legend: A legend explains the color coding for pollution levels, enhancing user understanding of the visualization.
3. Interactive Map: Clicking on a commune redirects users to the Details Page (details.html) for more information and interactive features.

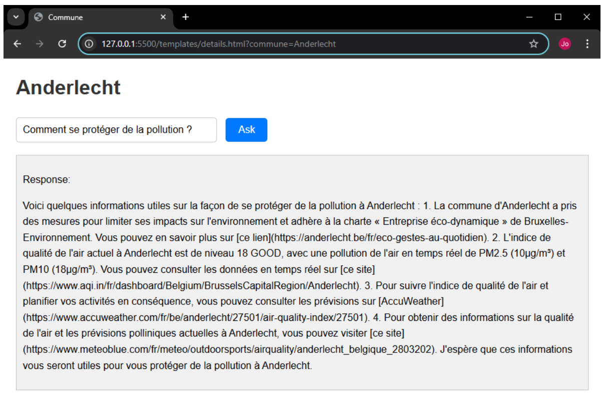

Map of Brussels
127.0.0.1:5500/templates/index.html
Legend - Air quality
🟩 No health impact
🟩 Negligible health impact
🟨 Sensitive groups slightly affected
🟧 Breathing difficulties for sensitive groups
🟧 Respiratory symptoms visible in sensitive groups and early impact on the general public
🟥 Serious effects on sensitive groups and inconvenience for all
🟥 Dangerous for sensitive groups and the onset of serious effects in healthy people

Details Page (details.html)

The details page provides an interface for users to interact with the LangChain AI agent by asking commune-specific questions.

Key Features:

1. AI-Powered Responses: The page sends the user's question, along with the commune context, to a Flask-based backend API (http://127.0.0.1:5000/ask). The response from the API is displayed in the results section.



Anderlecht

Comment se protéger de la pollution ?

Response:

Voici quelques informations utiles sur la façon de se protéger de la pollution à Anderlecht : 1. La commune d'Anderlecht a pris des mesures pour limiter ses impacts sur l'environnement et adhère à la charte « Entreprise éco-dynamique » de Bruxelles-Environnement. Vous pouvez en savoir plus sur [ce lien](https://anderlecht.be/fr/eco-gestes-au-quotidien). 2. L'indice de qualité de l'air actuel à Anderlecht est de niveau 18 GOOD, avec une pollution de l'air en temps réel de PM2.5 (10µg/m³) et PM10 (18µg/m³). Vous pouvez consulter les données en temps réel sur [ce site] (https://www.aqi.in/fr/dashboard/Belgium/BrusselsCapitalRegion/Anderlecht). 3. Pour suivre l'indice de qualité de l'air et planifier vos activités en conséquence, vous pouvez consulter les prévisions sur [AccuWeather] (https://www.accuweather.com/fr/be/anderlecht/27501/air-quality-index/27501). 4. Pour obtenir des informations sur la qualité de l'air et les prévisions polliniques actuelles à Anderlecht, vous pouvez visiter [ce site] (https://www.meteoblue.com/fr/meteo/outdoorsports/airquality/anderlecht_belgique_2803202). J'espère que ces informations vous seront utiles pour vous protéger de la pollution à Anderlecht.

Discussion

The web application effectively bridges the backend AI capabilities and user accessibility:

Strengths:

- Dynamic visualization makes air quality data approachable for users.
- Integration with the AI backend provides actionable insights beyond mere data presentation.

IV. GHG Protocol and CO_2 Footprint Evaluation

IV.1. Introduction

The Greenhouse Gas (GHG) Protocol is an internationally recognized framework for measuring and managing greenhouse gas emissions. It divides emissions into categories.

IV.2. CO_2 Footprint Evaluation

For this project, the carbon footprint evaluation focuses on emissions generated during:

1. Data Collection and Preprocessing
   - Activities:
     - Fetching historical air quality and weather data using APIs.
     - Processing data for merging and cleaning.
   - Emissions Estimate:
     - Data fetching requires network requests and basic processing on a local machine. The consumption values of these queries are difficult to estimate and so is the carbon footprint.
2. Model Training and Serialization
   - Activities:
     - Training machine learning and deep learning models.
     - Saving trained models (.pkl files) to disk.
   - Emissions Estimate:
     - My machine learning models are very small, so they only take a few seconds to train. What's more, they don't need to be re-trained regularly, thus energy consumption is negligible and so is the carbon footprint.
3. API and Web Application Hosting
   - Activities:
     - Flask API server for processing queries.
     - Hosting the web interface for user interaction.
   - Emissions Estimate:
     - According to the website https://rootwebdesign.studio/articles/how-much-carbon-does-a-website-produce/, an average website with 10,000 page views per month could generate 211 kg of CO_2 per year.
4. User Queries
   - Activities:
     - LangChain processing for predictions and internet searches.
   - Emissions Estimate:
     - According to the website https://piktochart.com/blog/carbon-footprint-of-chatgpt/?utm_source=chatgpt.com, each message send to ChatGPT produces about 4.32 grams of CO_2.
       => 1,000queries/month×4.32g CO_2/querie =4.32kg CO_2/month => 4.32×12=51.84 kg CO_2/year

IV.3. Discussion

Not counting the carbon footprint of data collection and langchain tools (being difficult to define), I'm already at an estimated 262.84 kg of CO2 produced per year, which corresponds to the CO2 absorption in 1 year of around 12 trees. The solution would be to balance this out by planting trees.

Not to mention compensation solutions, it is still possible to reduce the carbon footprint in many other ways. For example, I can also do it by using a smaller llm. For the moment, I'm using gpt-3.5-turbo, which consumes less per request than the latest OpenAI model, but I could still use a smaller llm given the low difficulty of my requests. Another solution would be to allow the llm to respond without using a tool when it already knows the answer, so as to avoid polluting Internet searches.

It's also possible to reduce the footprint by using low-power hosting solutions, such as ARM-based cloud servers.

V. Conclusion

This project demonstrates the power and versatility of artificial intelligence in addressing critical urban challenges like air pollution. By integrating machine learning models, data preprocessing pipelines, and an interactive web interface, the system successfully predicts PM10 and PM2.5 levels while also providing valuable environmental insights through a LangChain-powered API.

The system's predictive capabilities enable individuals, particularly those with respiratory vulnerabilities, to take proactive measures to safeguard their health. Additionally, by offering access to information on ongoing and future sustainable projects, the system fosters awareness and promotes environmentally conscious decision-making within the community.

Key achievements of the project include:

Accurate predictions of air quality using advanced deep learning models, such as LSTM, tuned for optimal performance. A user-friendly platform that visualizes real-time and predicted air pollution levels across Brussels communes. Alignment with Sustainable Development Goals (SDG 3: Good Health and Well-being, and SDG 11: Sustainable Cities and Communities), supporting healthier and more sustainable urban living. The system is designed to be scalable, making it adaptable for other cities and regions facing similar environmental challenges. Future enhancements, such as incorporating additional features like real-time traffic or industrial data, could further improve prediction accuracy and utility.

This project not only contributes to public health and environmental awareness but also demonstrates how AI-driven solutions can empower individuals, policymakers, and researchers to make informed decisions for a cleaner and healthier future.

VI. Bibliographie

VI.1. Website about SDGs & Environmental Report

- https://sdgs.un.org/fr/goals (consulted in November 2024)
- https://www.who.int/data/gho/data/themes/topics/indicator-groups/indicator-group-details/GHO/sdg-target-3.9-mortality-from-environmental-pollution (consulted in November 2024)
- https://www.eufje.org/images/docConf/visio2020/REPORT_BELGIUM.pdf (consulted in November 2024)

VI.2. Website about API

- https://aqicn.org/here/fr/ (consulted in November 2024)
- https://www.langchain.com/langsmith (consulted in November 2024)
- https://platform.openai.com/docs/overview (consulted in November 2024)
- https://app.tavily.com/ (consulted in November 2024)

VI.3. Website about LangChain Tuto

- https://python.langchain.com/docs/tutorials/ (consulted in November 2024)

VI.4. Website about Carbon Footprint

- https://piktochart.com/blog/carbon-footprint-of-chatgpt/?utm_source=chatgpt.com (consulted in November 2024)
- https://rootwebdesign.studio/articles/how-much-carbon-does-a-website-produce/ (consulted in November 2024)
- https://ghgprotocol.org/ (consulted in November 2024)