Implementation of a water quality remote monitoring system based on ISFET sensors and Deep Learning

Enrique Takiguchi

February 4, 2025

Resum– Els ISFET (Ion-Sensitive Field-Effect Transistors) són capaços de mesurar de forma compacta i eficient concentracions iòniques com el pH, d'interès en la monitorització de la qualitat d'aigua. Per aconseguir-ho, s'han de tenir en compte les interferències d'altres ions i condicions ambientals, ja que alteren el comportament dels ISFETS. Aquest projecte es centra en construir un sistema de monitorització que permeti als usuaris veure i analitzar les lectures ISFET en temps real.
El projecte també proporciona diferents models predictius per obtenir el pH a partir de les lectures ISFET basades en la regressió lineal i una implementació de Model-Agnostic Meta-Learning (MAML) amb un Multi-Layer Pereptron (MLP)

Paraules clau– MQTT, ISFET, MAML, ML, DL, MLP, Deep Learning, Django, React, Daphne, I2C,

Abstract–
ISFETs (Ion-Sensitive Field-Effect Transistors) are capable of compactly and efficiently measuring ionic concentrations such as pH, of interest in water quality monitoring. To achieve this, interference from other ions and environmental conditions must be taken into account, as they alter the behavior of ISFETS. This project focuses on building a monitoring system that allows users to view and analyze ISFET readings in real time.
The project also provides distinct predictive models to obtain the pH from the ISFET readings based on Linear Regression and an implementation of Model-Agnostic Meta-Learning (MAML) with a Multi-Layer Pereptron (MLP).
Keywords– MQTT, ISFET, MAML, ML, DL, MLP, Deep Learning, Django, React, Daphne, I2C,

1 Introduction

ISFETs (Ion-Sensitive Field-Effect Transistors) are transistors that change their gate voltage depending on the concentration of specific ions in a solution [1]. Ideally, an ISFET designed to detect H+ ions vary their response depending on the concentration of such ions and, if configured as a voltage follower, should output a source voltage that is directly related to the pH of the substance being measured. However, in practice, the relationship between the pH and the ISFET's output voltage is influenced by multiple factors, making it a non-linear function. The main factors affecting this behavior are the temporal drift of the ISFET interference from the activity of other ions in the solution. In this project, we focus on the interferences caused by sodium (Na+) and potassium (K+) ions.

Additionally, significant inconsistencies are observed in the offset, sensitivity and selectivity of different sets of ISFETs, further complicating the application of traditional predictive methods to address these non-linearities effectively.

1.1 Objectives

To address these challenges, the aim of this project is to develop a monitoring system capable of tracking target values (such as pH), ion interferences, and real-time reference

• E-mail de contacte: enrique.takiguchi@autonoma.cat
• Mención realitzada: Tecnologies de la Informació
• Treball tutoritzat per: Marc Ortega Gil (Departament d'Enginyeria de la Informació i de les Comunicacions)
• Curs 2024/25

pH values. The project also aims to use machine learning (ML)-based algorithms as a reliable method for predicting pH from ISFET measurements. Thus, the project is divided into two main components:

1. Design, training, and evaluation of predictive models based on machine learning to correct for non-linearities and interferences in ISFET measurements for accurate pH predictions.
2. Development of the IoT monitoring system. The transmission of electrochemical sensor readings on an Ethernet network by using MQTT protocols to track both pH and ion concentrations.

2 Workflow Plan

This project was divided according to the established objectives into:

- Pre-processing of the data: The goal of this phase is to prepare the collected data for the ML algorithms. This involves normalizing the data and transforming it into an appropriate format. This process is a prerequisite for developing the Machine Learning models.
- Definition and training of the ML models: This phase involves both constructing and training the ML models to achieve satisfactory performance. If adequate performance is reached, the ML models will be implemented for neuromorphic sensors.
- Neuromorphic Implementation: This phase focuses on translating the ML model into a form compatible with neuromorphic hardware.
- Development of the IoT system: Implementation of networked smart electrochemical sensors interconnected through IoT protocols such as MQTT.
- Development of a web interface: A platform to visualize real-time data and predict pH values. This interface will allow users to remotely monitor sensors, view prediction results, and assess the effectiveness of model adaptation.

Respect the initial plan, two major deviations were made:

- Due to the initial low performance of the ML models, several iterations of data pre-processing and model training were performed. Consequently, the time allocated to data processing increased significantly.
- Since the final evaluation of the ML models did not support a neuromorphic implementation, a decision was made to exclude it.

3 Materials and Methods

3.1 Predictive models

3.1.1 Brief Theoretical Introduction

In simple terms, a MOSFET (Metal-Oxide-Semiconductor Field-Effect Transistor) is a three-terminal device where one of the terminals (the gate) controls the channel through which electrical current flows between the other two terminals (the source and drain) [2]. CMOS (Complementary Metal-Oxide-Semiconductor) technology is the most common implementation of MOSFETs, where both n-channel and p-channel MOSFETs are combined to create efficient digital circuits. In MOSFETs, the current can be controlled by applying different voltage levels to the gate, which modulates the conductivity of the channel between the source and drain.

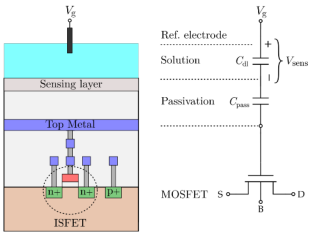

Figure 1: ISFET model used in the project

ISFET (Ion-Sensitive Field-Effect Transistor) devices, on the other hand, are a specialized type of transistor designed for chemical sensing. In ISFETs, the gate electrode is replaced during fabrication with an ion-sensitive membrane. This membrane is configured to respond to a specific type of ion, such as hydrogen ions (H+) for pH sensing. The ion concentration near the membrane alters the surface potential, which in turn controls the channel allowing current to flow between the source and drain. Thus, the electrical response of the ISFETs is modulated by the concentration of the specific ion that the membrane is designed to detect [1].

3.1.2 Related Work

Many methods have been proposed to compensate the non-linearities of the pH ISFET sensors. These methods can be divided into:

- On-chip methods: In general, these techniques work in the hardware level, by compensating on the readout electronics. For instance, on a 2007 paper, the output of two ISFETS was correlated eliminate the drift [3]. Other research focused on resetting the gate of the ISFETs [4] or correcting the effect of temperature the devices by regulating it [5].
- Off-chip methods: These are software-oriented solutions, that process the signals analytically or with Machine Learning based techniques. The thermal and drift effect compensation were done individually by differentiating the drift and biasing dynamically at the thermal point in a 2008 study [6]. Machine Learning techniques also proved to be useful by the simultaneous compensation of thermal and drift effect using non-linear autoregressive neural networks (NARX) [7] or by using both simple and complex ML algorithms [8]

In a research by Margarit et al. [9] different ML models were trained with a subset of the data described in Section 3.1.3. The predictive models developed and evaluated in this project can be considered an extension of this research,

with the use of a larger and more varied dataset and the subsequent use of a more polyvalent ML algorithm.

3.1.3 Experimental setup for data collection

Between the 20th of February of 2018 and the 16th of March 2020, a continuous reading of ISFET and pH values was made in a monitoring station. This station was located in the drinking water treatment plant (DWTP) of Sant Joan Despí, Spain. A probe was installed collecting water from the Llobregat's river. The probe included sensors with ISFETs designed to measure hydrogen (H+), sodium (Na+), and potassium (K+) ions, alongside a reference electrode. The ISFETs were fabricated using microelectronic technology, with modifications to make them selective to Na+ and K+ using specialized polymeric membranes [9]. These sensors were included given that Na+ and K+ were demonstrated to be the main interferences for pH measurements [10] [11].

During the whole data collection process, six different combinations of ISFET sensors were utilized, meaning that in five different occasions, the ISFET sensors were replaced by other sets. This leads to six different data distributions partitioned in six different time intervals (Table 1).

As can be seen in Figure 2, there are clear changes in the distributions of the ISFET readings between different tasks. These changes can even be appreciated inside the same tasks. With the sudden changes that were explain earlier in this section. Note that some periods have no data and are plotted as a straight line between the last value before the break and the first after. Examples of these periods are the one contained between 2018-10-15 and 2018-12-04 or between 2019-08-06 and 2020-02-19.

Task	Initial date	Final date
1	2018-02-20	2018-04-13
2	2018-04-13	2018-06-13
3	2018-06-13	2018-12-04
4	2018-12-04	2019-07-16
5	2019-07-16	2020-02-19
6	2020-02-19	2020-03-16

Table 1: Intervals for the different tasks

3.1.4 Data curation

To maximize the effectiveness of predictive algorithms, it is essential to curate the collected data and apply the necessary changes to assert consistency on the datasets.

As an initial step, a decision regarding the inputs of the predictive algorithms must be made. Since the ISFETS show a time-varying behaviour, it is decided that entering values corresponding to single timesteps would not be sufficient, and that a context window should be used as input. Subsequently, the predictive models will work with an input that will consist of 128 samples of each of the variables, thus, each input will consist of $128 \cdot 3 = 384$ values, corresponding to the previous 127 samples plus the current sample of all channels: H+, Na+ and K+.

To compromise between redundancy and information, we choose an interval between samples of 5 minutes, given that the change rate of the pH and ISFETs is relatively slow, and in a gap of 5 minutes, we don't expect significant changes in the level of either variables. Since generally the dataset presented a higher sampling rate, the dataset is down-sampled to match the chosen period of 5 minutes averaging the readings. This means that the context window will be of 640 minutes (hours and 40 minutes).

Since, during the data collection process, many unwanted periods were recorded—such as instances where the sensors were cleaned or recalibrated—it is necessary to identify and condition these occurrences. These periods are characterized by sudden changes in the sensor readings. To detect such occurrences, the dataset is time-differentiated to identify elevated peaks, which correspond to abrupt jumps in the readings. Each abnormality is then evaluated individually to determine whether it corresponds to cleaning or recalibration.

For cleaning events, intervals of inconsistent data are identified. Since predictions based on inconsistent data are not of interest, these intervals are removed from the dataset. For recalibration events, an offset is applied to the readings recorded prior to the recalibration to ensure consistency.

In some cases, it was suspected that the water that was being measured flowed with a delay in the ISFET sensors with respect to the reference pH sensor due to residues being accumulated in the probes. This could be appreciated in the dataset as progressive temporal delays of the ISFET measurements. To solve this, we used the differentiation from the previous step and searched for relative maximums and minimums in pH and H+ ISFET readings. Since we expect the minimums and maximums to match, the data was compressed / expanded in different intervals to align the minimums and maximums.

ISFET sensors accumulate charge during their lifespan, which contributes to a gradual increase in the magnitude of the readings. This gradual increase hinders the predictive capability of the models, since these accumulations can be interpretated as actual pH level rises. Additionally, as previously mentioned, ISFETS are characterized by having offsets that are not directly related to the ionic concentration the measure but instead they are the result of a base voltage from the fabrication and initial configuration. This offset and gradual increase don't have a meaningful value for our intended objective of pH prediction and instead represent unwanted characteristics of the data.

Since both characteristics present very low frequency components, a digital high-pass filter is designed to block these unwanted signal features. After experimenting with different filter configurations, it is determined that a Butterworth filter of order 2 and a cutoff frequency of $2\mu Hz$ (corresponding to roughly 6 days) implemented with scipy.signal from the SciPy library of Python.

The data contains many outliers that alter the distributions of the data, hence for each task and variable we filter all values that fall 4 standard deviations higher or lower than the mean. Finally, we standardize each task and variable to assert a mean of 0 and standard deviation of 1.

3.1.5 Linear Regression model

One of the simplest predictive algorithms is a linear regression model, which lies at the intersection of statistical methods and machine learning (ML) [12]. The simplicity

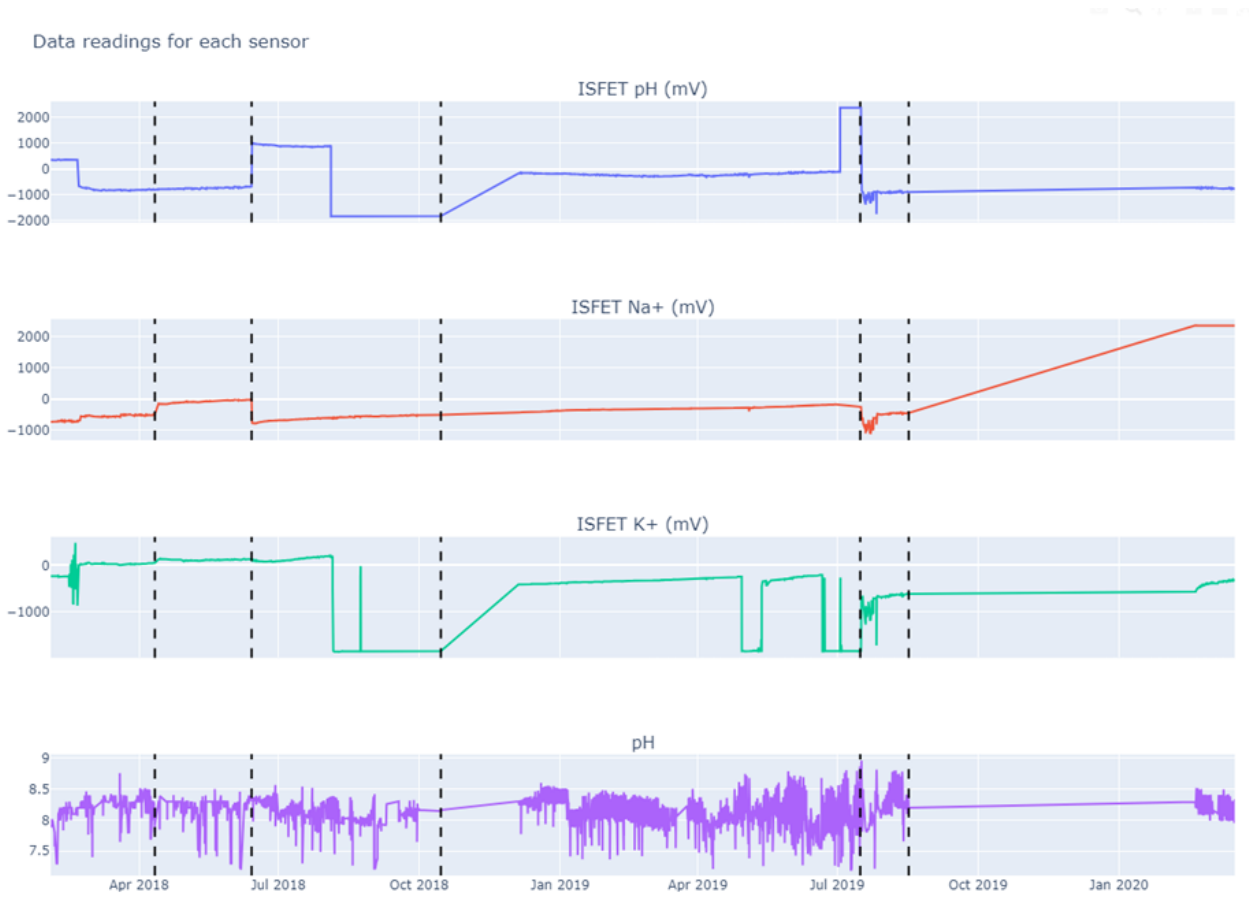

Figure 2: Raw data collected from Sant Joan Despí DWTP between the 20th of February 2018 and the 16th of March 2020

of linear regression makes it highly interpretable and computationally efficient, as it avoids the complexity and resource requirements characteristic of most advanced ML algorithms[13]. Although linear regression cannot model the non-linear behaviors of ISFETs, it enables fast and straightforward predictions, reducing the risk of overfitting caused by overly complex models and mappings.

The linear regression model was implemented using the numpy.linalg module from the Python library NumPy. Mathematically, the model can be expressed as:

$$\hat{y} = b + \sum_{c=1}^{3} \sum_{i=1}^{128} w_c(i) x_c(i) \tag{1}$$

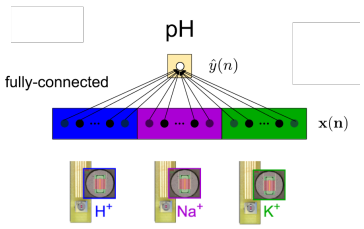

Figure 3: Structure of the Linear Regression model

3.1.6 MLP model

In contrast to linear regression, a multi-layer perceptron (MLP) incorporates multiple layers of neurons (nodes) between the input and output layers, which can vary in both depth (number of layers) and width (number of neurons per layer) [14]. Conceptually, an MLP can be considered as a stack of layers of linear regression models (multiply-accumulation nodes) including a posterior non-linear activation function, where the output of each layer serves as input to the next layer. This architecture enables MLPs to capture more complex, non-linear input-output relationships, making them better suited for modelling the intricate behaviours of ISFETs and other systems with non-linear characteristics.

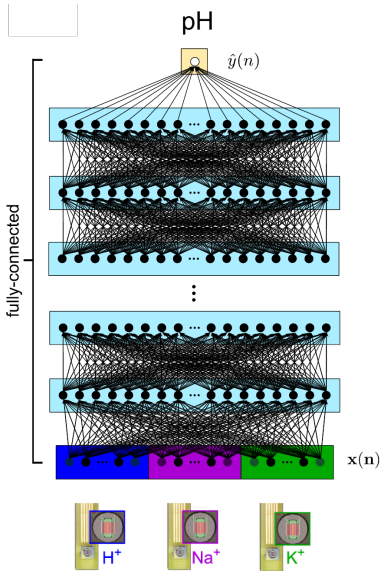

Figure 4: Structure of the MLP neural network

3.1.7 MAML Algorithm

In standard machine learning techniques, a model (such as a multi-layer perceptron, or MLP) is trained to predict outputs based on inputs under the assumption that the training and test data follow the same distribution. This assumption works well when the input-output relationships remain consistent across tasks or domains. However, when data comes from multiple distributions (e.g., tasks with different underlying patterns), a model trained on one distribution often struggles to generalize to unseen distributions. This is because the model has not been explicitly trained to adapt to new distributions, and thus it fails to correctly map inputs to the corresponding outputs when encountering unfamiliar data.

Model-Agnostic Meta-Learning (MAML) addresses this challenge by training models in a way that prepares them to quickly adapt to new distributions or tasks with minimal additional training. Instead of learning a single fixed set of parameters, MAML trains the model's parameters to be highly adaptable, enabling the model to efficiently learn new tasks from only a small amount of data [15].

MAML works in the following way:

- Objective: MAML aims to train a model with a set of initialization parameters that enable it to quickly adapt to new tasks using only a few gradient updates.
- Training Process: MAML assumes a set of tasks, each with its own data distribution but with some underlying relation. The training process involves two main steps, repeated iteratively
  - Inner Loop: A specific task is sampled, and the model's parameters are updated using a few gradient steps on the training data of that task. This step simulates task-specific learning.
  - Outer Loop: After the inner loop, the updated model is evaluated on the test data of the same task. The gradients from this evaluation are used to update the original parameters (before the inner loop) to optimize the model's initialization for adaptability across tasks.
- Outcome: Through repeated iterations of the inner and outer loops across many tasks, MAML learns a model that is not highly specialized for any single task but is primed to adapt quickly and effectively to new tasks with minimal additional training.

For our MAML implementation we divide the 6 tasks in 4 training tasks, a validation task and a test task. Additionally, each task is further divided in an "inner-loop" set and a "outer-loop" set. The training will consist of a number of epochs where an inner and outer loop is performed for each training task on each epoch.

- On each inner loop of each individual training task, the model is cloned, and the cloned model is trained with the task inner loop data. The data is divided into randomized batches to avoid the batches being temporally ordered. The model is trained through a number of inner steps, where on each step a batch is trained.
- After the inner step training, the cloned model predicts the data from the outer loop and corresponding gradients are generated, which are then applied to update the original model.

These two steps are repeated for every training task on each epoch, updating the model on each step towards a generalized model. To validate and test the model, only the inner loop is performed, while the loss from the outer loop prediction is taken as the validation or test loss.

After defining the MLP and the MAML algorithm, a hyperparameter search is performed to find an optimal configuration of the model. After the search, the MLP is set to have seven hidden layers with 211 nodes per layer, which translates into 394k learnable parameters. The model is trained with AdamW with a learning rate of 2.5e-5,

3.2 Monitoring setup

As a starting point, our setup has an array of chips containing either single or multiple ISFET sensors. In contrast the previous setup contained only a single ISFET sensor for each measured ion concentration. The chips receive the different readings of the ISFETs as voltages which are digitalized and sent to a Field-Programmable Gate Array (FPGA).

The communication between the chips and the FPGA is done via a bus controlled by Inter-Integrated Circuit (I2C) communication. I2C is a synchronous, serial communication protocol which allows us to communicate the different chips by assigning them addresses. It operates in a master-slave configuration, where the FPGA takes the master role handling the timing and signals and the chips act as slaves sending data to the FPGA [16].

From this setup, this project aims to provide an IoT infrastructure that is able to provide the data received by the FPGA to any user with access to a website where this data will be displayed.

Working on this framework, the FPGA, equipped with processing capabilities and an Ethernet connection, was programmed to save the data to a MicroSD card while publishing the data over Ethernet using the Message Queue Telemetry Transport (MQTT) protocol. MQTT enables communication between devices within the same network by employing a publish-subscribe architecture.

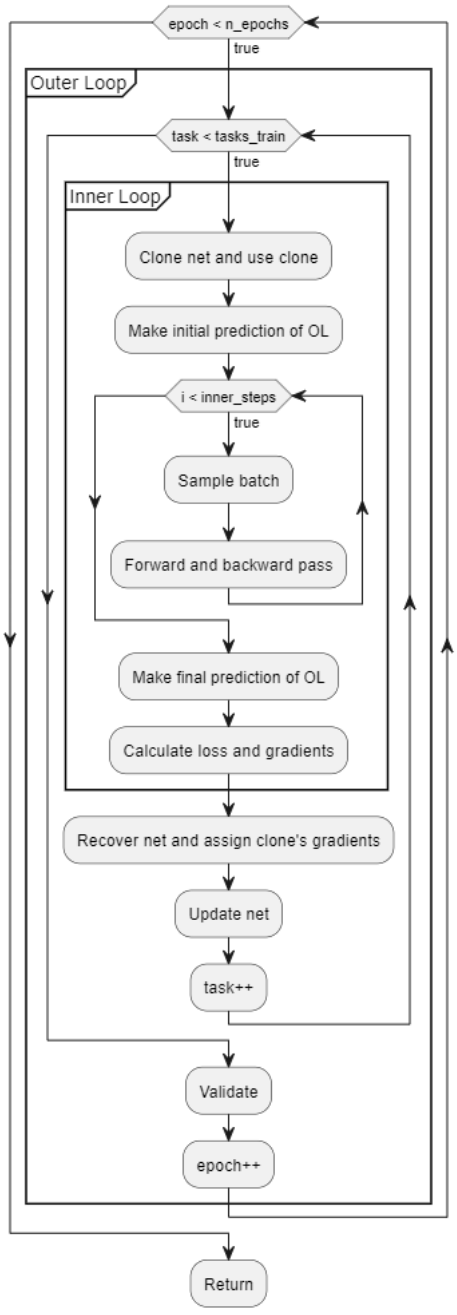

Figure 5: Diagram of MAML training

In this setup, the FPGA acts as a publisher, sending data to an MQTT broker (running on a PC connected to the same network). The broker is responsible for identifying and delivering messages to all devices subscribed to the topic associated with the message.

To ensure secure communication, the MQTT broker requires authentication, meaning both the publisher (FPGA) and subscribers must provide valid credentials (username and password) to send or receive messages. On the publisher side, the FPGA uses the Mosquitto libraries to implement MQTT communication.

Each message contains the following information:

- Timestep: Elapsed time (in nanoseconds) from the start of the readings to the time of the current reading
- Board Type: The boards can either be iAqua(which supports multiple ISFETs) or IMB (which supports single a ISFET) or pH
- Chip: Unique integer identifying the chip
- Sensor: Identifies the ISFET or pH
- Unit: Specifies the unit of the value
- Value: Magnitude of the reading recorded by the sensor

The MQTT messages are then received by a PC which will act as our server to handle both the data received from the sensors and the requests from the website. The server collects all the messages and saves them in a local database managed with MariaDB.

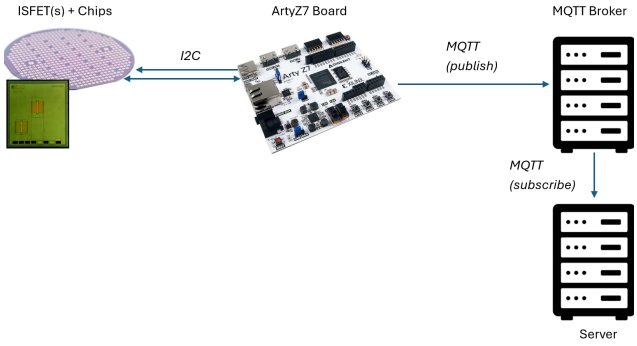

Figure 6: Setup of the monitoring scheme

The server backend is developed with Django. The choice of Django as the framework was made because it is based on Python and offers a wide range of built-in features, such as an ORM, authentication, and admin panel, which align well with the intended server functionalities. However, Django by itself only supports HTTP requests and is primarily synchronous when running with WSGI (Web Server Gateway Interface).

To enable asynchronous capabilities required for features like live monitoring via WebSockets, Django is run through Daphne, which is an ASGI (Asynchronous Server Gateway Interface) server. ASGI allows Django to support asynchronous features, such as WebSockets, alongside traditional HTTP handling. This setup is crucial for handling

real-time communication, making Daphne a key component in enabling WebSocket-based live updates in the system [17].

The server's frontend is developed using React, a robust JavaScript library chosen for its flexibility, efficiency, and ability to create dynamic, responsive user interfaces. React's component-based architecture allows for modular and scalable development, while its extensive ecosystem supports the integration of tools for real-time WebSocket communication, advanced data visualization, and interactive design.

To enhance the website's presentation and usability, the UI library Bootstrap is employed, offering pre-styled components and responsive layouts. The combination of React with Bootstrap provides a user-friendly interface which is at the same time able to handle live data visualization and interactive features.

The website will be structured the following way:

- Main page where the different views are introduced (Appendix A.1)
- Monitoring page where readings are shown in real time (Appendix A.2)
- Prediction page where the predictions of the different algorithms are shown (Appendix A.3)
- Analytics page where different a simple analysis of the data is shown

The monitoring page is the most important and critical component because it must receive and plot real-time data while determining when to send stored data from the database. The server collects data from multiple ISFET sensors, but for efficiency, only one reading per ion type (H+, Na+, K+) is selected to be sent to the monitoring page. Thus, the server sends the following four values to the clients:

The server collects data from multiple ISFET sensors, but for efficiency, only one reading per ion type (H+, Na+, K+) is selected to be sent to the monitoring page. Thus, the server sends the following four values to the clients:

1. One ISFET reading each for H+, Na+, and K+.
2. The reference pH value.

To maintain synchronization, only complete data sets (i.e., readings that include all required values for a specific timestamp) are sent to the client. As the server receives sensor data and stores it in the database, it checks whether all required values for a given timestamp are available. If so, the server immediately sends the complete set to the client.

This communication is implemented using WebSockets, which allow a persistent, bidirectional connection between the server and the client. Unlike AJAX polling, where repeated HTTP connections are established and terminated, WebSockets keep a continuous connection open. This design significantly reduces latency, which is crucial for real-time updates. The ISFET sensors have a sampling rate of 2 ms, meaning the server receives a constant stream of new data. However, this does not mean that the server transmits data to clients at the same high frequency. Instead, the implementation throttles the data transmission to 1 Hz (one reading per second), providing a more manageable update rate for the client while maintaining efficiency and usability.

Since the website's frontend and backend operate independently, the backend is responsible for processing the WebSocket messages. To facilitate communication, the frontend proxies WebSocket connections to the backend, which is possible because they run on different ports. Additionally, WebSocket connections are secured using TLS (Transport Layer Security), making the system more robust against potential security threats.

4 Results

4.1 Evaluation Metrics

The predictive models will be evaluated in terms of RMSE and R2. Evaluating with these two metrics allows us to quantify the quality of the fittings in terms of the absolute differences (RMSE) as well as the proportion of variance explained by the model (R2).

$$\text{RMSE} = \sqrt{\frac{\sum_{i=1}^{N} (y_i - \hat{y}_i)^2}{N}} \tag{2}$$

$$\text{R}^2 = 1 - \frac{\sum_{i=1}^{N} (y_i - \hat{y}_i)^2}{\sum_{i=1}^{N} (y_i - \bar{y}_i)^2} \tag{3}$$

Where N is the number of samples, y are the true values $\hat{y}$ are the predictions and $\bar{y}$ is the mean value of the true values y.

Its important to highlight that, as can be deduced from the Equations 2 and 3, RMSE indicates higher fitting with lower values, with a perfect fitting of RMSE=0, while R2 indicates higher fitting with higher values, with a perfect fitting of R2=1. An R2 of 0 would also indicate a fitting equivalent to just predicting the average of the true values, meaning a negative R2 show a considerably bad fitting.

4.2 Task considerations

Tasks one and two were discarded for the training since the pH was rounded to the first decimal, which reduced significantly the information and hindered gravely the predictability of the tasks. As can be seen, there many periods inside tasks 3 and 5 were also discarded because the data lacked consistency, and the predictability of the data was well below the optimal. The ISFET readings in Task 5 showed a great temporal delay respect the pH signal, and despite attempts of correcting it, the quality of the data was still inadequate to be able to predict the pH. For this reason Tasks 3,4 and 6 were used as training tasks and Task 5 was used as validation / test task, despite of the suboptimal quality.

4.3 Hyperparameter search

To find adequate parameters for the MAML model, a hyperparameter search was performed using 50 different combinations of hyperparameters. With each combination of hyperparameters, the MAML model was trained for 200 epochs to maximize the R2 loss. The following hyperparameters resulted in the best R2:

- Learning rate of the inner loop $1 \cdot 10^{-4}$

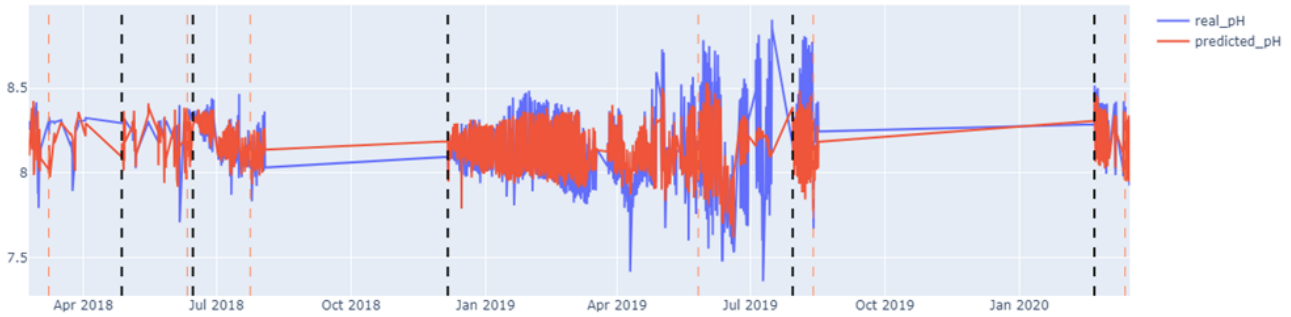

Figure 7: MLP predictions predictions for all dataset trained with MAML

- Learning rate of the outer loop $5 \cdot 10^{-3}$
- Batch size: 128
- Inner steps: 10
- Adam Betas: 0.9 and 0.999
- Dropout: 0.0735
- Weight Decay: 0.0172
- Batch normalization: Yes

4.4 Model Evaluations

The following tables present the results of the RMSE and R2 metrics for the tasks considered (3-6)

Task	RMSE (IL)	RMSE (OL)	R2 (IL)	R2 (OL)
3	0.14	0.33	0.63	0.07
4	0.09	1.54	0.51	-9.06
5	0.12	0.59	-0.81	-4.81
6	0.14	1.53	0.64	0.41

Table 2: RMSE and R2 for the different tasks predicted by the Linear Regression model

As can be appreciated in Table 2, Task 5 has notably worse fittings than the rest of tasks, even having a negative R2. We can also see in both RMSE and R2 that the inner-loop (IL) predictions are considerably better than the outer-loop (OL) data. This is because as explained because the IL data was used as training while the OL was used as test. Based on this, we can conclude that the Linear Regression model has an important overfitting factor, highlighting the need to use more complex models such as MLP which are able to make a better generalization. Despite of this, the results of tasks 3, 4 and 6 indicate that the Linear Regression model has a relatively good predictive capability despite its simplicity.

Task	RMSE (IL)	RMSE (OL)	R2 (IL)	R2 (OL)
3	0.24	0.39	0.88	0.69
4	0.42	1.55	0.75	0.25
5	1.62	0.99	-0.88	-2.85
6	0.3	0.37	0.9	0.47

Table 3: RMSE and R2 for the different tasks predicted by MLP trained with MAML

Table 3 shows better overall results than the ones obtained with the Linear Regression model. Task 5 on the other hand has worse RMSE and R2 values. This fact indicates that the generalized model that MAML trains is disadvantageous for the given task, but since the results in the Linear Regression model were considerably poor this is probably a sign of the unpredictability of the Task 5 data.

We now focus on one of the tasks (Task 3) to compare the results of the two considered predictive models. Analyzing the results of the linear regression model, we see for the most part, the oscillations of the pH are captured adequately. Still, the magnitude of the pH is not correct, given that the amplitude of the pH oscillations are slightly worse on the predicted pH. We can also observe little difference between the inner-loop (data before the red dashed line) and outer-loop predictions (after the red dashed line)

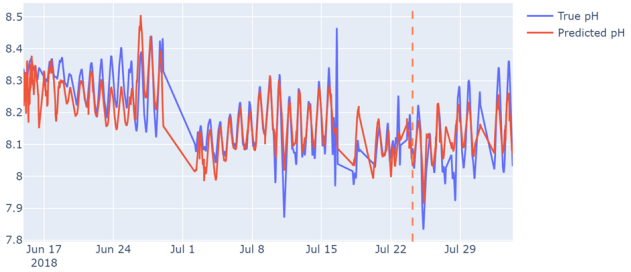

Figure 8: Linear Regression predictions for task 3

When looking at the predictions of the MLP trained with MAML we see similar results, but with a more accurate prediction of the amplitude of the pH oscillations. Still, both models fail to capture some of the features of the signal, which can also be seen on Figure 7 in the rest of tasks.

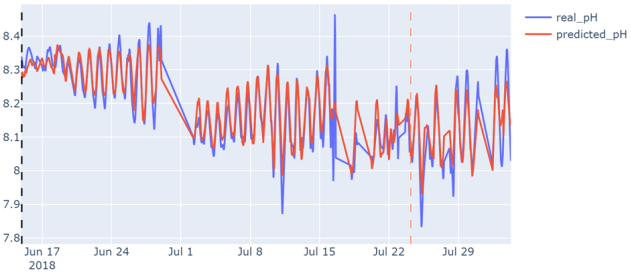

Figure 9: MLP predictions predictions for task 3 trained with MAML

5 CONCLUSIONS

In this project we have proven that the MAML algorithm generally overperforms simple Linear Regression models, with the exception of Tasks where data lacks the optimal quality. The under performance of the Linear Regression highlights the non-linearity of the ISFETs respect the pH references and is proof of the complexity of the ISFETs as well as the interferences the pH ISFET readings suffer due to other ionic concentrations.

The MAML performance has shown positive results but still leaves great margin for improvement. The low quality of some of the tasks' data and the little amount of available tasks are also factors that have hindered the results of the MAML evaluations.

Simultaneously, the proposed system effectively demonstrates a robust and efficient architecture for real-time monitoring and data analysis using ISFET-based sensors. By leveraging WebSocket technology, the system achieves low-latency, bi-directional communication between the server and the client, ensuring real-time updates and seamless data visualization.

The system design is also able to handle challenges as dealing with multiple ion sensors, the selectivity of relevant data and synchronization of the readings. The setup is also designed in a layered way, ensuring independence between the different communication protocols involved and adaptability to changes in the different nodes.

6 FUTURE WORK

Despite having obtained solid result on the MAML training, the margin of improvement in the evaluation metrics still calls for further investigation to improve the results. As previously mentioned, the MAML algorithm training would benefit from further training with a larger variety of tasks, expanding the six used for this project. Moreover, a more extensive hyperparameter search could be done to further optimize the hyperparameters of the MAML implementation.

Additionally, the experimentation of MAML implementation with other Deep Neural Network (DNN) architectures would ensure the MAML capabilities are maximized to obtain a more robust predictive model. Some of these architectures include Convolutional Neural Networks (CNN) or Recursive Neural Networks (RNN).

In terms of the monitoring setup, the WebSocket and MQTT communication scheme can also allow bidirectional communication, meaning that a client connected to a page, could send instructions to the FPGA to change configurations on the way the data is processed in the board. This would allow a more effective and efficient monitoring as it would allow for the possibility of remotely adjusting the board, instead of having to do it manually, as it is the case currently, since the ArtyZ7 board can only be configured by connecting through it with a PC.

6.1 Neuromorphic sensing and computing

Neuromorphic sensors and processors are devices that mimic the way biological systems process sensory information [18]. While they enable real-time operation and optimize data acquisition and AI computation since the very point of transduction, they also come with certain limitations. Neuromorphic sensors are used together with Spiking Neural Networks (SNNs), to extend the system efficency and energy autonomy, but compared to traditional deep learning (DL) models like typical Multi-Layer Perceptrons (MLPs), they generally exhibit lower accuracy [19].

Despite initially setting the neuromorphic implementation of DL algorithms as an objective for this project, the lower-than-expected model performance, combined with the additional accuracy degradation associated with an SNN implementation, led to the decision that further accuracy improvements are needed before proceeding to optimize energy autonomy through neuromorphic implementations of the neural network.

REFERENCES

1. Sakurai T and Husimi Y. Real-time monitoring of DNA polymerase reactions by a micro ISFET pH sensor. Analytical chemistry 1992; 64:1996–7
2. Locher R. Introduction to power MOSFETs and their applications. Fairchild Semiconductor (TM), Application Note 1998; 558
3. Premanode B, Silawan N, and Toumazou C. Drift reduction in ion-sensitive FETs using correlated double sampling. Electronics Letters 2007; 43:1
4. Shah S and Christen JB. Pulse width modulation circuit for ISFET drift reset. *SENSORS, 2013 IEEE*. IEEE. 2013 :1–4
5. Toumazou C, Shepherd LM, Reed SC, Chen GI, Patel A, Garner DM, Wang CJA, Ou CP, Amin-Desai K, Athanasiou P, et al. Simultaneous DNA amplification and detection using a pH-sensing semiconductor system. Nature methods 2013; 10:641–6
6. Chen D and Chan PK. An intelligent ISFET sensory system with temperature and drift compensation for long-term monitoring. IEEE Sensors Journal 2008; 8:1948–59
7. Amir MIS, Othman MR, and Syono MI. Drift compensation for pH ISFET senso r using NARX neural networks. Int. J. Eng. Technol 2018; 7:472–8
8. Sinha S, Bhardwaj R, Sahu N, Ahuja H, Sharma R, and Mukhiya R. Temperature and temporal drift compensation for Al2O3-gate ISFET-based pH sensor using machine learning techniques. Microelectronics Journal 2020; 97:104710
9. Margarit-Taulé JM, Martín-Ezquerra M, Escudé-Pujol R, Jiménez-Jorquera C, and Liu SC. Cross-compensation of FET sensor drift and matrix effects in the industrial continuous monitoring of ion concentrations. Sensors and Actuators B: Chemical 2022; 353:131123
10. Jimenez JC. Sensors quimics tipus ISFET. 1996
11. Alegret S, Bartrolí J, Jiménez C, Del Valle M, Dominguez C, Cabruja E, and Merlos A. pH-ISFET with NMOS technology. Electroanalysis 1991; 3:355–60

12. Su X, Yan X, and Tsai CL. Linear regression. Wiley Interdisciplinary Reviews: Computational Statistics 2012; 4:275–94

13. Rong S and Bao-Wen Z. The research of regression model in machine learning field. *MATEC Web of Conferences*. Vol. 176. EDP Sciences. 2018 :01033

14. Taud H and Mas JF. Multilayer perceptron (MLP). Geomatic approaches for modeling land change scenarios 2018 :451–5

15. Goerttler T, Müller L, and Obermayer K. Towards Efficient Gradient-Based Meta-Learning in Heterogenous Environments

16. Mankar J, Darode C, Trivedi K, Kanoje M, and Shahare P. Review of I2C protocol. International Journal of Research in Advent Technology 2014; 2

17. Williams M, Benfield C, Warner B, Zadka M, Mitchell D, Samuel K, Tardy P, Williams M, Benfield C, Warner B, et al. Twisted and Django Channels. Expert Twisted: Event-Driven and Asynchronous Programming with Python 2019 :365–71

18. Zeng M, He Y, Zhang C, and Wan Q. Neuromorphic devices for bionic sensing and perception. Frontiers in Neuroscience 2021; 15:690950

19. Schuman CD, Potok TE, Patton RM, Birdwell JD, Dean ME, Rose GS, and Plank JS. A survey of neuromorphic computing and neural networks in hardware. arXiv preprint arXiv:1705.06963 2017

APENDIX

A.1 Home Page

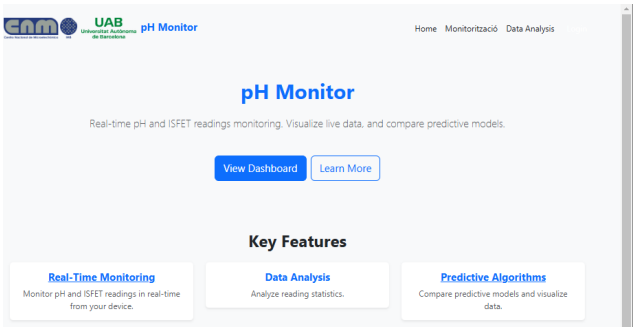

Figure 10: Home Page

A.2 Monitoring page

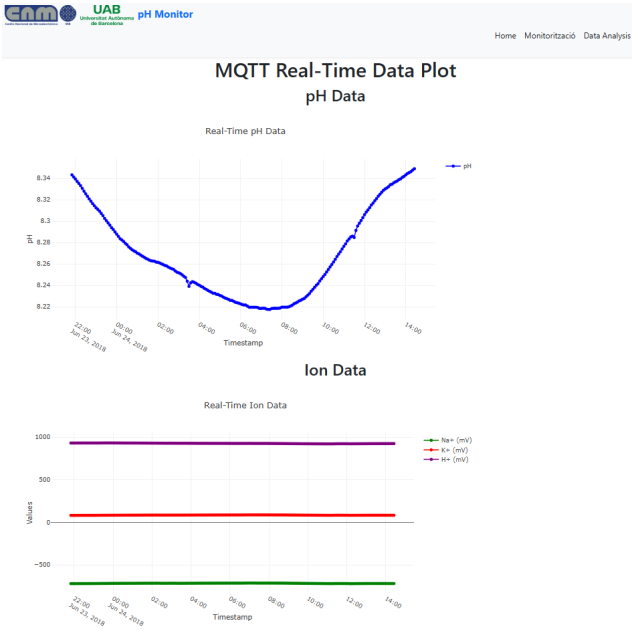

Figure 11: Monitoring Page

A.3 Predictions Page

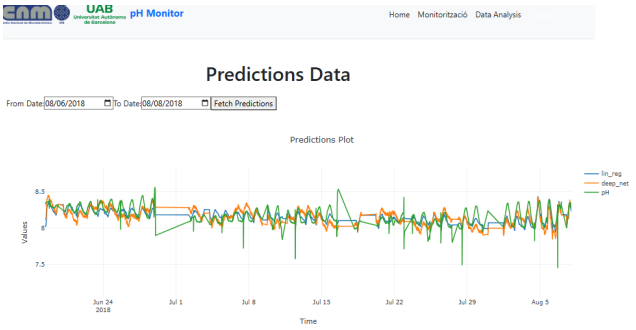

Figure 12: Predictions Page