IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Date of publication xxxx 00, 0000, date of current version xxxx 00, 0000.

*Digital Object Identifier 10.1109/ACCESS.2017.DOI*

# Detection of Wastewater Pollution through Natural Language Generation with a Low-Cost Sensing Platform

**KEVIN ROITERO[1], BEATRICE PORTELLI[1], GIUSEPPE SERRA[1], VINCENZO DELLA MEA[1], STEFANO MIZZARO[1], GIANNI CERRO[3](MEMBER, IEEE), MICHELE VITELLI[2,4] AND MARIO MOLINARA[2](MEMBER, IEEE)**

[1]University of Udine, Italy (e-mail: (kevin.roitero, stefano.mizzaro, giuseppe.serra, vincenzo.dellamea)@uniud.it, portelli.beatrice@spes.uniud.it )
[2]Department of Electrical and Information Engineering, University of Cassino and Southern Lazio, 03043 Cassino, Italy
(e-mail: michele.vitelli@unicas.it, m.molinara@unicas.it)
[3]Dept. of Medicine and Health Sciences, University of Molise, 86100 Campobasso, Italy (e-mail: gianni.cerro@unimol.it)
[4]Sensichips s.r.l., 04011 Aprilia, Italy (e-mail: michele.vitelli@sensichips.com)

Corresponding author: Kevin Roitero (e-mail: kevin.roitero@uniud.it).

This work was partially supported by: Horizon Europe research and innovation program under grant agreement "RHE-MEDiation - Responsive Hub for long term governancE to distress the MEDItarrean seA from chemical polluTION" with (GA 101113045), by the REACT-EU project “Data-Driven Multiutility Grid: Supporto alle Decisioni per Garantire la Sostenibilità dal Real Time al Lungo Termine” with “PON 2014-2020 AZIONE IV.6 GREEN”, and it was carried out within the Agritech National Research Center and received funding from the European Union Next-GenerationEU (PIANO NAZIONALE DI RIPRESA E RESILIENZA (PNRR) – MISSIONE 4 COMPONENTE 2, INVESTIMENTO 1.4 – D.D. 1032 17/06/2022, CN00000022). Disclosure: Michele Vitelli works as a Software Engineer in the R&D area of Sensichips s.r.l, the company which produced the chips used to collect the data employed in this work.

**ABSTRACT** The detection of contaminants in several environments (e.g., air, water, sewage systems) is of paramount importance to protect people and predict possible dangerous circumstances. Most works do this using classical Machine Learning tools that act on the acquired measurement data. This paper introduces two main elements: a low-cost platform to acquire, pre-process, and transmit data to classify contaminants in wastewater; and a novel classification approach to classify contaminants in wastewater, based on deep learning and the transformation of raw sensor data into natural language metadata. The proposed solution presents clear advantages against state-of-the-art systems in terms of higher effectiveness and reasonable efficiency. The main disadvantage of the proposed approach is that it relies on knowing the injection time, i.e., the instant in time when the contaminant is injected into the wastewater. For this reason, the developed system also includes a finite state machine tool able to infer the exact time instant when the substance is injected. The entire system is presented and discussed in detail. Furthermore, several variants of the proposed processing technique are also presented to assess the sensitivity to the number of used samples and the corresponding promptness/computational burden of the system. The lowest accuracy obtained by our technique is 91.4%, which is significantly higher than the 81.0% accuracy reached by the best baseline method.

**INDEX TERMS** Water Pollution, Language Models, Causal Models, Low-Cost Sensors

## I. INTRODUCTION

The task of accurate environmental monitoring is a pressing worldwide issue which is bound to become increasingly more important in the near future. There are many aspects that should be kept under control and concern the quality of the air, soil, and water [1, 2]. In fact, their continuous monitoring would allow targeted and timely actions aimed at restoring optimal conditions following dangerous events such as the appearance of pollutants. In this context, monitoring wastewater (WW) is particularly important [3]. WW is the water that has already been used for some purpose (civil or industrial uses) and must be subjected to purification before being returned to the natural cycle. To function at their best and effectively, the purification systems must know a priori the type of substances mixed with the water. It follows that a purification system for water for industrial use will be different from a purification plant for water for civil use. Hence, there is a strong need for protocols to promptly detect incompatible substances, to guarantee the correct and effective operation of purification plants[4].

Currently, this is solved by organizing periodic monitoring activities at particular points of the water path, which are carried out by the control institutes in charge using specialized laboratory instruments. Although this is an effective method, the quality of the water between two consecutive checks is unknown, and the checks may be not frequent enough to promptly identify problems. The ideal solution would combine automated continuous and distributed early warning monitoring, alongside periodic manual checks carried out by the control institutes.

To solve the problems of cost and installation of a distributed and continuous monitoring system, it is necessary to resort to low-cost and IoT-ready systems [5], which are able not only to collect environmental data but also to process them relying on centralized data collection and elaboration points.

In this context, the data collected from the sensors need to be processed by an algorithm that is used to analyze and forecast the presence (or absence) of polluting substances in the WW. Current state-of-the-art systems for this task rely on machine learning algorithms such as decision trees [6, 7].

In this paper, we propose a novel system based on deep learning, and in particular on causal generative models developed for natural language tasks, for the detection and classification of pollutants in WW, starting from the data collected by a multisensory system based on SENSIPLUS (Sensichips srl, Pisa, Italy) [8]. Note that the present paper does not present the infrastructure necessary for data transport as any solution based, for example, on MQTT or message queuing protocols could be used for this purpose.

The effectiveness of the proposed classifier is tested against a set of state-of-the-art baselines on a dataset created in collaboration with Sensichips s.r.l. and made available to the scientific community [9]. Results show that the proposed methodology outperforms the baseline methods and its effectiveness allows for practical usage of the developed methodology.

## II. RELATED WORK

The monitoring of wastewater is a widely discussed topic in the scientific literature. In particular, several kinds of technologies contribute to developing sensors that discriminate and classify undesired substances to ensure an adequate water quality level. Some of the authors developed systems able to monitor both water and air thanks to the SENSIPLUS platform [10, 11, 12, 13]. The monitoring outputs can vary, ranging from a classification of the pollutants to a simple binary decision on the presence of contaminants in general. Precise solutions to specific problems are often preferred to the development of generic monitoring system that can work properly in very wide contexts. As an example, Lim [14] describes a system to detect pollutants in the WW framework, although the distinction between different substances is missing and the technologies appear outdated nowadays. A different approach is taken by Lepot et al. [15], where the presence of illegal connections in the sewage system is monitored using an infrared camera. Ji et al. [16] present an image processing system, intended to estimate the WW amount without taking care of the distinction among substances. The cameras adopted to acquire images do not suffer from sensors' corrosion problems but they require a high energy budget, thus making the system far from the low-cost condition. There are other cases where the classification accuracy is very high but the energy/cost constraints are not taken into account. This is the case of Pisa et al. [17], who developed a system to detect ammonium and total nitrogen based on another one that is more broadly designed to detect all components derived from nitrogen. Drenoyanis et al. [18] propose an interesting portable device to monitor sewer pumping station pumps in order to generate alarms whenever anomalies are detected. The system is surely of great interest, but it does not include any pollutant classification stage. In terms of processing techniques, to the best of our knowledge, this is the first work leveraging natural language processing techniques, and in particular causal models developed for natural language generation, for the task of detecting WW pollution. Nevertheless, in literature we can find examples of the usage of natural language processing techniques and language models for non-canonical tasks. Language models have been used in the medical domain after the application of a "reverse encoding" (i.e., translating codes back to their description) for the classification of diagnostic tests [19, 20, 21] and for diagnostic rule encoding [22]. Furthermore, they have been used with a similar technique for the task of human mobility forecasting [23, 24]. More in general, transformer-based models originally designed for NLP tasks have demonstrated successful applications in a wide variety of non-NLP tasks [25], including: images [26, 27, 28], videos [29, 30, 31], speech and audio recognition [32, 33], conversational systems [34, 35], recommender systems [36, 37], reinforcement learning [38, 39], graphs [40, 41], protein structure predictions [42, 43], autonomous driving [44, 45], and anomaly detection problems [46, 47].

## III. SYSTEM AT A GLANCE

The proposed system is end-to-end and contains hardware and software components, which are detailed in the following.

### A. HARDWARE

The hardware part of the acquisition chain can be seen in Figure 1 where the following components are depicted: the Smart Cable Water (SCW), that is the sensing element; the SENSIBUS cable, a proprietary one-wire cable that allows communication with and control of the SCW; and a Micro Control Unit with an onboard firmware for controlling SCW, gathering and transmitting data to the cloud.

The SCW is a low-cost multi-sensory proprietary system,[1]

[1] https://sensichips.com/smart-cable-water/

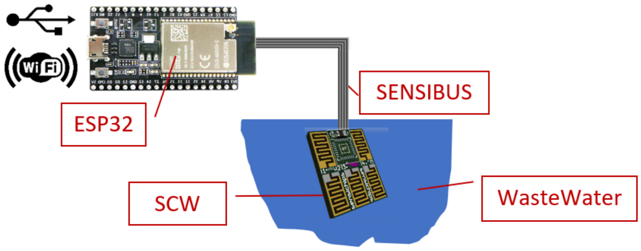
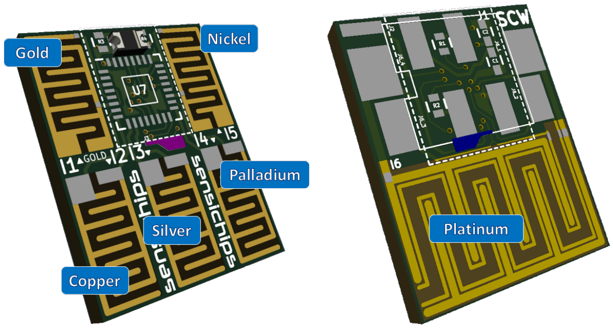


FIGURE 1: The hardware acquisition chain at a glance.



(a) Front (b) Rear

FIGURE 2: Smart Cable Water with its Inter Digitated sensors.

based on SENSIPLUS technology, capable of carrying out Electrochemical Impedance Spectroscopy (EIS) and Voltammetry measurements. SENSIPLUS is a proprietary technology of Sensichips s.r.l. developed in collaboration with the University of Pisa [8].

The SCW is equipped with multiple sensors consisting of 6 Inter Digitated Electrodes (IDE) realized on a base made of copper and functionalized with Gold, Oxide of Copper, Platinum, Silver, Nickel, and Palladium (see Figure 2). These sensitive elements, in conjunction with the EIS available on the chip, constitute the sensors adopted for the collection of the samples present in the dataset. In detail, measurements have been performed on five of these sensors. The IDE metalized with Platinum and Gold has been analyzed at two specific frequencies: 200Hz and 78KHz, while Copper, Silver, and Nickel only at 200Hz. Different stimulus frequencies allow the exploitation of different frequency responses since the interactions between the metals on the sensors and the pollutants vary according to this parameter. In total, 12 quantities proportional to the capacity and resistance of the IDEs were acquired. Table 1 reports the correspondence between IDEs and frequencies.

### B. ACQUISITION AND PRE-PROCESSING SOFTWARE

The software components of the elaboration chain can be seen in Figure 3 where the following components are visible: the C API implemented as firmware for the MCU, a Finite State Machine (FSM) for baseline acquisition and injection detection of substances, the classification system. The MCU's firmware controls the SENSIPLUS chip through the SENSIBUS channel, collecting raw data and transmitting it to the computational module. The computational module (that could be a workstation connected through USB or systems in the cloud connected with TCP/IP through Wi-Fi) is responsible for running the FSM and the classification system. The classification system is described in Section V-D, while the FSM is described in the following.

TABLE 1: Sensors used in the experiments.

| IDE | Acquisition Frequency | Integer Value Proportional to |
|---|---|---|
| Platinum | 78 kHz | Resistance |
| Gold | 78 kHz | Resistance |
| Platinum | 200 Hz | Resistance |
| Platinum | 200 Hz | Capacitance |
| Gold | 200 Hz | Resistance |
| Gold | 200 Hz | Capacitance |
| Copper | 200 Hz | Resistance |
| Copper | 200 Hz | Capacitance |
| Silver | 200 Hz | Resistance |
| Silver | 200 Hz | Capacitance |
| Nickel | 200 Hz | Resistance |
| Nickel | 200 Hz | Capacitance |

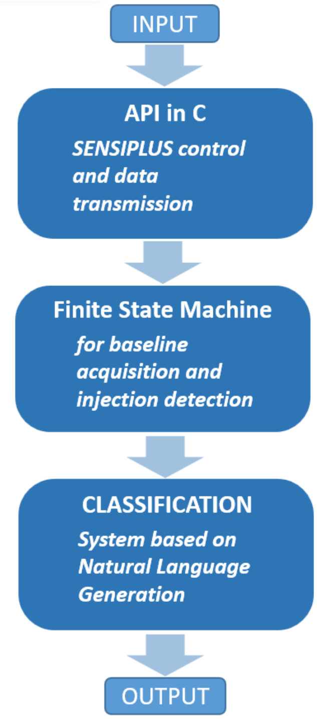
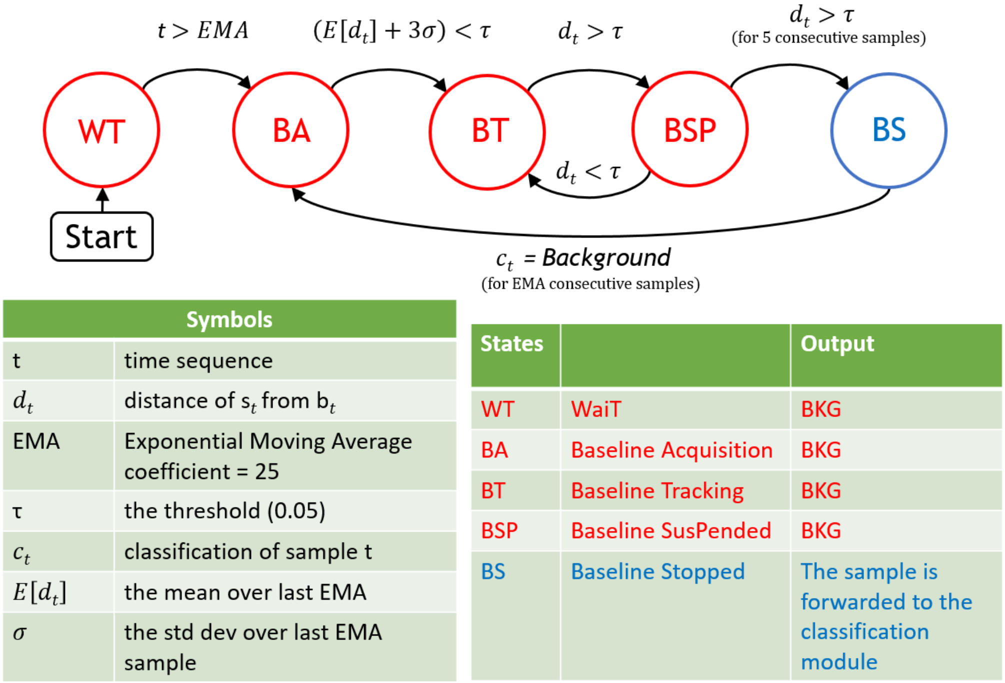

INPUT
API in C
SENSIPLUS control and data transmission
Finite State Machine
for baseline acquisition and injection detection
CLASSIFICATION
System based on Natural Language Generation
OUTPUT

FIGURE 3: The software chain at a glance.

The FSM is represented in Figure 4 and works in two steps:

- baseline extraction: a baseline signal is extracted to normalize raw data;
- forwarding decision: for each sample, the FSM decides whether to forward it to the classifier, also providing the injection time.

The FSM generates the baseline signal $b_t$ by an Exponen-


$t > EMA$
$(E[d_t] + 3\sigma) < \tau$
$d_t > \tau$
$d_t > \tau$
(for 5 consecutive samples)
WT
BA
BT
BSP
BS
$d_t < \tau$
Start
$c_t$ = Background
(for EMA consecutive samples)

| Symbols | |
|---|---|
| t | time sequence |
| $d_t$ | distance of $s_t$ from $b_t$ |
| EMA | Exponential Moving Average coefficient = 25 |
| τ | the threshold (0.05) |
| $c_t$ | classification of sample t |
| $E[d_t]$ | the mean over last EMA |
| $\sigma$ | the std dev over last EMA sample |

| States | | Output |
|---|---|---|
| WT | WaiT | BKG |
| BA | Baseline Acquisition | BKG |
| BT | Baseline Tracking | BKG |
| BSP | Baseline SusPended | BKG |
| BS | Baseline Stopped | The sample is forwarded to the classification module |

FIGURE 4: Finite State Machine.

tial Moving Average (EMA):

$$\boldsymbol{b}_t = \begin{cases} \boldsymbol{s}_t & t = 0 \\ \boldsymbol{b}_{t-1} & t > 0, S \in \{BS, BSP\} \\ \alpha \boldsymbol{s}_t + (1-\alpha)\cdot \boldsymbol{s}_{t-1}, & t > 0, S \in \{WT, BA, BT\}, \end{cases} \quad (1)$$

where $s_t$ are the sensors' raw data at time $t$; {WT, BA, BT, BSP, BS} are the possible states of the FSM and correspond respectively to {Wait, Baseline Acquisition, Baseline Tracking, Baseline Suspended, and Baseline Stopped}.

The $\alpha$ parameter in EMA is the reciprocal of EMA$_c$ that has been empirically set to 25. The normalized signal $\boldsymbol{f}_t$ forwarded by the FSM, is evaluated as

$$\boldsymbol{f}_t = \frac{\boldsymbol{s}_t}{\boldsymbol{b}_t}, \quad (2)$$

where $\boldsymbol{s}_t$ is the raw data collected from sensors and ~~the~~ $\boldsymbol{b}_t$ is the baseline signal computed as described by Equation 1. $\boldsymbol{f}_t$, $\boldsymbol{s}_t$, and $\boldsymbol{b}_t$ are $n$-dimensional vectors with $n$ equal to the number of sensors (see Table 1); the division in the equation is element-wise.

Thanks to this baseline, the system can mitigate sensor drift, environmental noise, signal spikes, and variability between sensors.

In the schema of the FSM reported in Figure 4, $t$ is the current time, $\tau$ is a threshold empirically set to $0.05$, $\boldsymbol{d}_t$ is the Euclidean distance between $\boldsymbol{f}_t$ and a vector of ones denoted as $\boldsymbol{u}$ in the feature space.

When $\boldsymbol{b}_t$ is equal to $\boldsymbol{s}_t$ in Equation 2, the $\boldsymbol{f}_t$ vector is the unit vector. As a consequence, the Euclidean distance has been computed with respect to the unit vector, and $\boldsymbol{d}_t$ equal to zero means that the baseline signal $\boldsymbol{b}_t$ is perfectly tracking the sensors signals $s_t$:

$$\boldsymbol{d}_t = \|\boldsymbol{f}_t - \boldsymbol{u}\|\,. \quad (3)$$

The WT state is conceived to "fill" the EMA; the BA state is reached automatically after EMA$_c$ samples. Through the BA state, the FSM starts to follow the signal waiting for good tracking. Good tracking is obtained by analyzing the distance with the baseline. Once the variability of the distance, computed as its mean plus three times the standard deviation, is below a given threshold $\tau$ (empirically established to $0.05$), the system can move to the BT state. The system will then check if a substance has been spilled in the water by checking when the current distance is greater than $\tau$. When the signal moves away from the baseline, the state becomes BSP for a while. Once the FSM moves to the BSP state, the system will check that the current distance remains above the threshold for five consecutive samples (BSP); otherwise, the system comes back to the BT state (to avoid confusing the spill of a substance with a measurement spike or noise). Finally, when the FSM reaches the BS state, the current normalized sample $\boldsymbol{f}_t$ is forwarded to the classification module.

### C. THE CLASSIFICATION MODULE

The proposed classification module is based on deep learning for natural language processing, and in particular on Transformer-based [48] models. We employ T5 [49], which is a large text-to-text language model pre-trained on a multi-task mixture of unsupervised and supervised tasks, the former being unsupervised de-noising objective tasks, while the latter being text-to-text language modeling objective ones.

For a complete overview of tasks and prompts please refer to Raffel et al. [49, Appendix Section].

The T5 model architecture is similar to that of a general Transformer model and it is composed of a stack of encoder blocks, which transform the input text into a latent representation, and a stack of decoder blocks which translate the latent distribution into a new output text. Each block comprises a self-attention module, optional encoder-decoder attention, and a feed-forward network. Since it is a text-generation model, it takes a textual input and generates a textual response.

We leverage the pre-training knowledge of the model, and adapt it to the task of substance prediction by textifying the raw sensors' input and training the model to produce a string stating the nomenclature of the pollutant present in the wastewater. All the parts of the classifications module are detailed in Section V.

## IV. DATA

In this section, the acquisition process for dataset creation is described in all its aspects.

### A. SUBSTANCES

The dataset used in this work aims to identify pollutants in WasteWater (WW), paying attention to spills of chemical compounds that could compromise public safety and/or the efficiency of purification systems. The acquisitions were made in the laboratory to simplify data collection. Measurements at experimental sites were excluded for two reasons: to ensure safety due to biological risks related to the presence of unknown bacteria or pollutants and to have controllable measurement conditions. In fact, the composition of WW is not stable over time, for example, due to atmospheric events such as rain. In detail, all the samples were acquired between 2019 and 2021 in two different laboratories in Poland and in Italy and were recently made public [9]. Table 2 reports the substances used. The dataset consists of 10 acquisitions for each substance (including the WW or background) and was obtained using the measurement protocol described below.

### B. THE DATASET

To create the dataset which is used in the experimental part of this paper, we employed a measurement system composed of a PC as control device, a micro-controller[2] which manages the communication between the PC and the multi-sensor system, and the SCW that acquires the sensor's signals. The different substances have been injected into a beaker containing 300ml of WW, where the SCW is immersed. A magnetic stirrer was used to simulate the movement of the WW, ensuring the same conditions for each measurement session. The rotation of the anchor, 25mm long, was set at 50 rpm in such a way as to reduce the presence of air bubbles that could make the measurement noisy (turbulent regime). The acquisition of the samples present in the dataset was carried out according to a measurement protocol divided into two steps:

1) initially 600 samples are collected in WW, in warm-up mode;
2) subsequently, the substance of interest was injected, and an additional 1000 samples are collected.

This protocol was repeated ten times for each substance (ten acquisitions for each substance). A total of 1600 samples are collected for each substance and for each acquisition, with an acquisition rate of about 1.6 seconds, for an overall run time of about 40 minutes for acquisition (Figure 5).

TABLE 2: Substances used in the experiments.

| Substance | Description |
|---|---|
| Wastewater | pH=7.4, conductivity=1.341mS |
| Acetic acid | $CH_3COOH$ |
| Acetone | $C_3H_6O$ |
| Ammonia | $NH_3$ |
| Ethanol | $C_2H_5OH$ |
| Formic acid | $CH_2O_2$ |
| Hydrochloric acid | HCl |
| Hydrogen peroxide | $H_2O_2$ |
| Phosphoric acid | $H_3PO_4$ |
| Sodium hypochlorite | NaClO |
| Sulphuric acid | $H_2SO_4$ |

## V. EXPERIMENTS

### A. PROBLEM FORMULATION

We have the measurements obtained on 10 substances plus the background substance (i.e., WW). To obtain definitive and stable results and avoid randomness and bias, our experiment utilizes the k-fold validation methodology. Specifically, we implement a 10-fold validation process whereby we rotate the experiment used for testing purposes from 1 to 10 and utilize the remaining experiments for training. This approach ensures that the samples from every experiment are used for testing once and that the overall performance metrics are averaged across all 10 test sets. As such, we are able to achieve a robust and reliable evaluation of our models' effectiveness. For each substance we have 600 samples collected in the so called "warm-up mode", which means that in that period of time the monitoring sensors have been exposed to WW only. Then, we also have 1000 samples collected after substance injection, which means that in this period of time the sensors have been exposed to both wastewater and the specific substance (if injected). Following the k-fold technique detailed above, for each substance (including WW), 9 acquisitions have been used as training set and 1 acquisition has been used as the test set. The effectiveness of the models is thus evaluated on the the average of the test acquisitions, and only on the 1000 samples collected after the warm-up phase. Details on the composition of the dataset folds can be found in Table 3.

### B. METRICS

To evaluate the ~~performances~~ of the model, we rely on different effectiveness metrics, aimed at measuring different

[2] ESP8266 https://www.wemos.cc/en/latest/d1/d1_mini.html

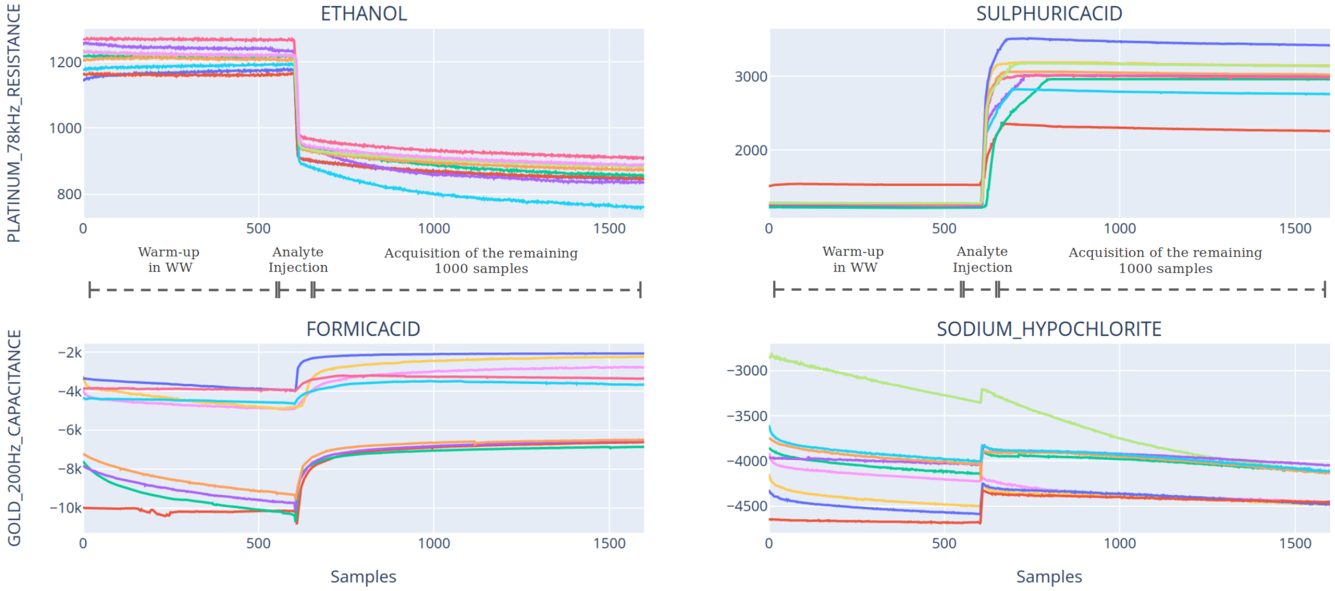


FIGURE 5: Example of acquired signals. The figure shows a subset of the signals present in the dataset. In detail, there are two features collected on four different substances. For each substance, there are nine experiments.

TABLE 3: Number of samples by substance. For each acquisition, 600 samples are collected in Wastewater and 1000 samples after a substance injection. The training set of each fold is composed of 9 acquisitions for each substance, while the test set comprises of 1 acquisition.

| **Substance** | **Training set** | **Test set** |
|---|---|---|
| Wastewater | $(600 \cdot 10 \cdot 9 + 1600 \cdot 9)$ | $(600 \cdot 10 \cdot 1 + 1600 \cdot 1)$ |
| Acetic acid | $(1000 \cdot 9)$ | $(1000 \cdot 1)$ |
| Acetone | $(1000 \cdot 9)$ | $(1000 \cdot 1)$ |
| Ammonia | $(1000 \cdot 9)$ | $(1000 \cdot 1)$ |
| Ethanol | $(1000 \cdot 9)$ | $(1000 \cdot 1)$ |
| Formic acid | $(1000 \cdot 9)$ | $(1000 \cdot 1)$ |
| Hydrochloric acid | $(1000 \cdot 9)$ | $(1000 \cdot 1)$ |
| Hydrogen peroxide | $(1000 \cdot 9)$ | $(1000 \cdot 1)$ |
| Phosphoric acid | $(1000 \cdot 9)$ | $(1000 \cdot 1)$ |
| Sodium hypochlorite | $(1000 \cdot 9)$ | $(1000 \cdot 1)$ |
| Sulphuric acid | $(1000 \cdot 9)$ | $(1000 \cdot 1)$ |
| Total samples | 158,400 | 17,600 |

aspects of the models effectiveness. We use the following notation for the set of correctly and incorrectly identified sampled:

- TP (True Positives): samples correctly identified as belonging to a substance of interest.
- TN (True Negatives): correctly identified negative samples, i.e., samples collected in WW.
- FP (False Positives): samples collected in WW, but classified as one of the 10 substances of interest.
- FN (False Negatives): samples collected in presence of substance of interest but classified as WW.

Furthermore, we use the following notation to denote the metric averaging method:

- $\text{metric}_\text{m}$: micro averaged metric; we aggregate the contributions of all classes to compute the average metric.
- $\text{metric}_\text{M}$: macro averaged metric; we compute the metric independently for each class and then take the average, hence treating all classes equally.
- $\text{metric}_\text{W}$: weighted metric; each class contribution is weighted by the relative number of samples available for such a class.

We also compute the following metrics:

- Accuracy, defined as

$$\text{Acc} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{FP} + \text{TN} + \text{FN}};$$

- Matthews Correlation Coefficient MCC for multi-classification (also called Rk statistic), defined as

$$\text{MCC} = \frac{(\text{TP} \cdot \text{TN}) - (\text{FP} \cdot \text{FN})}{\sqrt{(\text{TP} + \text{FP}) \cdot (\text{TP} + \text{FN}) \cdot (\text{TN} + \text{FP}) \cdot (\text{TN} + \text{FN})}};$$

- Precision, defined as

$$\text{Prec} = \frac{\text{TP}}{\text{TP} + \text{FP}};$$

- Recall, defined as

$$\text{Rec} = \frac{\text{TP}}{\text{TP} + \text{FN}};$$

- F1-Score, defined as

$$\text{F1} = 2 \cdot \frac{\text{Prec} \cdot \text{Rec}}{\text{Prec} + \text{Rec}}.$$

### C. BASELINE METHODS

A set of learning algorithms was selected and applied to the collected dataset to compare the proposed solution with standard Machine Learning techniques and obtain a reference baseline. The choice was made to have a sufficiently exhaustive representation of different approaches with different complexity. As a result, we adopted algorithms belonging to the following categories: boosting, bagging, tree-based, instance-based, kernel-based, Artificial Neural Networks, and ensemble classifies.

As for the boosting algorithm, we selected AdaBoost [50] with different types of weak classifiers: decision stump, J48 tree [51] and a more complex Random Forest [52]. For the Bagging [53] we selected REPTree, a simple tree learner that uses the information gain heuristic to choose an attribute and a binary split on numeric attributes (faster than C4.5) [54]. For decision tree-based algorithms, we have chosen Random Forest [52]. The classic k-nearest neighbors algorithm (KNN) [55] has been selected for the instance-based algorithms, Support Vector Machines (SVM) [56] for kernel-based algorithms, and a Multi Layer Perceptron (MLP) for the Artificial Neural Networks category algorithms [57]. Finally, a majority vote between MLP, KNN, SVM and RandomForest for ensemble-based algorithms.

The selected ML algorithms were preliminarily optimized through a grid search on their respective hyperparameters. Table 4 shows, for each algorithm, which hyperparameters were selected. The WEKA (Waikato Environment for Knowledge Analysis – version 3.8.6) implementation of these algorithms was used in the experimental phase [58]. The input of the ML algorithms is represented by the feature vector calculated in Equation 2 and therefore contains the instantaneous measurement of all the sensors identified in Table 4, following normalization with respect to the baseline.

### D. TEXTIFICATION OF THE INPUT

In contrast to the baseline methods, our proposed T5 model requires textual input to be trained for natural language generation. For this reason, we first need to define a methodology to describe the input features (i.e., the observations of the set of sensors) in natural language. To this aim, we rely on the so called "textification" (or prompting) of the input features, an approach that has been successfully applied in the medical domain, in particular in the automatic encoding and prediction of diagnostic texts [19, 20, 21], as well as in human mobility forecasting [23, 24].

This transformation essentially takes an array of floating point values corresponding to the input features and translate it into a text, which will be then the input of our model.

Our approach works as follows. First, let us recall that each measurement is made of 1600 timestamps, $t \in [1, 1600]$ indicating the warm-up phase where the sensors are exposed to wastewater only, injection time happening at $t = 600$, and $t \in [601, 1600]$ indicating the phase after injection, where sensors are exposed to wastewater and the substance.

For each acquisition (out of the 10 present in each training set), we sample two timestamps: $t_b$ in the warm-up phase, and $t_a$ after the injection phase. Then, we create a piece of text with the following pattern, for each of the sensors:

> the {capacitance/resistance} of {sensor}
> at {frequency} is {value}

By linking together the text for all the sensors and adding contextual information, we then create a text with the following pattern:

> at time $t_b$ ($600 - t_b$ before injection),
> the {capacitance/resistance} of {sensor}
> at {frequency} is {value},
> ...
> at time $t_b$ ($t_a - 600$ after injection),
> the {capacitance/resistance} of {sensor}
> at {frequency} is {value},
> ...

where $t_b$ and $t_a$ are the sampled timestamps, and the other variables (such as {capacitance/resistance} are depending on the specific sensor used. Let us make it clear by reporting a real full example, where $t_b = 582$ and $t_a = 1141$:

> at time 582 (18 before injection),
> the Resistance of Platinum at 78kHz is 0.0444,
> the Resistance of Gold at 78kHz is 0.0151,
> ...
> the Capacitance of Nickel at 200Hz is 0.002.
>
> at time 1141 (541 after injection)
> the Resistance of Platinum at 78kHz is 0.783,
> the Resistance of Gold at 78kHz is is 0.9439,
> ...
> the Capacitance of Nickel at 200Hz is 0.992.

We repeat the process for all the acquisitions present in our dataset. Note that the proposed methodology allows to sample multiple $t_b$ points to predict the substance present at $t_a$; in this case, the final prediction is obtained by taking the mode (i.e., majority voting) over the different predictions made by the model for the same acquisition.

### E. MODEL TRAINING AND INFERENCE

We develop our model using the PyTorch[3] and HuggingFace[4] frameworks. All the data and code used in the paper are made available at: (to be inserted upon acceptance.)

We rely on the `T5-base` model,[5] which is composed of an encoder and decoder stacks comprising 12 blocks each. Each block contains self-attention mechanisms, optional encoder-decoder attention, and a feed-forward network. The attention is of dimension 64, while embeddings are of dimension 768. The final model has about 220 million parameters.

[3] https://pytorch.org/
[4] https://huggingface.co/
[5] https://huggingface.co/t5-base

TABLE 4: The selected hyper-parameters for the various ML algorithms employed.

| Model | Sub Model | Parameter | Selected value | Description |
|---|---|---|---|---|
| K-NN | – | KNN | 100 | The number of neighbours to use |
| K-NN | – | algorithm | BallTree | BallTree algorithm for nearest neighbour search |
| K-NN | – | distance | Euclidian | The distance function to use for finding neighbours |
| SVM | – | kernelType | RBF | The type of kernel to use |
| SVM | – | cost | 1 | The cost parameter C for C-SVC |
| SVM | – | gamma | 0 | The gamma to use, if 0 then 1/max_index is used |
| AdaBoost | – | weightThreshold | 100 | Weight threshold for weight pruning |
| AdaBoost | Decision Stump | – | – | – |
| AdaBoost | J48 | confidenceFactor | 0.25 | The confidence factor used for pruning (smaller values incur more pruning) |
| AdaBoost | J48 | numDecimalPlaces | 2 | The minimum number of instances per leaf |
| AdaBoost | Random Forest | numIterations | 100 | The number of trees in the random forest. |
| AdaBoost | Random Forest | maxDepth | 0 | The maximum depth of the tree, 0 for unlimited. |
| MLP | – | hiddenLayers | 20 | This defines the hidden layers of the neural network |
| MLP | – | decay | TRUE | This will cause the learning rate to decrease |
| MLP | – | learningRate | 0.3 | The learning rate for weight updates |
| MLP | – | momentum | 0.2 | Momentum applied to the weight updates |
| Random Forest | – | numIterations | 100 | The number of trees in the random forest. |
| Random Forest | – | maxDepth | 0 | The maximum depth of the tree, 0 for unlimited. |
| Bagging | – | bagSizePercent | 100 | Size of each bag, as a percentage of the training set size |
| Bagging | – | numIterations | 10 | The number of iterations to be performed |
| Bagging | REPTree | minNum | 2 | The minimum total weight of the instances in a leaf |
| Bagging | REPTree | numFolds | 3 | Determines the amount of data used for pruning |
| Bagging | REPTree | noPruning | FALSE | Whether pruning is performed |

We initialized the model weights with the pre-trained ones of the original T5 model. To feed the textual input to the model we used the custom prefix “predict:”, and we used the strings “input:” and “target:” to discern between the model input and the target. We train the model on a Linux server equipped with 16x Intel(R) Core(TM) i7-10700 CPU @ 2.90GHz, 64GB of RAM, and 2x NVIDIA Geforce RTX 3090 GPU GPUs for 3 epochs. As objective we use the conventional multi-class cross-entropy loss function, where the number of classes is equal to the size of the vocabulary, defined as

$$\mathcal{L} = -\frac{1}{B}\sum_{b=1}^{B}\sum_{k=1}^{|V|} y_k^b \log(\hat{y}_k^b),$$

where the superscript $b$ represents the batch and $B$ is the batch size, $|V|$ is the vocabulary size, $y$ represents the true token to be predicted, and $\hat{y}$ is the output probability distribution over the vocabulary at each time-step.

To perform inference, we generate the output text using beam search, thus generating token-by-token the output sequence by feeding the input via cross-attention layers to the decoder, and auto-regressively generate the decoder output. We set the early stopping parameter to true so that the beam generation is finished when all beam hypotheses reached the `EOS` (End-of-Sequence) token. Experimentally we found that our fine-tuned model generates substance names for each beam, so there was no need to implement a constrained beam search to force the model to output only correct strings as output (i.e., only produce one of the 10 substance names or WW). Since we can augment the training set by sampling multiple time stamps for each acquisition, we aggregate the final predictions of the model using majority voting (i.e., the mode function) to have a single prediction for each $t_a$.

## VI. RESULTS

### A. EFFECTIVENESS

Table 5 shows the effectiveness metrics computed considering the average effectiveness score over each test set for the baselines (upper part of the table) and for the proposed approach (lower part of the table). Given that the proposed methodology can be provided in input with multiple $t_b$ timestamps (see Section V-D), the lower part of Table 5 shows the different effectiveness scores computed when sampling 1, 2, 5, 10, and 100 $t_b$ timestamps and aggregating the predictions of the model.

As we can see by inspecting the table as a whole, the proposed methodology outperforms the whole set of baselines for all the considered metrics, even in the most restrictive case where only one $t_b$ timestamp is provided (i.e., T5 (1-sample)). This behavior is also visually shown in Figure 6, which shows the value for the $\mathrm{F1_W}$ metric on the test sets: the different models are arranged along the x-axis, while the y-axis shows the metric value, and the horizontal dashed line represents the performance of the best baseline method. The bars on top of each value represent the variance of the metric over the different folds. The plots for the other metrics are similar and thus not reported.

Figure 7 shows the confusion matrix produced by the most effective methodology, i.e., T5 (100-samples), shown in the last row of Table 5. The confusion matrix reports the distribution of test sample predictions, displaying the real substances as rows and the predicted substances as columns.

TABLE 5: Average effectiveness metrics computed over the test sets. $\text{metric}_\text{m}$ denotes the micro-averaged metric, $\text{metric}_\text{M}$ the macro-averaged metric, and $\text{metric}_\text{W}$ the weighted metric.

| Model | Acc | MCC | $\text{Prec}_\text{m}$ | $\text{Prec}_\text{M}$ | $\text{Prec}_\text{W}$ | $\text{Rec}_\text{m}$ | $\text{Rec}_\text{M}$ | $\text{Rec}_\text{W}$ | $\text{F1}_\text{m}$ | $\text{F1}_\text{M}$ | $\text{F1}_\text{W}$ |
|---|---|---|---|---|---|---|---|---|---|---|---|
| AdaBoostM1_DecisionStump | 0.418 | 0.208 | 0.418 | 0.004 | 0.192 | 0.418 | 0.111 | 0.418 | 0.418 | 0.031 | 0.267 |
| AdaBoostM1_J48 | 0.802 | 0.767 | 0.802 | 0.720 | 0.744 | 0.802 | 0.732 | 0.802 | 0.802 | 0.718 | 0.765 |
| AdaBoostM1_RandomForest | 0.767 | 0.722 | 0.767 | 0.672 | 0.705 | 0.767 | 0.673 | 0.767 | 0.767 | 0.652 | 0.720 |
| Bagging | 0.764 | 0.718 | 0.764 | 0.687 | 0.720 | 0.764 | 0.672 | 0.764 | 0.764 | 0.649 | 0.721 |
| knn | 0.803 | 0.769 | 0.803 | 0.732 | 0.746 | 0.803 | 0.730 | 0.803 | 0.803 | 0.722 | 0.765 |
| MLP | 0.794 | 0.758 | 0.794 | 0.724 | 0.745 | 0.794 | 0.715 | 0.794 | 0.794 | 0.707 | 0.759 |
| RandomForest | 0.779 | 0.737 | 0.779 | 0.682 | 0.711 | 0.779 | 0.693 | 0.779 | 0.779 | 0.672 | 0.733 |
| SVM | 0.802 | 0.768 | 0.802 | 0.749 | 0.754 | 0.802 | 0.726 | 0.802 | 0.802 | 0.722 | 0.765 |
| Vote | 0.810 | 0.778 | 0.810 | 0.747 | 0.756 | 0.810 | 0.739 | 0.810 | 0.810 | 0.732 | 0.773 |
| T5 (1-sample) | 0.915 | 0.865 | 0.915 | 0.950 | 0.950 | 0.915 | 0.950 | 0.950 | 0.915 | 0.895 | 0.895 |
| T5 (2-samples) | 0.914 | 0.864 | 0.914 | 0.936 | 0.936 | 0.914 | 0.936 | 0.936 | 0.914 | 0.894 | 0.894 |
| T5 (5-samples) | 0.927 | 0.877 | 0.927 | 0.958 | 0.958 | 0.927 | 0.958 | 0.958 | 0.927 | 0.907 | 0.907 |
| T5 (10-samples) | 0.958 | 0.908 | 0.958 | 0.976 | 0.976 | 0.958 | 0.976 | 0.976 | 0.958 | 0.938 | 0.938 |
| T5 (50-samples) | 0.944 | 0.894 | 0.944 | 0.958 | 0.958 | 0.944 | 0.958 | 0.958 | 0.944 | 0.924 | 0.924 |
| T5 (100-samples) | 0.975 | 0.925 | 0.975 | 0.988 | 0.988 | 0.975 | 0.988 | 0.988 | 0.975 | 0.955 | 0.955 |

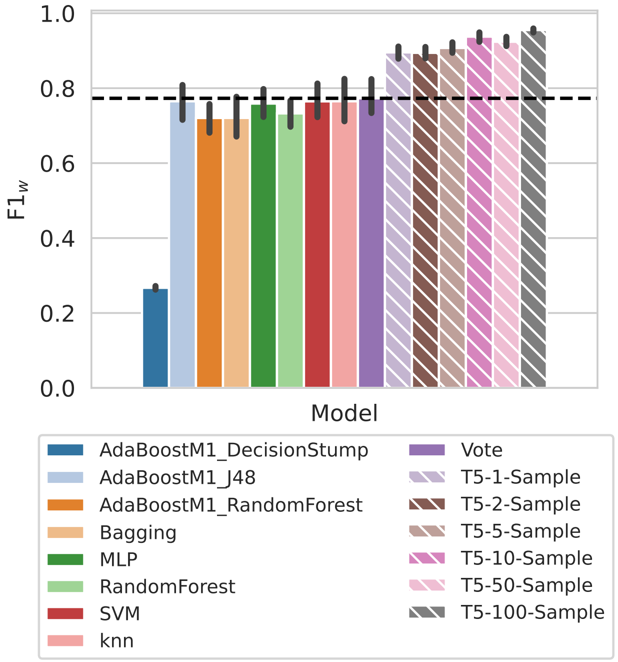
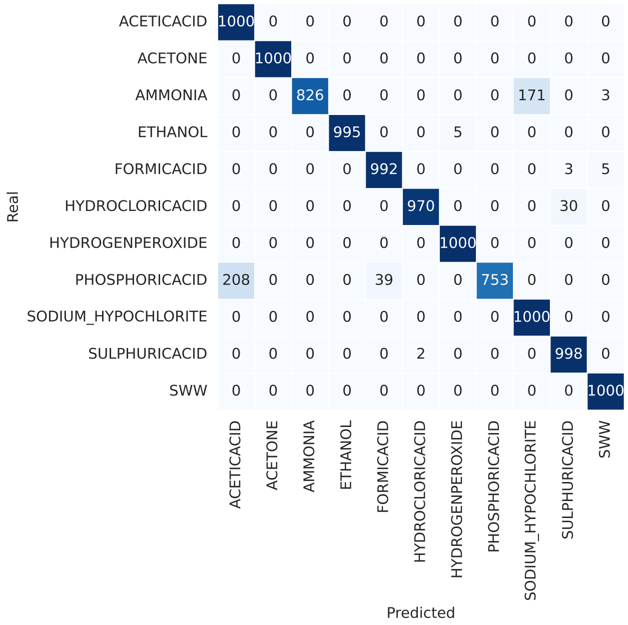


FIGURE 6: Average effectiveness metrics computed on the test sets for $\text{F1}_\text{W}$. The dotted line represents the best baseline method.



| | ACETICACID | ACETONE | AMMONIA | ETHANOL | FORMICACID | HYDROCLORICACID | HYDROGENPEROXIDE | PHOSPHORICACID | SODIUM_HYPOCHLORITE | SULPHURICACID | SWW |
|---|---|---|---|---|---|---|---|---|---|---|---|
| ACETICACID | 1000 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| ACETONE | 0 | 1000 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| AMMONIA | 0 | 0 | 826 | 0 | 0 | 0 | 0 | 0 | 171 | 0 | 3 |
| ETHANOL | 0 | 0 | 0 | 995 | 0 | 0 | 5 | 0 | 0 | 0 | 0 |
| FORMICACID | 0 | 0 | 0 | 0 | 992 | 0 | 0 | 0 | 0 | 3 | 5 |
| HYDROCLORICACID | 0 | 0 | 0 | 0 | 0 | 970 | 0 | 0 | 0 | 30 | 0 |
| HYDROGENPEROXIDE | 0 | 0 | 0 | 0 | 0 | 0 | 1000 | 0 | 0 | 0 | 0 |
| PHOSPHORICACID | 208 | 0 | 0 | 0 | 39 | 0 | 0 | 753 | 0 | 0 | 0 |
| SODIUM_HYPOCHLORITE | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1000 | 0 | 0 |
| SULPHURICACID | 0 | 0 | 0 | 0 | 0 | 2 | 0 | 0 | 0 | 998 | 0 |
| SWW | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1000 |

FIGURE 7: Distribution of the real and predicted values for the T5 (100-samples) model.

As we can see from the matrix, the model correctly classifies almost all substances perfectly (the values on the diagonal are close to 1000), with the exception of ammonia and phosphoric acid, where the model reaches an accuracy of about $0.8$. In particular, we see that ammonia is often mistaken for sodium hypochlorite (171 times out of 1000) and phosphoric acid is mistaken for acetic acid (208 times out of 1000). By investigating the dataset, we assume that this is probably caused by the fact that the signals acquired by the sensors show similar trends between the two pairs of substances. In particular, the starting conditions before the time of injection are extremely similar.

By focusing on the lower part of Table 5, we see that there is a correlation between the number of timestamps measured before the injection and the model effectiveness. More in detail, we can see that, overall, the more timestamps $t_b$ we provide to the model, the higher the effectiveness scores. This behavior is also shown in Figure 8, which displays in the x-axis the number of timestamps fed into the model, and in the y-axis the value for the different effectiveness metrics. This result suggests that practically, we should provide the model with multiple samples from the warm-up phase to get a more accurate prediction. This is due to the fact that seeing more timestamps during warm-up allows the model to

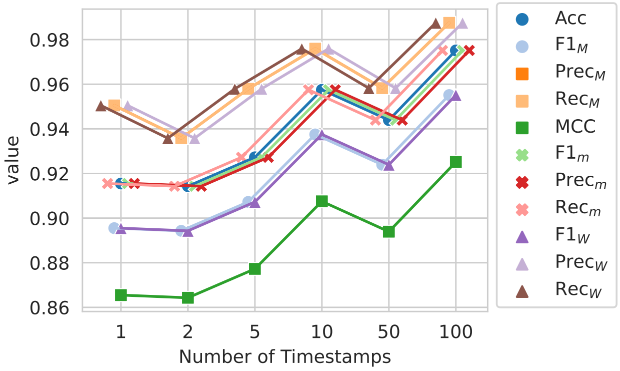


FIGURE 8: Average effectiveness metrics obtained when varying the number of timestamps $t_b$ fed to the T5 model.

better capture the signal variance that might happen before the substance is injected into the wastewater.

### B. EFFICIENCY

The inference time has been measured over repeated experiments on a NVIDIA Geforce RTX 3090 GPU. Disabling batching, it takes the system on average 76 milliseconds to perform one prediction (i.e., one full text-generation performed with beam search and the early stopping parameter enabled, see Section V for more details on the inference process) for a timestamp, or 76 seconds for 1000 timestamps. The system can therefore output approximately 13 predictions per second. We can further reduce the inference time by performing batched inference (the GPU used for the experiments allows for batch sizes greater than 512 samples).

We also measured the inference time on the available CPUs (16x Intel(R) Core(TM) i7-10700 CPU @ 2.90GHz). It takes the system on average 450 milliseconds to process one sample.

Such results provide evidence that we can use the trained model and deploy it for real-time predictions, both in an environment equipped with a GPU as well as in a machine that is only powered by CPUs.

In addition to the inference efficiency, which is crucial for deploying a model in real-world scenarios, it is important to consider the computational cost and complexity of the training phase in the proposed approach. T5 and other causal models have millions or even billions of parameters, which necessitate a large amount of data and computational power to optimize. These models are typically trained using the standard transformer architecture, which has a complexity of $\mathcal{O}(N^2d)$, where $N$ is the sequence length and $d$ is the hidden dimension of the model [59]. This means that the number of operations required to train the model grows quadratically with the input sequence length and linearly with the model hidden dimension.

Despite the high computational cost, pretrained models can be repurposed for various tasks through finetuning, which involves adapting a pretrained model to a specific task by training it on a small amount of task-specific data. This process is considerably less expensive than training the model from scratch because most pretrained models are already optimized for the underlying language modeling task. In practice, finetuning a pretrained model usually entails training it for just a few epochs, typically 3, which can take anywhere from a few minutes to a few hours, depending on the size of the task-specific dataset. For this work, we obtained the model's weights from the HuggingFace library and conducted only the finetuning phase of the proposed approach, which took approximately one hour on the GPU architecture described above. Once finetuned, the model can be used indefinitely for the specific task discussed in this paper.

## VII. DISCUSSION AND CONCLUSION

In this paper we studied the capabilities of natural language processing models, especially generative causal models and more in detail T5, for the task of detecting the presence of polluting substances in wastewater. To this end, differently from state-of-the-art machine learning models, we applied a transformation of the input features called textification in order to translate them into a textual form and be able to feed them into a generative natural language model. The latter is trained to classify each sample based on whether it contains or not a polluting substance, and to identify it if present. We experimentally evaluated the proposed methodology testing its effectiveness against a set of state-of-the-art baselines, and we measured its efficiency. Experimental results show that the proposed methodology outperforms the baseline methods, and its efficiency and effectiveness allow for its deployment and for practical use.

Given that the purposed approach is non-conventional, and it might seem strange or counter-intuitive at first sight, in the following we discuss why such approach makes sense and works in practice. Recent work demonstrated the vast ability of transformers and attention based models to generalize on a large variety of tasks, including those where the model has not been trained on [60, 61, 62, 63], or even to tasks not directly related to or not naturally expressed using natural language processing, such as for example images [26, 27], videos [29], reinforcement learning [39], and graphs [40].

The ability of transformer-based models for generalization comes from the attention mechanism, and from the almost task-agnostic training procedure. In fact it consists, in its base form, in reconstructing part of the input item, being it masked or perturbed using domain-specific techniques or to predict the continuation of the input (if the masked part is the last part of the input). Combined together, these techniques allow the model to learn meaningful and -most importantly- general latent relationships in input sequences, and the ability to relate those to the network's output. For example, networks applied to texts show the ability to reconstruct missing text or generate it from a prompt, for images and videos the ability to reconstruct corrupted or missing images and frames, for graphs to learn complex graph sub-structures (i.e., arrangements of set of nodes and edges), and so on. Besides those specific abilities, network based on transformers and trained with masking or causal objectives (i.e., predict masked parts or predict the continuation of the input) show high generalization abilities across tasks and domains. For the same reason, we believe that the textual description gathered from the sensors which we use to train our neural network allows for accurate forecasting predictions for the possible polluting substances present in wastewater.

Despite the promising results obtained with our approach, there are some limitations that need to be reported. One of the main limitations is that the proposed approach relies on the knowledge of the injection time of the polluting substances. This means that if the injection time is not known, the system may not be able to accurately classify the contaminants in wastewater. In this paper we solved this issue by relying on a finite state machine which is able to accurately identify injection time. Nevertheless, in future research we would focus on developing integrated methods to overcome this limitation and deploy an integrated single system.

Another limitation of the proposed approach is related to the availability of data, since a certain amount of labeled data is needed to train the deep learning model. Obtaining such data requires access to polluting substances or contaminated wastewater, and this can be difficult in practical situations. Future work will investigate alternative ways to generate synthetic data or explore transfer learning techniques to mitigate the data scarcity issue.

Results of this work open to a new research direction that will allow to tackle environmental tasks such as the analysis and detection of polluting substances by means of language models. Future work aims precisely at pursuing a broad adoption of natural language based models on a variety of domains and tasks related to the identification of substances. Furthermore, it will also focus on studying the generalization and explanation abilities of the model by leveraging zero and few-shot learning techniques as well as interpretability frameworks.

## REFERENCES

[1] L. T. Lee and E. R. Blatchley, "Long-term monitoring of water and air quality at an indoor pool facility during modifications of water treatment," *Water*, vol. 14, no. 3, 2022. [Online]. Available: https://www.mdpi.com/2073-4441/14/3/335

[2] H. Chojer, P. Branco, F. Martins, M. Alvim-Ferraz, and S. Sousa, "Development of low-cost indoor air quality monitoring devices: Recent advancements," *Science of The Total Environment*, vol. 727, p. 138385, 2020. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0048969720318982

[3] L. S. Hillary, S. K. Malham, J. E. McDonald, and D. L. Jones, "Wastewater and public health: the potential of wastewater surveillance for monitoring covid-19," *Current Opinion in Environmental Science & Health*, vol. 17, pp. 14–20, 2020.

[4] A. Trubetskaya, W. Horan, P. Conheady, K. Stockil, S. Merritt, and S. Moore, "A methodology for assessing and monitoring risk in the industrial wastewater sector," *Water Resources and Industry*, vol. 25, p. 100146, 2021.

[5] E. Syrmos, V. Sidiropoulos, D. Bechtsis, F. Stergiopoulos, E. Aivazidou, D. Vrakas, P. Vezinias, and I. Vlahavas, "An intelligent modular water monitoring iot system for real-time quantitative and qualitative measurements," *Sustainability*, vol. 15, no. 3, p. 2127, 2023.

[6] D. Jalal and T. Ezzedine, "Decision tree and support vector machine for anomaly detection in water distribution networks," in *2020 International Wireless Communications and Mobile Computing (IWCMC)*, 2020, pp. 1320–1323.

[7] D. G. Eliades and M. M. Polycarpou, "Water contamination impact evaluation and source-area isolation using decision trees," *Journal of Water Resources Planning and Management*, vol. 138, no. 5, pp. 562–570, 2012. [Online]. Available: https://ascelibrary.org/doi/abs/10.1061/%28ASCE%29WR.1943-5452.0000203

[8] A. Ria, M. Cicalini, G. Manfredini, A. Catania, M. Piotto, and P. Bruschi, "The sensiplus: A single-chip fully programmable sensor interface," in *Applications in Electronics Pervading Industry, Environment and Society*, S. Saponara and A. De Gloria, Eds. Cham: Springer International Publishing, 2022, pp. 256–261.

[9] M. Molinara, C. Bourelly, L. Ferrigno, L. Gerevini, M. Vitelli, A. Ria, F. Magliocca, L. Ruscitti, R. Simmarano, A. Trynda, and P. Olejnik, "A new dataset for detection of illegal or suspicious spilling in wastewater through low-cost real-time sensors," in *2022 IEEE International Conference on Smart Computing (SMARTCOMP)*, 2022, pp. 293–298.

[10] M. Ferdinandi, M. Molinara, G. Cerro, L. Ferrigno, C. Marrocco, A. Bria, P. Di Meo, C. Bourelly, and R. Simmarano, "A novel smart system for contaminants detection and recognition in water," in *2019 IEEE International Conference on Smart Computing (SMARTCOMP)*, 2019, pp. 186–191.

[11] C. Bourelly, A. Bria, L. Ferrigno, L. Gerevini, C. Marrocco, M. Molinara, G. Cerro, M. Cicalini, and A. Ria, "A preliminary solution for anomaly detection in water quality monitoring," in *2020 IEEE International Conference on Smart Computing (SMARTCOMP)*, 2020, pp. 410–415.

[12] A. Bria, G. Cerro, M. Ferdinandi, C. Marrocco, and M. Molinara, "An iot-ready solution for automated recognition of water contaminants," *Pattern Recognition Letters*, vol. 135, pp. 188–195, 2020.

[13] M. Molinara, M. Ferdinandi, G. Cerro, L. Ferrigno, and E. Massera, "An end to end indoor air monitoring system based on machine learning and sensiplus platform," *IEEE Access*, vol. 8, pp. 72 204–72 215, 2020.

[14] J. Lim, "Mobile sensor network to monitor wastew-

ater collection pipelines," https://escholarship.org/uc/item/0d9813bn, 2012, [Online: accessed 11-November-2022].

[15] M. Lepot, K. F. Makris, and F. H. Clemens, "Detection and quantification of lateral, illicit connections and infiltration in sewers with infra-red camera: Conclusions after a wide experimental plan," *Water Research*, vol. 122, pp. 678–691, 2017. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0043135417305067

[16] H. W. Ji, S. S. Yoo, B.-J. Lee, D. D. Koo, and J.-H. Kang, "Measurement of wastewater discharge in sewer pipes using image analysis," *Water*, vol. 12, no. 6, 2020. [Online]. Available: https://www.mdpi.com/2073-4441/12/6/1771

[17] I. Pisa, I. Santín, J. L. Vicario, A. Morell, and R. Vilanova, "Ann-based soft sensor to predict effluent violations in wastewater treatment plants," *Sensors*, vol. 19, no. 6, 2019. [Online]. Available: https://www.mdpi.com/1424-8220/19/6/1280

[18] A. Drenoyanis, R. Raad, I. Wady, and C. Krogh, "Implementation of an iot based radar sensor network for wastewater management," *Sensors*, vol. 19, no. 2, 2019. [Online]. Available: https://www.mdpi.com/1424-8220/19/2/254

[19] M. H. Popescu, K. Roitero, S. Travasci, and V. Della Mea, "Automatic assignment of ICD-10 codes to diagnostic texts using transformers based techniques," in *2021 IEEE 9th International Conference on Healthcare Informatics (ICHI)*. IEEE, 2021, pp. 188–192.

[20] K. Roitero, B. Portelli, M. H. Popescu, and V. Della Mea, "DiLBERT: Cheap embeddings for disease related medical NLP," *IEEE Access*, vol. 9, pp. 159 714–159 723, 2021.

[21] V. Della Mea, M. H. Popescu, and K. Roitero, "Underlying cause of death identification from death certificates using reverse coding to text and a nlp based deep learning approach," *Informatics in Medicine Unlocked*, vol. 21, p. 100456, 2020.

[22] M. H. Popescu, K. Roitero, and V. Della Mea, "Explainable classification of medical documents through a text-to-text transformer," in *HC@AIxIA 2022: 1st AIxIA Workshop on Artificial Intelligence For Healthcare*, 2022.

[23] H. Xue, F. D. Salim, Y. Ren, and C. L. Clarke, "Translating human mobility forecasting through natural language generation," in *Proceedings of the Fifteenth ACM International Conference on Web Search and Data Mining*, 2022, pp. 1224–1233.

[24] H. Xue, B. P. Voutharoj, and F. D. Salim, "Leveraging language foundation models for human mobility forecasting," *arXiv preprint arXiv:2209.05479*, 2022.

[25] T. Lin, Y. Wang, X. Liu, and X. Qiu, "A survey of transformers," *AI Open*, 2022.

[26] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly *et al.*, "An image is worth 16x16 words: Transformers for image recognition at scale," *arXiv preprint arXiv:2010.11929*, 2020.

[27] H. Wu, B. Xiao, N. Codella, M. Liu, X. Dai, L. Yuan, and L. Zhang, "Cvt: Introducing convolutions to vision transformers," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 22–31.

[28] R. Azad, M. T. Al-Antary, M. Heidari, and D. Merhof, "Transnorm: Transformer provides a strong spatial normalization mechanism for a deep segmentation model," *IEEE Access*, vol. 10, pp. 108 205–108 215, 2022.

[29] Y. Wang, Z. Xu, X. Wang, C. Shen, B. Cheng, H. Shen, and H. Xia, "End-to-end video instance segmentation with transformers," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 8741–8750.

[30] H. Yuan, Z. Cai, H. Zhou, Y. Wang, and X. Chen, "Transanomaly: Video anomaly detection using video vision transformer," *IEEE Access*, vol. 9, pp. 123 977–123 986, 2021.

[31] S. Khan, M. Naseer, M. Hayat, S. W. Zamir, F. S. Khan, and M. Shah, "Transformers in vision: A survey," *ACM computing surveys (CSUR)*, vol. 54, no. 10s, pp. 1–41, 2022.

[32] Z. Tian, J. Yi, Y. Bai, J. Tao, S. Zhang, and Z. Wen, "Synchronous transformers for end-to-end speech recognition," in *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2020, pp. 7884–7888.

[33] C. Subakan, M. Ravanelli, S. Cornell, M. Bronzi, and J. Zhong, "Attention is all you need in speech separation," in *ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2021, pp. 21–25.

[34] L. Yang, M. Qiu, C. Qu, C. Chen, J. Guo, Y. Zhang, W. B. Croft, and H. Chen, "Iart: Intent-aware response ranking with transformers in information-seeking conversation systems," in *Proceedings of The Web Conference 2020*, 2020, pp. 2592–2598.

[35] R. Ferreira, M. Leite, D. Semedo, and J. Magalhaes, "Open-domain conversational search assistants: the transformer is all you need," *Information Retrieval Journal*, vol. 25, no. 2, pp. 123–148, 2022.

[36] F. Sun, J. Liu, J. Wu, C. Pei, X. Lin, W. Ou, and P. Jiang, "Bert4rec: Sequential recommendation with bidirectional encoder representations from transformer," in *Proceedings of the 28th ACM international conference on information and knowledge management*, 2019, pp. 1441–1450.

[37] Y. Gu, Z. Ding, S. Wang, L. Zou, Y. Liu, and D. Yin, "Deep multifaceted transformers for multi-objective ranking in large-scale e-commerce recommender systems," in *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*,

2020, pp. 2493–2500.

[38] E. Parisotto, F. Song, J. Rae, R. Pascanu, C. Gulcehre, S. Jayakumar, M. Jaderberg, R. L. Kaufman, A. Clark, S. Noury *et al.*, "Stabilizing transformers for reinforcement learning," in *International conference on machine learning*. PMLR, 2020, pp. 7487–7498.

[39] L. Chen, K. Lu, A. Rajeswaran, K. Lee, A. Grover, M. Laskin, P. Abbeel, A. Srinivas, and I. Mordatch, "Decision transformer: Reinforcement learning via sequence modeling," *Advances in neural information processing systems*, vol. 34, pp. 15 084–15 097, 2021.

[40] V. P. Dwivedi and X. Bresson, "A generalization of transformer networks to graphs," *arXiv preprint arXiv:2012.09699*, 2020.

[41] J. Kim, D. Nguyen, S. Min, S. Cho, M. Lee, H. Lee, and S. Hong, "Pure transformers are powerful graph learners," *Advances in Neural Information Processing Systems*, vol. 35, pp. 14 582–14 595, 2022.

[42] J. Jumper, R. Evans, A. Pritzel, T. Green, M. Figurnov, O. Ronneberger, K. Tunyasuvunakool, R. Bates, A. Žídek, A. Potapenko *et al.*, "Highly accurate protein structure prediction with alphafold," *Nature*, vol. 596, no. 7873, pp. 583–589, 2021.

[43] R. Zaman, M. H. Newton, F. Mataeimoghadam, and A. Sattar, "Constraint guided neighbor generation for protein structure prediction," *IEEE Access*, vol. 10, pp. 54 991–55 001, 2022.

[44] J. Wang, X. Lin, and H. Yu, "Poat-net: Parallel offset-attention assisted transformer for 3d object detection for autonomous driving," *IEEE Access*, vol. 9, pp. 151 110–151 117, 2021.

[45] X. Chen, H. Zhang, F. Zhao, Y. Cai, H. Wang, and Q. Ye, "Vehicle trajectory prediction based on intention-aware non-autoregressive transformer with multi-attention learning for internet of vehicles," *IEEE Transactions on Instrumentation and Measurement*, vol. 71, pp. 1–12, 2022.

[46] D. Martín-Gutiérrez, G. Hernández-Peñaloza, A. B. Hernández, A. Lozano-Diez, and F. Álvarez, "A deep learning approach for robust detection of bots in twitter using transformers," *IEEE Access*, vol. 9, pp. 54 591–54 601, 2021.

[47] Y. E. Seyyar, A. G. Yavuz, and H. M. Ünver, "An attack detection framework based on bert and deep learning," *IEEE Access*, vol. 10, pp. 68 633–68 644, 2022.

[48] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention Is All You Need," in *Proceedings of NIPS*, 2017.

[49] C. Raffel, N. Shazeer, A. Roberts, K. Lee, S. Narang, M. Matena, Y. Zhou, W. Li, and P. J. Liu, "Exploring the limits of transfer learning with a unified text-to-text transformer," *Journal of Machine Learning Research*, vol. 21, pp. 1–67, 2020.

[50] Y. Freund and R. E. Schapire, "Experiments with a new boosting algorithm," in *ICML*, 1996.

[51] J. R. Quinlan, *C4. 5: programs for machine learning*. Elsevier, 2014.

[52] L. Breiman, "Random forests," *Machine Learning*, vol. 45, no. 1, pp. 5–32, Oct 2001.

[53] ——, "Bagging predictors," *Machine Learning*, vol. 24, no. 2, pp. 123–140, Aug 1996.

[54] E. Frank and B. Pfahringer, "Improving on bagging with input smearing," in *Pacific-Asia Conference on Knowledge Discovery and Data Mining*. Springer, 2006, pp. 97–106.

[55] T. Cover and P. Hart, "Nearest neighbor pattern classification," *IEEE Transactions on Information Theory*, vol. 13, no. 1, pp. 21–27, 1967.

[56] C. Cortes and V. Vapnik, "Support-vector networks," *Machine Learning*, vol. 20, no. 3, pp. 273–297, Sep 1995.

[57] T. Hastie, R. Tibshirani, and J. Friedman, *The elements of statistical learning: data mining, inference and prediction*, 2nd ed. Springer, 2009. [Online]. Available: http://www-stat.stanford.edu/~tibs/ElemStatLearn/

[58] I. H. Witten, E. Frank, and M. A. Hall, "Data mining: Practical machine learning tools and techniques (third edition)," in *Data Mining: Practical Machine Learning Tools and Techniques (Third Edition)*, 3rd ed., ser. The Morgan Kaufmann Series in Data Management Systems. Boston: Morgan Kaufmann, 2011, p. 1–621.

[59] M. Phuong and M. Hutter, "Formal algorithms for transformers," *arXiv preprint arXiv:2207.09238*, 2022.

[60] R. Csordás, K. Irie, and J. Schmidhuber, "The devil is in the detail: Simple tricks improve systematic generalization of transformers," *arXiv preprint arXiv:2108.12284*, 2021.

[61] M. Xu, Y. Shen, S. Zhang, Y. Lu, D. Zhao, J. Tenenbaum, and C. Gan, "Prompting decision transformer for few-shot policy generalization," in *Proceedings of the 39th International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, K. Chaudhuri, S. Jegelka, L. Song, C. Szepesvari, G. Niu, and S. Sabato, Eds., vol. 162. PMLR, 17–23 Jul 2022, pp. 24 631–24 645. [Online]. Available: https://proceedings.mlr.press/v162/xu22g.html

[62] Y. Li and J. L. McClelland, "Systematic generalization and emergent structures in transformers trained on structured tasks," *arXiv preprint arXiv:2210.00400*, 2022.

[63] H. W. Chung, L. Hou, S. Longpre, B. Zoph, Y. Tay, W. Fedus, E. Li, X. Wang, M. Dehghani, S. Brahma *et al.*, "Scaling instruction-finetuned language models," *arXiv preprint arXiv:2210.11416*, 2022.

**KEVIN ROITERO** is an Assistant Professor (RT-D/a) at the University of Udine, Italy. His research interests include Information Retrieval, Crowdsourcing, Data mining and analysis, and Machine Learning and Artificial Intelligence. He visited and collaborated with multiple top universities across the globe as well as with leading industry partners, publishing papers in top ranked conferences and in top-tier journals. As result of the his work, he received multiple grants and awards, including the participation in the 7th edition of the prestigious Heidelberg Laureate Forum (top 200 young researchers in mathematics and computer science), the "con.Scienze2020" prize for the best Ph.D. thesis in Computer Science discussed in 2020 in Italy, as well as the Best Short Paper Award at "ECIR2020", and the Best Paper Award at "NL4AI 2022".

**BEATRICE PORTELLI** is a PhD Student in the National PhD program in Artificial Intelligence PhD-AI.it, at University of Udine, Italy. She is and member of the AILAB-Udine. She works employing Deep Learning and Natural Language Processing techniques. She has worked on several projects related to Language Models for Adverse Drug Event extraction and normalization from social media texts, as well as Fact Verification Models. Her current research interests are Machine Learning and Deep Learning methods for risk management in the agricultural-forestry sector.

**GIUSEPPE SERRA** is an Associate Professor at University of Udine and he is leading the Artificial Intelligence Laboratory. He received his Ph.D. in Computer Engineering, Multimedia and Telecommunications in 2010 at the University of Florence, Italy. From 2014 to 2016 he was an Assistant Professor at the University of Modena and Reggio Emilia, Italy. He was a visiting researcher at Carnegie Mellon University, Pittsburgh, USA, and at Telecom ParisTech/ENST, Paris, in 2006 and 2010 respectively. His research interests include Machine Learning and Deep Learning. He was the lead organizer of the "International Workshop on Egocentric Perception, Interaction and Computing (EPIC)" in 2016 and 2017 (ECCV'16 – ICCV'17) and he gave tutorials at two international conferences (ICPR'12, CAIP'13). He also serves as an Editor Board of IEEE THMS and ACM TOMM. He was a Technical Program Committee member of several conferences and workshops. He regularly serves as reviewer for international conferences and journals such as AAAI, ICML, NIPS, ACL, ECCV, CVPR, IEEE TPAMI, IEEE TMM. He has published more than 70 publications in the most prestigious journals and conferences in the field.

**VINCENZO DELLA MEA** is associate professor of Medical Informatics and of Cloud Technologies at the University of Udine, Italy. He is also head of the Medical Informatics, Telemedicine and eHealth Lab (MITEL). V.Della Mea has been national delegate for the COST Action "EUROTELEPATH" (2008- 2011), local responsible for the EU MSCA project "AIDPATH" (2013-2017) and for the EU MSCA Doctoral Network "Bosomshield" (2022-2026); he also participated in other national projects. V. Della Mea served as WHO Informatics and Terminologies Committee co-chair in the WHO Network of the Family of International Classifications. In the same network he was also member of the Joint Task Force on ICD-11 until the end of mission, and now is member of the ICHI Task Force (International Classification of Health Interventions). He is former vice-President and currently member of the Executive Board of the Italian Scientific Society for Biomedical Informatics (SIBIM). He is Associate Editor of Digital Health (SAGE).

**STEFANO MIZZARO** is Full Professor of Information processing systems at the University of Udine, Italy, where he teaches courses on Web Information Retrieval, Social Computing, Artificial Intelligence, and Object Oriented Analysis and Design. His main research interests are information retrieval evaluation, crowdsourcing, misinformation detection, mobile devices and systems, and scholarly publishing and peer review. According to Google Scholar (https://scholar.google.com/citations?user=2wvJC6IAAAAJ&hl=en) he has more than 150 publications, about 4500 citations, and an h-index of 30.

**GIANNI CERRO** (Member, IEEE) is currently a Research Fellow with the Department of Medicine and Health Sciences "Vincenzo Tiberio", University of Molise, Italy. His research interests involve magnetic localization systems for biomedical and industrial applications, cognitive radio systems for new generation communication technologies, measurements in telecommunication networks, sensor networks for environmental monitoring, and measurement characterization of medical devices, such as brain–computer interfaces.

**MICHELE VITELLI** received a Master's degree in 2020 in Computer Engineering at the University of Cassino and Southern Latium, Cassino, Italy. Since 2021 works as a Software Engineer in the Research and Development Area of Sensichips s.r.l company. From 2021 has been a Ph.D. Student at the University of Cassino and Southern Latium, Cassino, Italy. His main research interest concerns the application of artificial intelligence algorithms on devices with limited resources exploiting the IoT paradigm, focusing on automotive applications for the estimation of the State of Health and State of Charge of the battery cells, and the identification of pollutants in water and air.

**MARIO MOLINARA** Received an MSc degree in Computer Science from the University of Sannio in 1999 and a Ph.D. Degree in "Computer Science and Telecommunication" from the University of Salerno in 2003. In 2004 he joined the Department of Electrical and Information Engineering (DIEI), where is now an Assistant Professor in Computer Science and Artificial Intelligence at the University of Cassino and Southern Lazio. He has authored over a hundred International Journals and Conference Proceedings research papers. He has been guest editor of special issues on "Pattern Recognition for Cultural Heritage" and on "Smart Distributed Sensors" hosted in Pattern Recognition Letters. He is a member of the editorial board of the "Journal of Ambient Intelligence and Humanized Computing" (Springer Verlag), and a member of the Topical Advisory Panel of the Journal of Imaging (MDPI). He is a member of the International Association of Pattern Recognition (IAPR) and of the IEEE. His current research interests include image analysis and interpretation, classification techniques, biomedical imaging, neural networks, optical character recognition, map and document processing, intelligent measurement systems for fault detection and diagnosis, smart sensors, IoT, Artificial Intelligence on the Edge, and pattern recognition applied to Cultural Heritage.

• • •