Engineering Department

Master's Degree Artificial Intelligence and Data Engineering

Università degli studi di Pisa
Accademic year - 2023-2024

HydroSense: Real-Time Forecasting of water pump breakages

STUDENT:
Giulio Fischietti

Internet Of Things

Contents

1 Introduction

Water pump breakages can cause significant problems and distress, regardless of whether they are installed in industrial, agricultural, or private settings.

In industrial settings, breakages can heavily impact and delay production, leading to substantial increases in costs and decreases in units sold due to operational downtime. Additionally, they can cause floods and pose security risks for workers. On the other hand, in agricultural settings, breakages are likely to produce floods and damage crops, resulting in food waste and reduced productivity.

As we transition from Industry 4.0 to Industry 5.0, the development of AI-based applications and enhanced machine-human interaction is becoming increasingly prevalent. AI is particularly suitable for industrial settings, as it can detect complex patterns that may be difficult for humans to notice, such as anomalies in sensor values that could indicate an impending breakage. However, it is essential to have a specialized technician present to monitor the decisions made by the algorithm. The human oversight ensures that the AI's predictions are accurate and that any corrective actions are appropriate and timely.

In this project, we present a system capable of forecasting water pump breakages using smart sensors. This system alerts technicians if a breakage is likely to occur in the near future, allowing for preventive maintenance and minimizing downtime and associated costs.

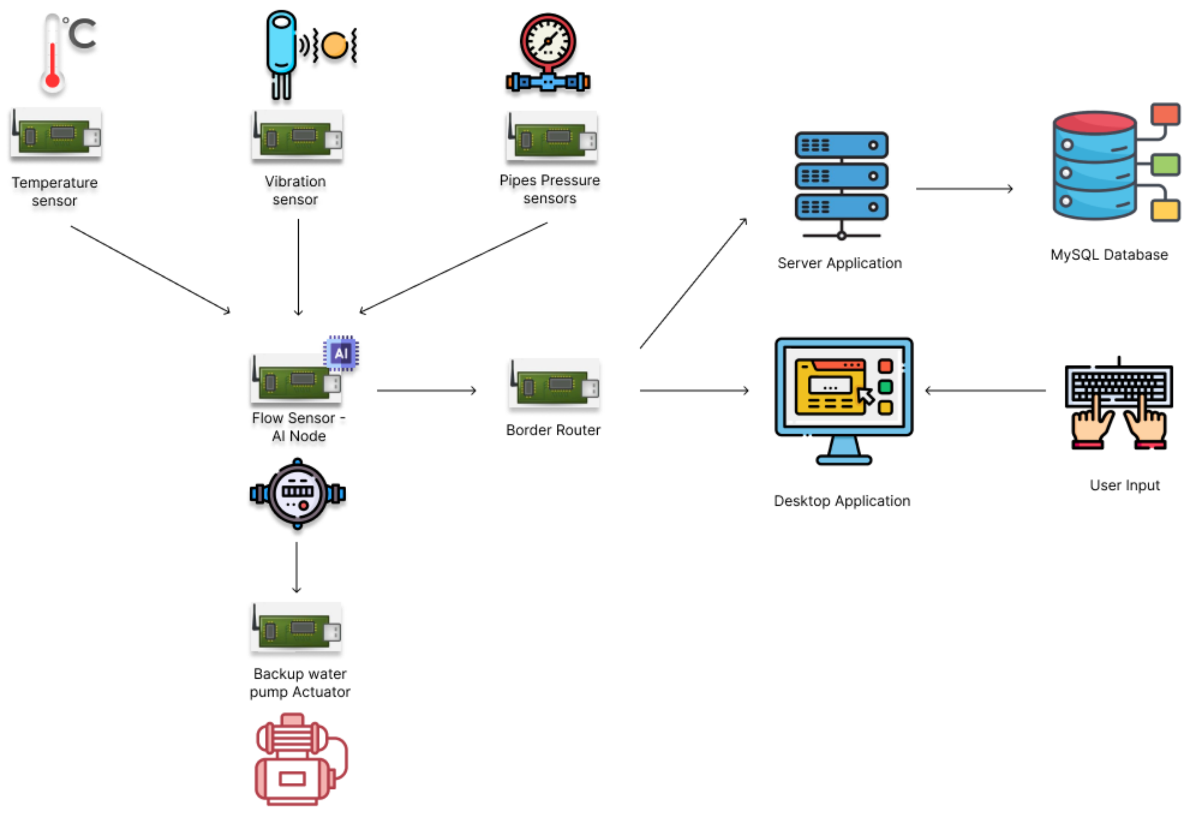
Code is available at this GitHub Link

2 Application Logic

As anticipated in the previous section HydroSense is an AI based application that simplifies the detection of future water pump breakages.
The logic behind its working flow is very simple and leaves the technician total control on the water pumps, while minimizing the risks: right after all the sensors have been set-up and initialized, the system continuosly retrieves data from the sensors, allowing the machine learning algorithm to detect any possibily of breakage.
The result of this prediction is easily visible from the smart sensor (the one with the machine learning model installed) by the color of its led, which will be red if there will be a future breakage or green if the machine is working normally.
When a potential breakage is detected the main water pumps are immediately stopped while the backup pump is powered on: in this period of time, the technician can check the values of sensors in real-time on the desktop application (connected to sensors): after inspecting the main water pump, and possibly performing mantainance, the technician can press a button on the application (or directly on the smart sensor) to bring back to normal functioning of the main water pump, while automatically de-activating the backup pumps.

3 Architecture

The complete system is composed of several applications and sensors, its architecture is shown below, along with a more detailed description:



The system is composed of the following **devices**:

- Temperature sensor
- Vibration sensor
- Pressure sensor
- Flow sensor - which is also entitled to perform predictions through the machine learning algorithm
- Backup pump actuator
- Border Router (only)

The system is also supported by the following **applications**:

- Desktop Application (client): shows real time sensor data to the technician, allows him/her to switch on/off the backup pump actuator, and manually setting the state of the water pump (broken/normal)
- Server application: allows sensors and actuator to register and forwards sensor data to the database. It also allows other nodes and application to retrieve the other nodes IP addresses for resource observation.

3.1 CoAP Network

All sensors and actuators expose their resources creating a CoAP Network: with this lightweight protocol, nodes and applications can communicate with each other encoding data in JSON format, allowing low overhead and high interoperability.

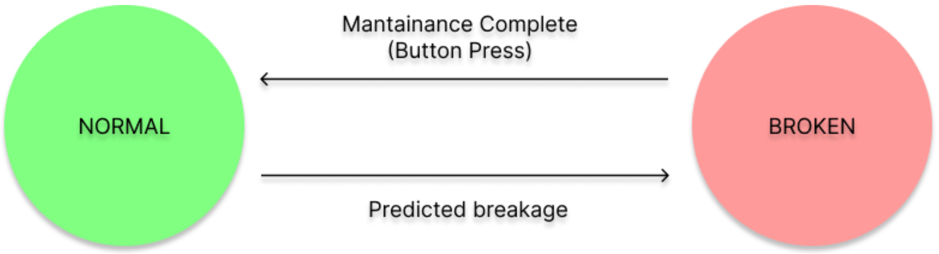
3.1.1 Sensors nodes

Each node exposes a resource, acting as a coap server, more precisely: Temperature, Vibration, and Pressure expose their resources as a **observable periodic** resources, to which the Flow sensor node can subscribe to retrieve periodically data and perform predictions.
Below an example of JSON document periodically sent by the temperature sensor:

```
{
    "sensor_name": "temp", // only specified for server application
    "value": 21.09
}
```

3.1.2 Main Pump Status

The state of the machine is held in the same node that performs predictions, the flow sensor. The state can be either NORMAL or BROKEN: after each prediction has been made by the algorithm, the state is updated with the corresponding value.

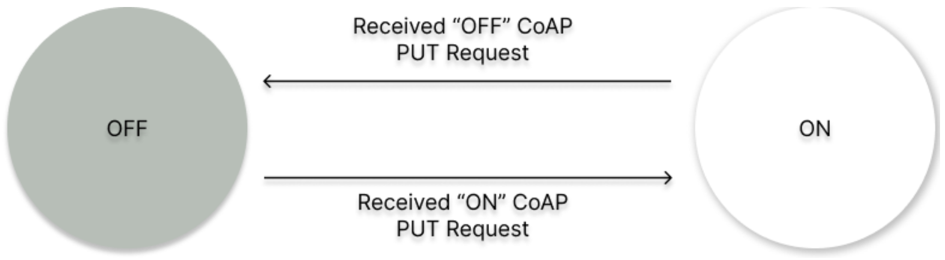


This resource is esposed through an **observable event resource** to which the subscribers (desktop application) receive notifications only when the state of the machine changes, and, at the same time this resource can also be updated with a PUT request: the JSON payload will contain the status the main pump will be, as shown below.

```
{
    "status": 0/1
}
```

3.1.3 Backup Pump Status

Also the backup pump actuator holds a state, which can be ON or OFF. This resource has been also been implemented as an **observable event resource** that will notify the desktop application when a change has been detected.


Received "OFF" CoAP PUT Request
OFF
ON
Received "ON" CoAP PUT Request

When a breakage is detected from main pump, a CoAP PUT Request is sent to the backup pump in order to switch it on and viceversa: the payload contains the desired action, as shown below.

```
{
    "action": "on"/"off"
}
```

When the sensors generate data, they notify the observers by sending a JSON payload as shoen previously: then, after the data is retrieved by the flow sensor, in order to reduce the traffic overhead by performing too many requests ti the server, it performs one single POST request with all the aggregated data, in a json array.

```
[
    {
        "sensor_name": "temp",
        "value": 21.09
    },
    {
        "sensor_name": "flow",
        "value": 12.56
    },
    {
        "sensor_name": "vibration",
        "value": 3.98
    },
    {
        "sensor_name": "pressure",
        "value": 8.34
    }
]
```

Then, the server will perform an INSERT operation into the database concluding the operation.

3.2 Desktop Application

The desktop application acts as a client, subscribing to all resources and showing to the user the real time value of sensors and actuators. More importantly, the state can also be changed by the desktop application through a put method, for backup pump actuator and main pump status.
All the requests the desktop application performs (sensor resource observation and update) have already been described in the previous sections.

3.3 Server Application

The server application is an application whose main functions are the registration of sensors and update of sensor data to the database.
With these two functions, the Flow sensor is able to get the ip address of the other sensors in the network and perform predictions and the desktop application is able to register to each resource correctly and observing/updating the exposed resources.
When nodes or the desktop application need to request the ip address of a node,they will perform a GET request, adding as query parameter the name of the sensor, building a URI like follows:

```
coap://[fd00::1]:5683/sensor_ip_by_name?sensor_name=temp
```

3.4 Database

The database has been implemented using MySQL.
It's composed of two tables: *sensors* and *sensor_data*. *sensor* table is designed in order to store IP addresses of the nodes registered in the network, while *sensor_data* stores all the data generated by sensors: all these informations are forwarded by the border router to the server application to the database.

4 Machine learning Model

As mentioned multiple times, the machine learning model has been deployed on the flow sensor node.
It consists of a deep neural network which has been trained on time series data available on kaggle.

4.1 Sensor data used

The data available on kaggle regarded the usage of 80 unlabeled sensors, which were not practically implementable for the scope of this project, so a feature selection has been performed using random forest regression, selecting the top 4 most relevant features which have been renamed based on their descriptive statistical characteristics as Temperature, Pressure, Vibration and Flow.
This data set has provided three different labels:

- NORMAL: normal functioning
- BROKEN: a breakage has just happened
- RECOVERING: phase following the breakage

For the objective of this algorithm RECOVERING label has been re-assigned to BROKEN so that it will be able to predict breakages periods.

Below are shown plots of each sensor values, with breakdowns highlighted in red, and recovering parts in orange:

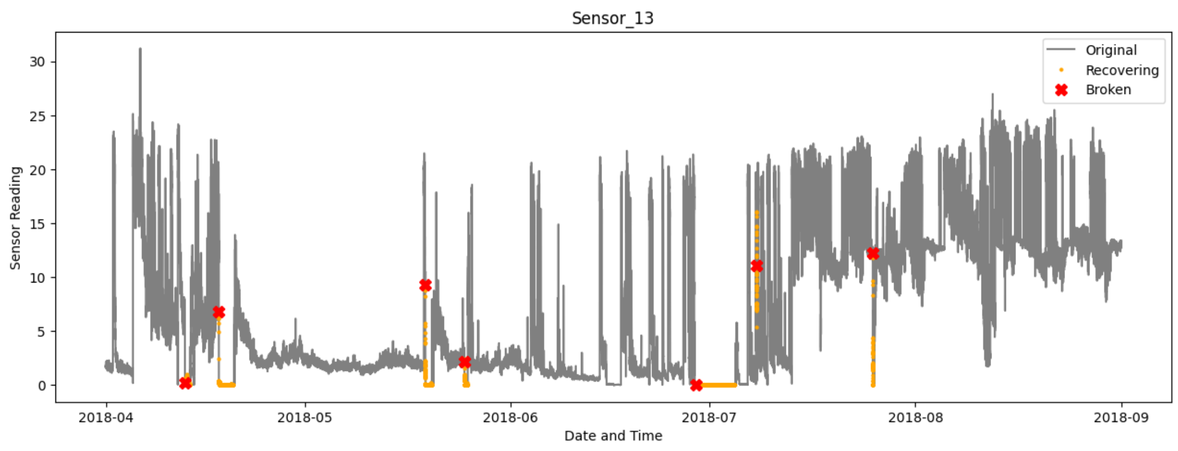


Figure 1: Sensor values: vibration

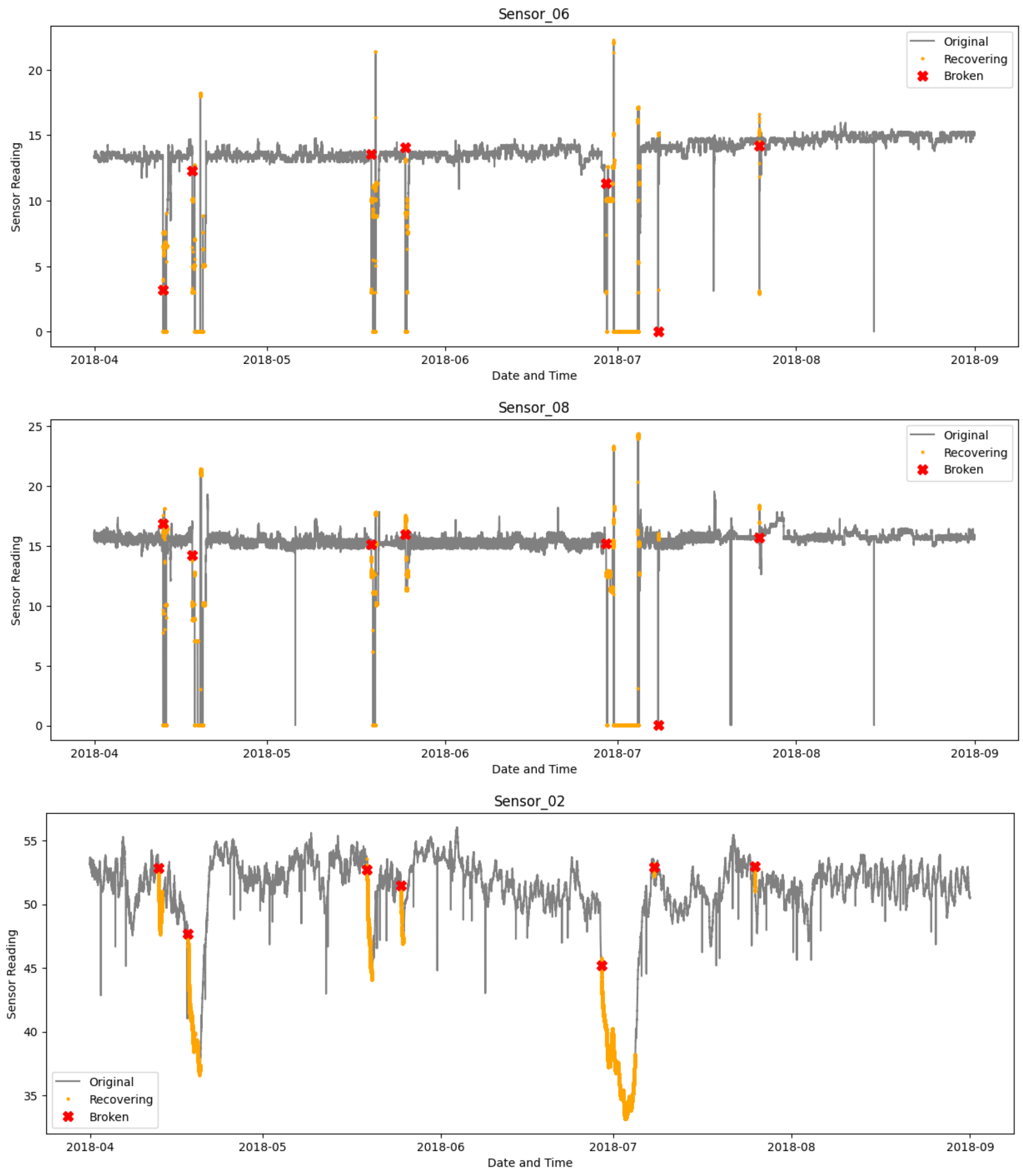


Figure 2: Sensor values: pressure, temperature and flow

4.2 Training

The training phase has been performed on Colab, exploiting free GPUs offered by Google.
It has been performed using a sliding window approach: each window contains past sensor data value to the current timestamp, and the model goal is to accurately predict if there is any breakage in the next one.
As main pump breakages can happen suddenly, a 10 minutes window has been used as past time frame, while a 20 minute window has been used as a target time frame for prediction.

4.3 EMlearn pipeline

As the microcontrollers used are constrained in terms of hardware resources, complex models or environments could not be exploited: optimizations such as pruning and quantization become necessary in order to overcome these problems.
In order to implement the trained algorithm into the microcontroller and perform inferences at runtime with such conditions, emlearn package has been used, following the steps:

1. Retrieve and process time series data
2. Train the model con Google Colab
3. Evaluate the model, retrain with different parameters if necessary
4. **Convert the model**
5. **Deploy on microcontroller**

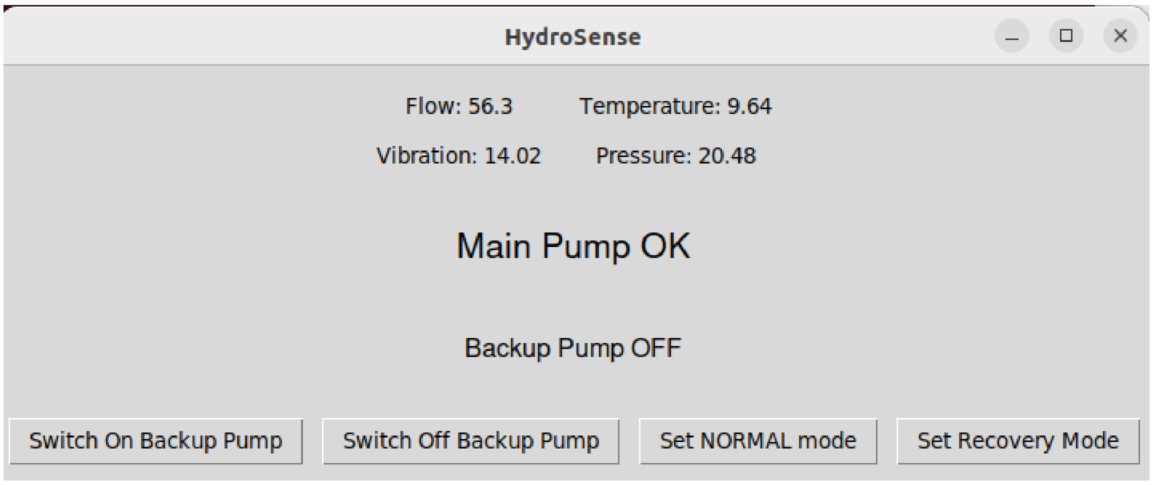
The result of these steps is a .c file that contains the weights of the trained neural network, along with some functions that will be used for prediction.

5 Application Deployment

The Desktop Application has been deployed using tkinter, chosen for its simplicity in development: the application runs on multiple threads, one for the GUI, and the other in which there will run the coap client, with a task for each observer.
When the application starts, it fetches the IP Addresses of the nodes from the server application, then it is able to observe the resources of the nodes, and buttons for activation of actuators become available.
In the next section it's shown its usage flow.

5.1 Usage

After starting the application server and connecting all the dongles, we wait the end of their registration phase (indicated by the light single led). After this, we can run the HydroSense application, which may look like this:



Flow: 56.3 Temperature: 9.64
Vibration: 14.02 Pressure: 20.48

Main Pump OK

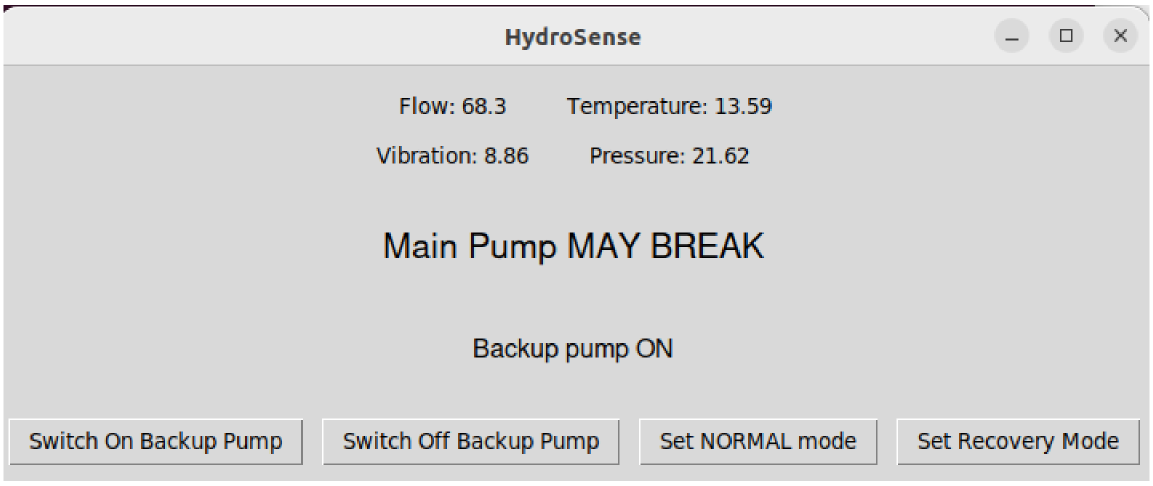
Backup Pump OFF

Figure 3: HydroSense user interface

As mentioned multiple times, all the sensor values are updated in real time with CoAP observers, so that we can monitor the status of the machine.

We also get notified about the main pump and backup pump states: we can not only monitor the state of the system, but interacting with the buttons we are able to switch on/off backup pumps or manually set to normal/recovery mode our machine, in case we detect some breakage that the algorithm did not find or viceversa.

After some time, the algorithm may detect impending breakdowns, so the state of the main water pump changes and consequently the label value on the user interface, which is notified by the observer. As a consequence, the backup pump actuator is switched on, as we can see:



Flow: 68.3 Temperature: 13.59
Vibration: 8.86 Pressure: 21.62

Main Pump MAY BREAK

Backup pump ON

Figure 4: HydroSense user interface - breakage

In case a false positive is detected or mantainance has been completed, the normal funcitons of the main pump can be restored via desktop application by pressing the "set NORMAL mode" button, obtaining the result:

Finally, backup pumps can be switched off by pressing the "Switch Off Backup Pump" button:

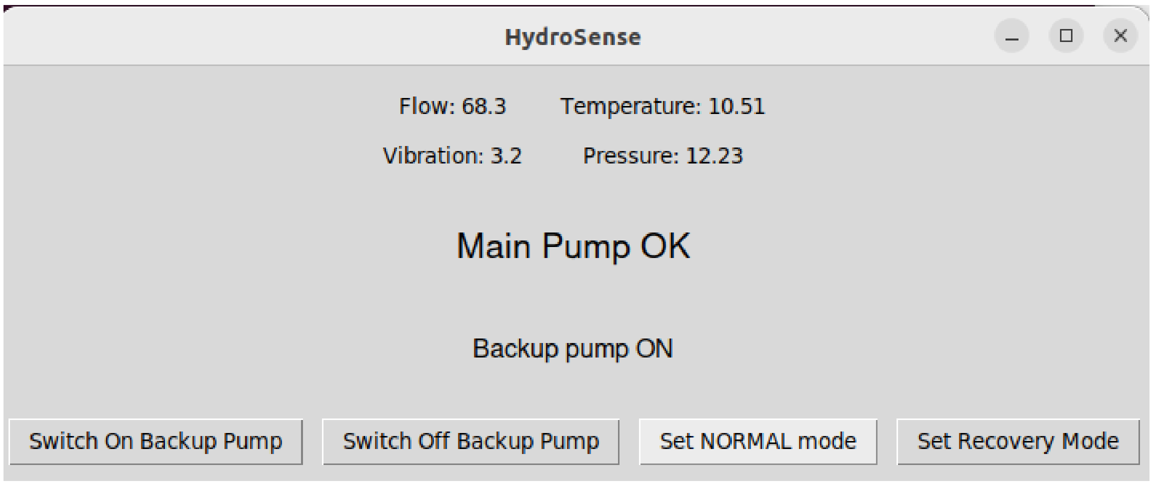


Figure 5: HydroSense user interface - manually setting to normal mode

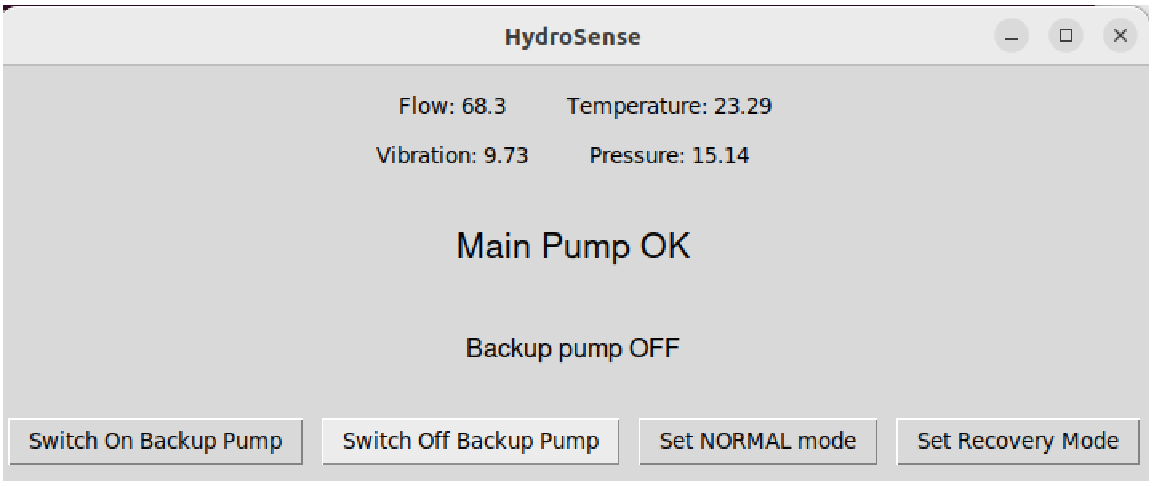


Figure 6: HydroSense user interface - manually switching off pumps