HVAC Waste Detection - Project Presentation

Daniele Polidori

Course of Internet of things
University of Bologna

Academic year 2023-24

IoT system

It **monitors** the temperature of an house, to prevent useless HVAC consumption, detecting it by rapid **temperature changes**.

Composed by an **ESP-WROOM-32** board linked to:

- 1 **indoor DHT22**,
- 1 **outdoor DHT22**,
- 1 **LED**.

Data acquisition - ESP32

`Thing.CoAP` : to act as a **CoAP server**.

`PubSubClient` : to act as a **MQTT subscriber**.

Through **CoAP**, the ESP32 is able to **send** the latest collected indoor and outdoor temperature value, **when asked**.

Through **MQTT**, the board **can receive** some **commands**:

- to start or stop the **sensors reading**,
- to change their **sampling rate**,
- to turn on or off the **LED**.

My laptop acts as a MQTT broker, through Mosquitto.

Data proxy - 1st Python script

`paho-mqtt` : to act as a **MQTT publisher**.

`aiocoap` : to act as a **CoAP client**.

`influxdb-client` : to **store** data.

Initially, through **MQTT**, the application gives commands, to the ESP32, to **start** the sensors reading and to set their **sampling rate**.

Periodically, through **CoAP**, the script **requests**, to the board, the latest collected indoor and outdoor temperature value.
It **continuously stores** these values on a local InfluxDB instance.

The network **latency**, between the temperature value request and its reception, is continuously monitored and, after a while from the beginning, the mean value is calculated.

Data analytics - 2nd Python script (1/2)

`influxdb-client` : to **get** and **store** data.

`prophet` : to **forecast** future temperature values.

`paho-mqtt` : to act as a **MQTT publisher**.

At the **beginning**, the script gets all past temperature values to **forecast** some indoor and outdoor values.
As the times are reached, the application **stores** the predicted values on the database.

Ciclically, the application retrieves some of the latest temperature values and **analyses** them to check a possible HVAC waste.
When the alarm **goes off**, the script **stores** the event on the database and, through **MQTT**, gives the command, to the ESP32, to turn on the **LED**; when the risk has passed, the script gives the command to turn it off.

Data analytics - 2nd Python script (2/2)

The alarm **goes off** if:

- the indoor temperature is **changing** rapidly (1) and
- the indoor temperature is **approaching** the outdoor one (2).

Mathematically speaking:

$$var(i_1, i_2, \ldots, i_n) > threshold \tag{1}$$

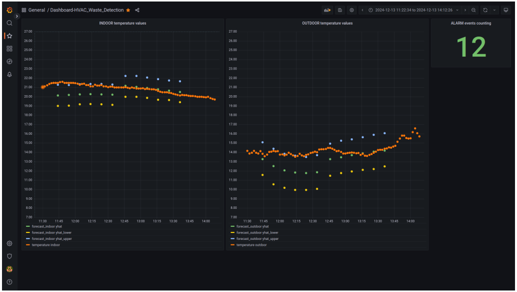
$$min(i_n, o_n) < mean(i_1, i_2, \ldots, i_n) < max(i_n, o_n) \tag{2}$$

where i is the indoor temperature, o is the outdoor temperature and $t_1, t_2, \ldots, t_n$ are the n latest temperature values retreived: t_1 is the newest and t_n is the farthest.

Data visualization

Local **Grafana** instance that shows:

- the **collected** temperature values,
- the **forecasted** ones,
- the counting of the **alarm** events.



Setup - Data proxy

ESP32)

Indoor DHT sampling rate : 3 sec.

Outdoor DHT sampling rate : 20 sec.

Data acquisition process)

Latest **temperatures** request : every 5 sec.

Mean network **latency** evaluation : after 1 hour.

Setup - Data analytics

Forecast)

Data **obtained**	: on start.
Temperatures **collection**	: past month (unevenly).
aggregateWindow	: every 20 sec (mean function).
Num. of values **retreived**	: 6500 indoor, 6500 outdoor.
Values **forecasted**	: every 10 min, for 6 times.

Alarm)

Data **obtained**	: every 30 sec.
Temperatures **collection**	: latest 2 min.
Alarm **threshold** (1)	: 0.03 $^\circ C^2$.

Evaluation

Day : on December 13th.

Data **proxy** started : at 11:30.

Data **analytics** started : at 12:36.

Mean network **latency** : 150 millis (ranging from 5 to 421).

Max difference **indoor** forecasted-observed : 0.39 °C.

Max difference **outdoor** forecasted-observed : 1.23 °C.

(Always within the forecasted range.)

Results - Indoor forecast

Current Time	Observed[a] Temperature (°C)	Forecasted Temperature (°C)		
		yhat	yhat lower	yhat upper
12:46	21.04	**21.17**	19.99	22.25
12:56	20.88	**21.07**	19.98	22.24
13:06	20.85	**20.95**	19.88	22.06
13:16	20.63	**20.81**	19.69	21.90
13:26	20.42	**20.66**	19.66	21.77
13:36	20.13	**20.52**	19.43	21.67

[a]Mean temperature value (of the 2 minutes around).

Results - Outdoor forecast

Current Time	Observed[a] Temperature (°C)	Forecasted Temperature (°C)		
		yhat	yhat lower	yhat upper
12:46	14.50	13.27	11.49	**14.97**
12:56	14.07	**13.49**	11.78	15.26
13:06	14.01	**13.71**	11.95	15.42
13:16	13.86	**13.92**	12.13	15.65
13:26	13.98	**14.09**	12.21	15.91
13:36	14.40	**14.21**	12.49	16.09

[a]Mean temperature value (of the 2 minutes around).

Considerations

Little **more reliable** in forecasting the **indoor** temperature.
It can be due to the **higher sampling rate** of the indoor sensor (useful to rapidly detect an eventual HVAC waste, unnecessary outdoor).

The experiment done has **limitations**:

- **not many** past temperature **values** available,
- temperature data **collected unevenly**;
- **ESP32** was **positioned** in a naive way (e.g. it should be arranged in such a way as to avoid direct sunlight).