Machine Learning for IoT

Homework 3

Matteo Donadio
Student ID: s303903
Politecnico di Torino

Alessandro Maria Feri'
Student ID: s305949
Politecnico di Torino

Mattia Sabato
Student ID: s305849
Politecnico di Torino

I. Data Collection, Communication, and Storage

When deciding between REST and MQTT as communication protocols, it's crucial to consider the specific requirements of the application being developed. Given the ultimate goal of monitoring the battery status of potentially numerous devices, it becomes evident that the communication aspects represent the primary bottleneck. MQTT, being lightweight and efficient due to its limited header size, makes managing packets considerably easier compared to REST. This advantage translates into significant benefits in terms of battery consumption and bandwidth usage, crucial in an IoT ecosystem. Furthermore, the asynchronous communication paradigm provided by MQTT's publish/subscribe nature facilitates dynamic communication across devices, eliminating the need for explicit human intervention. This feature ensures that, as soon as the information is produced, it is also delivered for consumption to other interested clients, enhancing scalability and enabling One-to-Many communications, fundamental in IoT environments. Additionally, the Quality of Service (QoS) feature offered by MQTT plays a crucial role, especially in situations where the batteries being monitored are used for sensitive or critical applications. This MQTT's ability to provide control over message delivery guarantees that information is reliably sent and received, further justifying its choice as the preferred communication protocol. Moreover, it's important to note that uniform and standardized data formats are expected across all devices where the application is aimed to be deployed. While the underlying hardware may vary, the unique format characterizing the information related to the battery status simplifies the communication and optimization processes. This contrasts with REST, which is preferred when dealing with different varieties of data formats over the web. In conclusion, MQTT's lightweight nature, asynchronous communication paradigm, QoS capabilities, and support for standardized data formats make it the preferred choice for efficiently enabling communication in our battery monitoring application.

II. Data Management & Visualization

Following the order adopted in Table I, we present the motivations for choosing each of the HTTP methods:

- The retrieval of the list of monitored MAC addresses can be accomplished implementing and making use of the **GET** method. It allows access to information from the server without modifying or removing it, making it suitable for the task at hand;
- Similarly to the previous case, the requirement is to retrieve the battery status associated with a single MAC address within a specified date range. The underlying rationale remains the same, as the information needs to be accessed without requiring additional manipulations. Once again, the **GET** method is well-suited for this scenario;
- To delete a timeseries associated with a specified MAC address, it is necessary to remove this information from the server. For this purpose, the **DELETE** method should be implemented, potentially together with a certification system to restrict this action to authorized users for security reasons.

Method	Endpoint	Description
GET	**/devices**	Retrieve the list of MAC addresses of the monitored devices.
GET	**/device/{mac_address}**	Retrieve battery status information of the device with the specified MAC address in the specified date range.
DELETE	**/device/{mac_address}**	Delete the timeseries associated to the specified MAC address.

TABLE I: *HTTP Methods*