Final Project Report - Smart Energy Assistant

AMS559 / CSE551: Smart Energy in Information Age

https://github.com/ankita-sethi/smartenergy-assistant

1. Objective

The objective of this project is to develop a smart energy assistant that leverages LLM to engage with users about energy consumption in a smart home setting. The assistant, powered by a fine-tuned large language model, will monitor appliance efficiency and anomalies, and predict energy consumption based on weather forecasts along with controlling the home appliances. This project aims to empower users to make informed decisions about their energy usage, ultimately leading to reduced energy costs and a more sustainable lifestyle.

2. Background

In 2022, the residential sector used 1.42 trillion kWh of electricity, 35% of U.S. total electricity use [4]. The integration of smart home technologies has fueled a demand for intelligent energy management solutions. As homes have become increasingly equipped with interconnected devices and systems, optimizing energy consumption has become paramount. Adding to this, residential energy consumption contributes significantly to overall usage, severely impacting both finances and the environment. The goal is to empower users to manage their energy consumption effectively, contributing to a more sustainable future by reducing energy consumption and costs.

3. Data

We are using the dataset 'Smart Home Dataset with Weather Information' [3] consisting of the CSV file - HomeC.csv. The dataset consists of the energy consumption of household appliances at 1-minute intervals of 350 days of house appliances in kW from a smart meter along with the weather conditions of the region. By leveraging the Smart Home Dataset with Weather Information, we gain a holistic understanding of energy usage dynamics within the smart home ecosystem.

4. Preprocessing

In the data preprocessing stage, several steps were undertaken to ensure the dataset's quality and readiness for analysis and modeling.

During the data cleaning and preprocessing phase it was observed that two columns exhibited high correlation as shown in Fig 4.1: "use" with the "House Overall" column and "gen" with "SOLAR." To address this, we dropped the "use" and "gen" columns. By eliminating these highly correlated columns, redundancy was reduced, and the dataset was streamlined for more efficient analysis and modeling. This step ensured that the resulting dataset remained concise and focused, containing only relevant features essential for the subsequent stages of analysis and modeling.

We have processed the data in such a way that we first standardized the column names by removing the "[kW]" suffix to ensure uniformity and clarity. Then, we aggregated related energy consumption readings by creating new columns, "Furnace" and "Kitchen", which represent the combined energy usage of furnaces and kitchen appliances, respectively. By consolidating these readings, we aimed to simplify the dataset and provide a clearer representation of energy consumption in these specific areas. This meticulous preprocessing enhances the dataset's coherence and prepares it for more effective analysis and modeling.

Additionally, we addressed invalid values in the 'cloudCover' column by applying a backfill method to replace them. This ensured the integrity of the weather data, allowing for accurate analysis. This preprocessing step ensured that the dataset remained reliable and suitable for subsequent analysis and modeling tasks.

We enriched the dataset by extracting temporal features from the index. This involved creating new columns for the month, day, weekday, hour, and minute, derived from the timestamp index. These additional features provide valuable temporal context, allowing for a more granular analysis of energy consumption patterns over different time intervals. By incorporating these temporal features, we enhanced the dataset's richness and enabled more nuanced insights into energy usage trends within the smart home environment.

Our primary emphasis lies on these key appliances - the refrigerator, furnace, and dishwasher, although the scope can be expanded as needed.

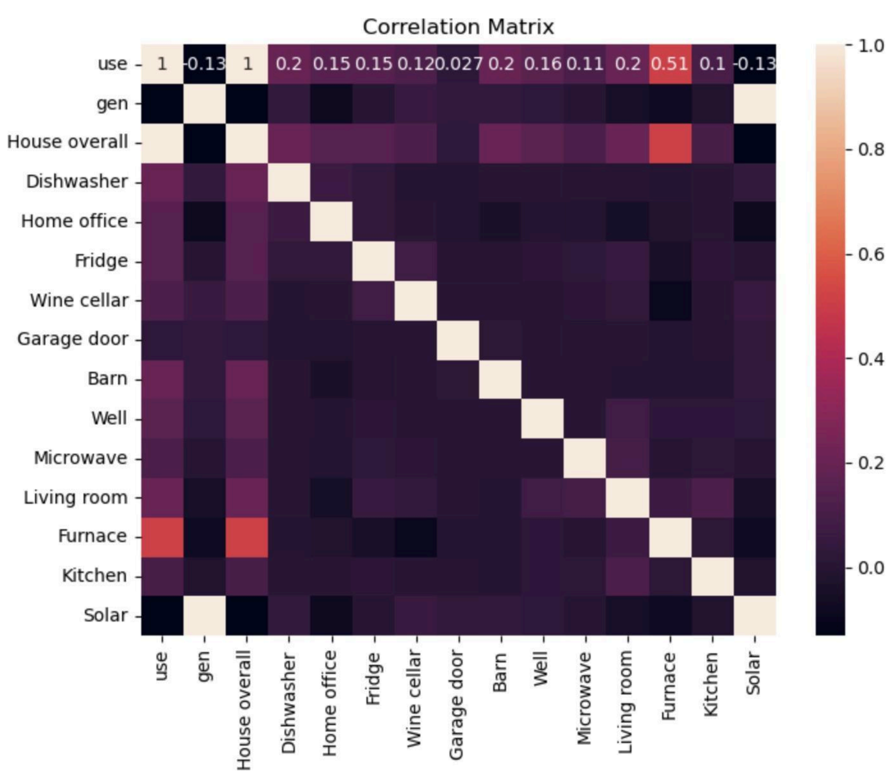

Fig: 4.1 Correlation Matrix

5. Prediction Models

In the data cleaning and analysis phase, various predictive models were explored to forecast energy consumption trends. Following a thorough evaluation, the ARIMA (AutoRegressive Integrated Moving Average) model emerged as the preferred choice due to its superior performance in terms of Mean Squared Error (MSE) [3].

We divided our dataset into training and testing sets, with 70% of the data allocated for training. The training set contained a total of [insert number] data points, while the testing set comprised [insert number] data points. Next, we employed a statistical model known as ARIMA (AutoRegressive Integrated Moving Average) to analyze the energy consumption patterns of our selected appliances. Our ARIMA model was trained on the training dataset using an order of (2,1,1). We then utilized the model to generate forecasts for the testing dataset. The predictions were plotted alongside the actual test data to visually assess the model's performance as seen in Fig 5.1.

In addition to forecasting, we also conducted walk-forward validation to evaluate the accuracy of our predictions. This involved iteratively updating the model with new data points and generating forecasts for each step. The forecasts were compared against the actual outcomes to compute performance metrics such as Mean Squared Error (MSE), Root Mean Squared Error (RMSE), Mean Absolute Error (MAE), R-squared score, and Akaike Information Criterion (AIC). These metrics provide insights into the model's predictive capability and overall performance.

After forecasting the energy consumption for furnace data, we obtained the following evaluation metrics: Mean Squared Error (MSE) of 0.00460, Root Mean Squared Error (RMSE) of 0.068, Mean Absolute Error (MAE) of 0.044, and an R-squared score of 0.737. Subsequently, we applied similar forecasting techniques to the fridge and dishwasher data.

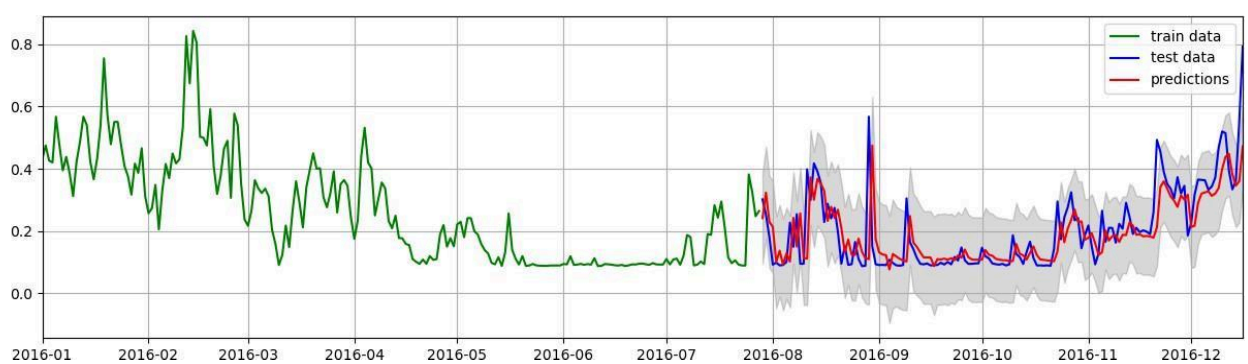

Fig 5.1 Plot with predicted and actual energy consumption values for a Furnace

6. LLM

6.1. Integration of LLM

For the integration of Large Language Models (LLMs), we have considered two approaches: fine-tuning and few-shot learning. Fine-tuning involves customizing a pre-trained LLM on specific datasets related to our application domain, enabling it to learn task-specific patterns and nuances. This approach allowed us to adapt the model's parameters to our specific requirements, improving its performance for tasks such as text generation, natural language understanding, and recommendation systems. On the other hand, few-shot learning enables the LLM to generalize from a small number of examples or prompts, making it suitable for scenarios where labeled data is limited or unavailable. By exploring both fine-tuning [6] and few-shot learning techniques, we aimed to identify the most effective approach for integrating LLMs into our system, ensuring optimal performance and versatility across different applications and use cases.

In addition to fine-tuning and few-shot learning, the integration of Large Language Models (LLMs) included specific functionalities like turn_off(), turn_on(), status(), and predict(). These enable the LLM to perform actions based on user prompts, enhancing user experience. For instance, turn_on() activates devices, turn_off() deactivates them, status() provides device status, and predict() anticipates trends. Few-shot prompting utilizes concise prompts to elicit actions, generating structured outputs for accurate execution. For example, "Turn on the kitchen lights" prompts a JSON output specifying the action, service, and target. By incorporating these functionalities, our LLM integration aims to streamline smart home interaction, ensuring convenience and efficiency. Based on our empirical analysis, we observed that the few-shot learning method outperformed fine-tuning and demonstrated greater efficiency. Consequently, we decided to adopt the few-shot learning approach as our final method.

We have chosen to implement few-shot prompting, enabling concise prompts for accurate action execution.

In our efforts to enhance the language model (LLM) for user assistance tasks and improve its ability to respond effectively to prompts with minimal training data, we developed synthetic datasets. These datasets were crucial for training and refining the LLM's understanding and responsiveness to user commands in various scenarios. For example, we considered the task of turning on an office light. In this case, as shown in Fig. 6.1.1, the output was structured to include the necessary details for executing the command, such as the response to be provided ("to_say"), the specific service or action required ("service"), and the target entity affected by the command ("target"). This structured approach ensured clarity and precision in the LLM's responses, facilitating seamless interaction between users and the assistant. By leveraging synthetic datasets like these, our goal was to improve the LLM's ability to comprehend and fulfill user requests across different contexts, thereby enhancing its utility as a user assistant.

```
web-ui > src > {} prompt.json > ...
{
    "user prompt": "Turn on the office light.",
    "formatted output": {
        "to_say": "Office lights successfully turned on",
        "service": "turn_on()",
        "target": "office.light"
    }
}
```

Fig 6.1.1 Synthetic dataset sample

Along with this, we integrated a Language Model (LLM) to facilitate the control of appliances through a user interface (UI). Initially, we targeted three appliances: the fridge, furnace, and dishwasher. Users can interact with a chatbot in the UI to control these appliances. Due to the lack of actual appliances, we simulated data by manually entering energy consumption values. This approach allowed us to trigger and test various processes within the system. The integration of LLM enabled seamless interaction and control of appliances, demonstrating the potential for real-world application in smart home environments. For example, if a user asks the chatbot to turn on a light, the chatbot responds with the service turn_on() due to the specifically engineered prompt, which when triggered, performs the rest of the operations to simulate turning on the light. This integration demonstrated the potential for real-world application in smart home environments.

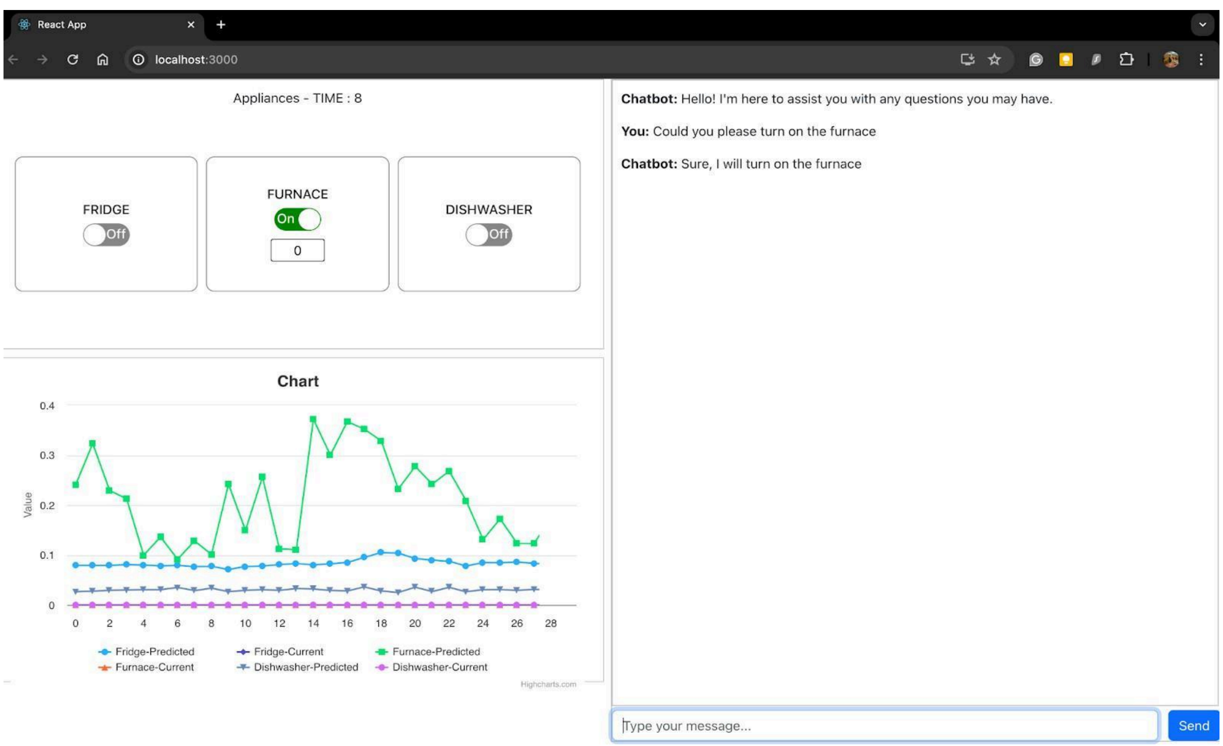

Fig 6.1.2 Screenshot of UI showcasing Appliance Management

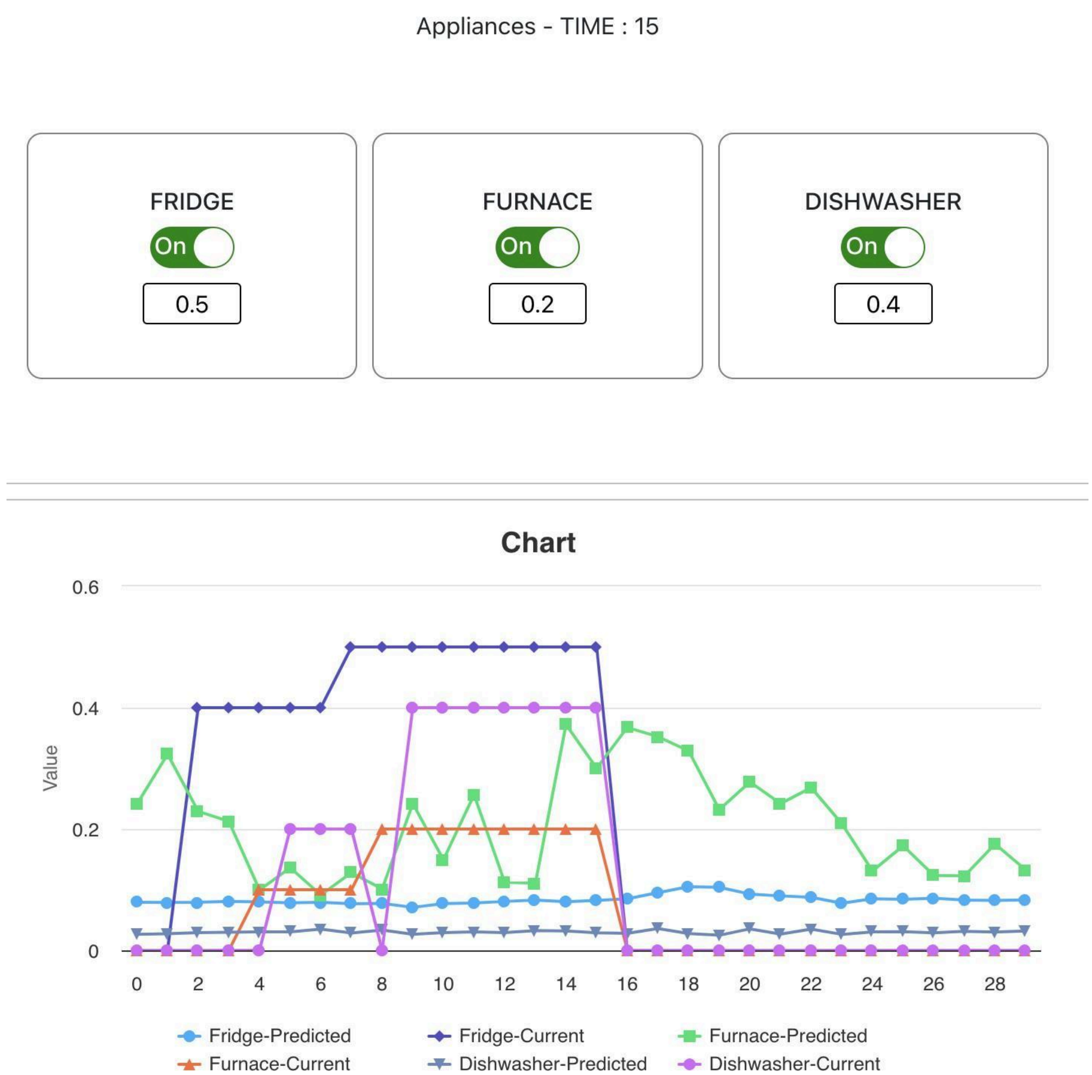

Fig. 6.1.3 Plotting the predicted energy consumption for the next 30 minutes along with the current value at the time that the user inputs

As depicted in Figure 6.1.3, the chart displayed on the user interface (UI) provides users with the predicted values for the next 30 minutes, alongside tracking current values. If an appliance is off, the energy consumption value is considered as 0 kWh, if it is on, it reflects the value entered at the current time. It updates the values in real time. This feature enhances the user experience by providing real-time insights and allowing for interactive control over the appliance values displayed on the chart.

6.2. Anomaly Detection

Anomaly detection is a crucial aspect of our project, as discussed during the presentation, aiming to identify unusual consumption patterns or deviations from expected behavior. By leveraging different techniques, we can detect anomalies in real-time data streams and alert users to potential issues in order to ensure that the output is desired. This proactive approach enables users to take corrective action promptly, minimizing downtime, optimizing energy usage, and preventing potential risks or inefficiencies.

In order to identify unusual patterns in the data, a statistical analysis was performed on the predicted versus actual values of temperature, humidity, and energy usage. Energy usage was given the most weightage due to its significant impact on the overall performance of the system. By applying statistical techniques such as mean absolute error and root mean squared percentage error, we were able to detect anomalies in the data that may indicate potential issues or outliers. To achieve this, we performed data preprocessing, feature engineering, and statistical modeling. First, we cleaned and normalized the data to ensure consistency and accuracy. Next, we extracted relevant features such as mean, median, and standard deviation of each parameter. Then, we applied a regression algorithm to predict the actual values based on historical data. Finally, we calculated the errors and flagged data points with errors exceeding a certain threshold as anomalies. This approach enabled us to identify unusual patterns and outliers in the data, providing valuable insights for further investigation.

We followed another approach by providing the data points to the LLM and asked to perform statistical analysis and explain using a chain of thought providing the context. The results were quite similar to our algorithmic approach and so, we replaced traditional statistical algorithms with LLM explainability, as it offered comparable results.

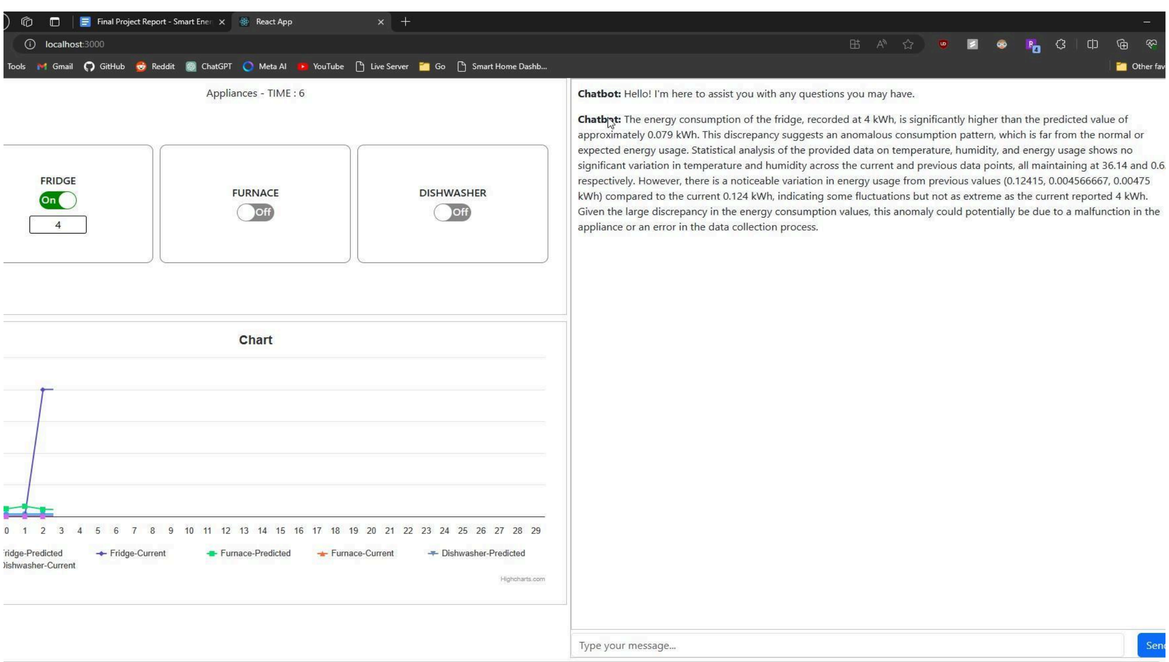

Fig. 6.2.1 Screenshot of UI showcasing anomaly detection by LLM

6.3. Explainability

Explainability in our project ensures transparency and clarity in how our smart energy assistant operates. We use transparent algorithms, provide interpretability of recommendations, offer a feedback mechanism, present visualizations, and provide educational resources. This empowers users to understand the reasoning behind recommendations and make informed decisions to reduce energy consumption and promote sustainability.

To improve the clarity of our model's findings, we've adopted the chain of thought method. This systematic approach enables us to thoroughly trace the reasoning behind the conclusions reached by the model. Particularly in anomaly detection, it's crucial for the model to provide explanations backed by solid statistical evidence. To aid in this endeavor, we provide the model with additional contextual data. This data includes historical records, allowing us to identify past patterns, as well as real-time information such as temperature, humidity, and energy usage from various sources. By incorporating this diverse range of contextual cues, our model can offer explanations that are not only statistically sound but also enriched with insights from a wide array of environmental factors.

When providing large amounts of data to Llama 3 and prompting it to interpret the given data directly, it hallucinated and fabricated new attributes, such as 'Drugs', ‘Lighting’ and 'Standby', which were not present in the original dataset. For instance, it claimed that ‘Lighting’ had a significant impact on the outcome, despite not being mentioned in the data. Although the values and attributes were entirely fictional, the explainability provided by Llama 3 was more reasonable and coherent compared to GPT-4's response, which failed to provide a logical explanation for its outputs.

```
date: 1/1/20
time: 0:01
House overall: 1100
Fridge: 200
Dishwasher: 0
Drugs: 0
Standby: 100
Cooking: 0
Lighting: 0
month: 1
day: 1
weekday: 3
hour: 0
minute: 1
temperature: 6.8
humidity: 65
visibility: 10
```

Interpretation:

These two entries represent energy usage and weather data for two consecutive minutes (00:00 and 00:01) on January 1st, 2020.

- **House overall** energy usage increased by 100 units (from 1000 to 1100) in just one minute, which could indicate a sudden surge in energy consumption, possibly due to a household member turning on a high-power appliance.
- **Fridge** energy usage remained constant at 200 units, suggesting normal operation.
- **Dishwasher** and other appliances showed no energy usage, indicating they were not in use during this time.
- **Temperature** increased by 0.1 degrees Celsius (from 6.7 to 6.8) in one minute, which is a relatively rapid change.
- **Humidity** decreased by 2% (from 67 to 65) in one minute, which could be a normal fluctuation.
- **Visibility** remained constant at 10, indicating no significant change in weather conditions.

These observations suggest a sudden change in household energy usage, possibly due to a specific event or action, and a slight change in temperature and humidity.

Fig. 6.3 Output from Llama 3 wrt to provided data and explainability

Due to hardware constraints and the lack of a publicly accessible API for Llama 3, we turned to few-shot learning on GPT-4 as a viable alternative. This approach enabled us to incorporate additional supporting context, including historical data, temperature, humidity, and energy usage, to provide a comprehensive understanding of the model's output. In the end, we replaced traditional statistical algorithms with LLM explainability, as it offered comparable results.

7. Future Scope

Testing the project with actual smart devices would be a crucial next step to validate its functionality and ensure that it can effectively monitor and control energy consumption. By connecting real devices, we could verify that the project can accurately collect data, make predictions, flag irregularities and control appliances based on the predictions.

Integrating with open-source LLMs like StabilityLM and the Home LLM model present a good opportunity to enhance the project's performance. These models are designed to understand and analyze natural language, which can be incredibly valuable for interpreting user commands, analyzing energy usage patterns, and providing intelligent recommendations. StabilityLM, for example, is known for its robustness and stability in generating coherent and contextually relevant text. By leveraging StabilityLM, the project could potentially improve its ability to generate insightful reports, alerts, and explanations related to energy consumption trends and anomalies. Similarly, the Home LLM model, trained specifically for smart home applications, could provide valuable insights into energy usage patterns and suggest optimized settings for energy-saving purposes. Its ability to understand context and learn from user interactions could make the project more intuitive and user-friendly. Overall, integrating with these open-source LLMs can bring advanced natural language processing capabilities to the project, enhancing its performance, usability, and overall value to users seeking to manage their energy consumption more effectively.

Expanding the project to integrate with HomeAssistant, an open-source home automation platform, opens up a wide range of possibilities. HomeAssistant supports a vast array of smart devices and protocols, allowing for seamless integration and control of various appliances. This integration could enhance the project's functionality by providing users with a centralized platform to monitor and control their smart devices, including energy-consuming appliances.

Overall, the future scope of this project is promising. Testing with real devices, integrating with advanced LLMs, and expanding to integrate with HomeAssistant can all contribute to making the project more effective and valuable for users seeking to manage their energy consumption more efficiently.

8. Conclusion

In conclusion, our project has aimed for smart energy management through the development of an assistant powered by large language models (LLMs). By leveraging advanced data preprocessing techniques, predictive modeling, and anomaly detection mechanisms, we have laid the groundwork for a system that empowers users to make informed decisions about their energy consumption and also provides an energy consumption forecast. The integration of LLMs has enabled seamless interaction and personalized assistance, enhancing user experience and promoting sustainability. Looking ahead, we're focused on spotting anomalies, adding more devices, and keeping data safe. With a focus on users and a commitment to progress, our goal is to make managing energy simpler and more sustainable for everyone.

9. Bibliography

1. King, E., Yu, H., Lee, S., & Julien, C. (2023). "Get ready for a party": Exploring smarter smart spaces with help from large language models. ArXiv. /abs/2303.14143
2. Dmitriy Rivkin, Francois Hogan, Amal Feriani, Abhisek Konar, Adam Sigal, Xue Liu, Gregory Dudek. SAGE: Smart Home Agent with Grounded Execution. arXiv:2311.00772v2 [cs.AI] 19 Jan 2024
3. Kaggle. Smart Home Dataset with weather Information. [(accessed on 10 May 2024)]. Available online: https://www.kaggle.com/datasets/taranvee/smart-home-dataset-with-weather-information/data
4. Residential Buildings Factsheet: https://css.umich.edu/publications/factsheets/built-environment/residential-buildings-factsheet
5. OpenAI. Open AI Cookbook [(viewed on 10 May 2024) Available online: https://github.com/openai/openai-cookbook/blob/main/examples/How_to_finetune_chat_models.ipynb
6. GiaZucc. Kaggle.Smart Home IoT-EDA, ARIMAs, LSTM and more https://www.kaggle.com/code/piergiacomofonseca/smart-home-iot-eda-arimas-lstm-and-more
7. ChatGPT: https://chat.openai.com/