NATIONAL TECHNICAL UNIVERSITY OF ATHENS
JOINT POSTGRADUATE PROGRAMME
COMPUTATIONAL MECHANICS

N.C.S.R. “DEMOKRITOS”
INSTITUTE OF INFORMATICS AND TELECOMMUNICATIONS
INSTITUTE OF NANOSCIENCE AND NANOTECHNOLOGY

Smart Gas Sensors

Deep learning for the identification and classification of various gaseous species by sensors

THESIS

by

ANDRIKOS KONSTANTINOS

Supervisors: Lagaros Nikolaos, Professor, N.T.U.A.
Davazoglou Dimitris, Director of Research, N.C.S.R. “Demokritos”
Klampanos Iraklis, Research Associate, N.C.S.R. “Demokritos”

Athens, June 2021

National Technical University of Athens
Joint Postgraduate Programme
Computational Mechanics

N.C.S.R. “Demokritos”
Institute of Informatics and Telecommunications
Institute of Nanoscience and Nanotechnology

Smart Gas Sensors
Deep learning for the identification and classification of various gaseous species by sensors

THESIS

by

ANDRIKOS KONSTANTINOS

Supervisors: Lagaros Nikolaos, Professor, N.T.U.A.
Davazoglou Dimitris, Director of Research, I.N.N.–N.C.S.R., “Demokritos”
Klampanos Iraklis, Research Associate, I.I.T.–N.C.S.R., “Demokritos”

Approved by the examination committee:

(Signature)	*(Signature)*	*(Signature)*
....	....	....
Lagaros Nikolaos	Theodorou Doros	Riziotis Vasilis
Professor, N.T.U.A.	Professor, N.T.U.A.	Professor, N.T.U.A.

Athens, June 2021

Acknowledgments

This present thesis was jointly conducted at the Artificial Intelligence Laboratory of the Institute of Informatics & Telecommunications (I.I.T.), and at the Institute of Nanoscience & Nanotechnology (I.N.N.) of the National Centre for Scientific Research (N.C.S.R.) «Demokritos».

I wish to express my sincere appreciation to my supervisors at the N.C.S.R. «Demokritos», Dr. Klampanos Iraklis, Research Associate at the Institute of Informatics & Telecommunications and Dr. Davazoglou Dimitris, Director of Research at the Institute of Nanoscience & Nanotechnology. Thank you for giving me the opportunity to work on this very interesting project, and for being so generous with your time and intellect. This thesis would not have been possible without your guidance, support and motivation.

I would like to express my gratitude to Professor Lagaros Nikolaos, for his acceptance to supervise this thesis, for the clear communication we had during the development of this project, as well as for all the valuable lessons I was lucky enough to receive during this postgraduate program, which aroused my interest in the fields of optimization and artificial intelligence.

I would like to extend my sincere thanks to Dr. Paliouras Georgios, Director of Research at the Institute of Informatics & Telecommunications and head of the Artificial Intelligence Laboratory, for showing his interest in this project and for providing me with valuable remarks which had a significant contribution to the development of this work. I would also like to thank Mr. Davvetas Athanasios, Ph. D. candidate at the Institute of Informatics & Telecommunications, for his assistance in the connection to the laboratory's high performance computer, where I was able to run all computational experiments.

Finally, I could not close this section without expressing my gratitude to my family for their constant support and unconditional love throughout the years of my studies and my life in general.

This page intentionally left blank

Abstract

The aim of this present thesis was the exploitation of Artificial Intelligence (AI) algorithms for the discrimination of various gaseous species based on their type and concentration. For the detection of the volatile organic compounds we used metal oxide semiconductor (MOX) gas sensors. A key feature of these sensors is the alteration of one or more of their physical properties upon exposure to a gas stimuli in a way that is possible to measure and quantify.

In our experiments during the exposure of an array of sensors to a deoxidizing gas, changes in the resistance of each sensor were measured. The acquired data were multivariate time series since we measured the response of an array of three sensors. The gas sensor array (GSA) delivers a unique fingerprint upon exposure to a gas stimuli. The next stage consists of pre-processing the acquired time series in order to use a pattern recognition algorithm for the recognition of the acquired fingerprint. In this context we used machine learning (ML) and deep learning (DL) models which after sufficient training, they are capable of predicting the class of a future recording of the GSA.

The experimental process includes an odour delivery system, consisting of a sealed chamber where the GSA was placed, a gas injection phase, and the measure of the responses of the GSA by the Keithley 2400 instrument. We constructed the GSA's electrical circuit and also a circuit for the control of the acquisition by an Arduino microcontroller.

This work demonstrates the utilization of ML and DL algorithms in the field of smart gas sensors. For this purpose we used the dataset created in the laboratory but also some relevant datasets freely available from the UCI Machine Learning Repository.

Keywords: smart gas sensing; MOX gas sensors; pattern recognition; machine learning; neural networks;

Περίληψη

Ο στόχος της παρούσας εργασίας ήταν η ανάπτυξη μιας μεθοδολογίας η οποία θα αξιοποιεί αλγορίθμους τεχνητής νοημοσύνης για την ταξινόμηση διάφορων αερίων με βάση το είδος τους και την συγκέντρωση στην οποία συναντώνται όταν ανιχνεύονται από αισθητήρες αερίων μεταλλικού οξειδίου. Η ανίχνευση των αερίων – στόχων γινόταν από αισθητήρες μεταλλικού οξειδίου (MOX gas sensors), βασικό χαρακτηριστικό των οποίων είναι η μεταβολή ορισμένων φυσικών τους ιδιοτήτων όταν βρίσκονται στο περιβάλλον αερίων που θα τους διεγείρουν.

Στα πειράματα μας, κατά την έκθεση μιας συστοιχίας αισθητήρων αερίων (GSA) σε κάποιο αέριο καταγράφαμε την μεταβολή της ειδικής αντίστασης των ανιχνευτών λαμβάνοντας έτσι δεδομένα χρονοσειρών. Ακολούθησε ένα στάδιο προ-επεξεργασίας αυτών των δεδομένων, προκείμενου αυτά να χρησιμοποιηθούν για την ανάλυση του κάθε αποτυπώματος που είχε η συστοιχία κατά την έκθεσή της σε κάποιο διαφορετικό αέριο-στόχο. Σε αυτό το πλαίσιο έγινε χρήση αλγορίθμων μηχανικής και βαθειάς μάθησης, οι οποίοι αφού εκπαιδευτούν στα μετρούμενα δεδομένα, θα είναι ικανοί να προβλέψουν την κατηγορία στην οποία θα ανήκει μια μελλοντική καταγραφή της απόκρισης της συστοιχίας.

Η πειραματική διαδικασία περιελάμβανε ένα σύστημα εκβολής των αερίων σε έναν κλειστό θάλαμο εντός του οποίου βρίσκεται η συστοιχία και την καταγραφή της απόκρισης όλων των αισθητήρων από το καταγραφικό μηχάνημα Keithley 2400. Κατασκευάστηκαν επίσης το ηλεκτρικό κύκλωμα της συστοιχίας και ένα κύκλωμα ελεγχόμενο από microcontroller (Arduino), που επέτρεπε την καταγραφή της αντίστασης όλων των αισθητήρων της συστοιχίας, από το καταγραφικό μηχάνημα.

Η εργασία αναδεικνύει την αξιοποίηση αλγορίθμων μηχανικής και βαθειάς μάθησης στο πεδίο των έξυπνων ανιχνευτών αερίων χρησιμοποιώντας τα πειραματικά δεδομένα που συλλέχθηκαν στο εργαστήριο αλλά και σύνολα δεδομένων που υπάρχουν διαθέσιμα στο διαδίκτυο.

Keywords: smart gas sensing; MOX gas sensors; pattern recognition; machine learning; neural networks;

This page intentionally left blank

Contents

1 *Introduction*

1.1 Smart Gas Sensing

In recent years with the development of Internet-of-Things (IoT) technology, gas sensors are increasingly becoming an important part of our everyday lives. They can be found in our homes (e.g., monitoring the level of CO in air from gas-fired boilers), in our workplace (e.g., checking the levels of toxic gases and odours in offices), and in hospitals (e.g., monitoring anaesthetic and respiratory gases during operations). In such complex sensing scenarios, the gas sensor shows the defects of cross sensitivity and low selectivity. Therefore, smart gas sensing methods have been proposed to address these issues by adding sensor arrays, signal processing, and machine learning techniques to traditional gas sensing technologies.

1.2 Problem Statement

As the size of semiconductors shrinks into the nanometer scale regime, many uncertainties may be introduced during the manufacturing process of the sensing material (metal oxide semiconductor) as new physical phenomena at short dimensions occur, and limitations in material properties are reached. Thus, it is impossible to know a-priori the exact response of a gas sensor device upon exposure to a volatile organic compound at a specified concentration. For that reason, essential part of the manufacturing process of a sensor is the calibration stage in which each sensor is being exposed to a variety of gases and concentrations in order to construct its sensitivity basis. The sensor is then ready to be used as a detector for specific target-gases in a limited range of concentrations. Nevertheless, the sensor will be affected by non-target gases which have similar chemical characteristics, even if the sensor's target gas is not present. This is known as cross sensitivity of the sensor, and together with the problem of low selectivity which is the variations in sensor responses depending on the environmental temperature and humidity, are the main reasons why gas sensors can not be used for the characterization of gases, and their applications are limited to alarm activation.

To cope with the problem of low selectivity and cross sensitivity, we use multiple sensors of different sensitivity characteristics. In this project we combine sensors that show different response to the same gas-concentration combination, in order to construct an array of sensors which will provide a unique fingerprint upon exposure to an odor. The target-gas of each individual sensor could either be different or the same (as long as they provide different response).

By using data analysis tools, we can extract information about the nature of the gas stimuli by analyzing the fingerprint provided by gas the sensor array.

1.3 Thesis overview

This thesis consists of the following chapters:

- In Chapter 2 we mention some related works.
- In Chapter 3 we refer to the theoretical background of gas sensors and also to the data preprocessing techniques and pattern recognition methods used in smart gas sensing technology.
- In Chapter 4 we go through the experimental process, we refer to the characteristics of the datasets that we used and we mention the computational frameworks we used for the analysis.
- In Chapter 5 we present the experimental results and the performance measures of the models we used for the analysis.
- Finally, Chapter 6 includes a discussion section on the derived results and some thoughts on future work.

2 *Related work*

Many studies have been held regarding the use of pattern recognition (PARC) methods for the classification of gas sensor array data. In [1] by *Hines et al.*, the authors provide guidelines about the preparation of GSA raw data through a preprocessing stage, as well as the most commonly used PARC methods applied to processed data, in order to extract insights about the nature of the gas stimuli. One of the first articles on the use of PARC methods to GSA data is presented by *Gardner et al. [2]*[3] where the use of principal component analysis (PCA), clustering methods and artificial neural networks for the classification of GSA data is presented.

In later studies, *Pardo and Sberveglieri* [4] investigates the use of Support Vector Machines (SVM), where the authors express the error of the SVM as a function of the number of principal components, the kernel parameter value for both the polynomial and the RBF kernel, and the regularization parameter *C*. In the same context, the use of Gradient Tree Boosting algorithm is investigated by *Luo et al.*, [5]. This approach is proposed for fast recognition as the sensors in the GSA do not reach their steady-state, hence, only the transient signals are being analyzed.

In another study held by *Krivetskiy et al.* [6], the authors make use of PARC methods such as random forest, support vector machine and shallow multi-layer perceptron algorithms, to selectively detect the presence of individual gases at low concentrations by a single SnO_2 gas sensor. It is reported that artificial neural networks are more effective compared to the other PARC methods, as they exhibit an error of 13.2%. The same study also investigates the ability of a single sensor to detect the presence of a gas mixture. For that case the results indicate an error less than 10%.

Regarding the problem of fault detection by sensors, *Yang et al.* [7] make use of clustering-k-Nearest Neighbors (kNN) algorithm to address this problem. The results indicate that the proposed method solves the problem of fault detection faster than the traditional kNN. Hence, the use of clustering-kNN is more suitable for handling bigger datasets as well as for real-time process monitoring.

One of the main problems in the operation of gas sensors is the effect of drift which is the gradual variation of the chemo-sensory signal responses when exposed to the same analyte under identical conditions, caused by the reorganization of the sensor's surface over long periods of time. *Belhouari et al.* [8] demonstrate a system which provides fast recognition of volatile organic compounds, accompanied by a Gaussian Mixture model for the counteraction of sensor drift. In the same context, *Vergara et al.* [9] shows the effect of drift in sensors in a dataset collected over a period of three years. They address the problem of drift by using an ensemble of SVMs, each one trained at different points of time.

There are also many reports on the literature regarding the use of deep convolutional neural networks (DCNN) for gas classification. *Zhao et al.*[10] presented the use of a one-dimensional DCNN for automatically extracting features and classifying mixture gases. This network exhibits higher recognition accuracy (96.3%) compared to conventional pattern recognition algorithms such as SVM, ANN, k-nearest neighbor and random forest. *Peng et al.* [11] proposed a novel 1D-DCNN consisting of 38 layers, named GasNet, tailored for gas classification. It is shown that GasNet can provide higher classification accuracy than comparable SVM methods and Multiple Layer Perceptron (MLP).

Two-dimensional DCNN, which are being widely used for computer vision applications, have also been used in gas classification. In order to create input data for a 2D-DCNN, one should transform the multivariate time series data of a GSA into a 2-D coloured image. Three different transformation methods for encoding time series as images (Gramian Angular Summation Field, Gramian Angular Difference Field, Markov Transition Field) are presented by *Yang et al.* in [12]. In another study held by *Wei et al.* [13] a gas identification CNN based on the LeNet-5 architecture proposed by Yann LeCun [14] is used for the discrimination of CO, CH_4 and their mixtures in various concentrations. The time series data are encoded as greyscale images and the final accuracy reached is 98.67%. Another approach regarding the encoding of time series as images is presented by *Han et al.* in [15]. In this study the images which are lately fed into a 2D-DCNN are the plots of the time series acquired from the GSA. Four different mapping methods of the time series and five pre-trained 2D-DCNN were used. The final accuracy reached is 96.67%.

3 *Machine Olfaction*

Machine olfaction can be defined as the instrumental replication of the human olfactory sense. A system capable of simulating the sense of smell consists of several different gas sensors which produce electrical signals depending on the nature of the surrounding atmosphere. Next, the acquired signals go through a processing stage in order to prepare the raw data for multivariate pattern analysis by utilizing a pattern-recognition method. By the end of this stage, we can derive meaningful insights about the volatile compounds that compose the environment of the sensors [16].

This chapter aims at introducing the reader to the fundamental elements of which an intelligent olfactory system consists.

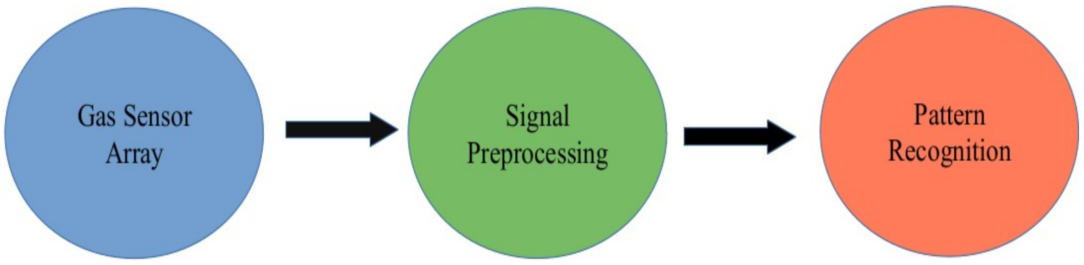


Figure 1: Steps of Smart Gas Sensing

3.1 Chemical Gas Sensors

A chemical gas sensor is a device which upon exposure to volatile compounds alters one or more of its physical properties, such as mass, electrical conductivity or dielectric properties, in a way that is possible to measure and quantify. These changes deliver an electrical signal, with a magnitude that is proportional to the concentration of the gas under test [17]. Most commonly used chemical gas sensors are the metal oxide (MOX) semiconductors, organic crystals and conductive polymers [18]. In this present thesis we used MOX gas sensors.

3.1.1 MOX Sensing Mechanism

In 1953 Brattain and Bardeen discovered that the adsorption of a volatile compound on the surface of a semiconductor can cause a great change of its resistance. This phenomenon has been observed since then in many metal oxides including SnO_2, ZnO, TiO_2 and In_2O_3 with the SnO_2 being the most commonly used material for gas sensing applications. Tin oxide under certain circumstances behaves as an n-type semiconductor at oxygen-containing environment (e.g. air). The basic reactions that occur within the porous sintered film can be represented by the following reactions:

$$\frac{m}{2}O_2 + \{vacancy\} + e^- \xrightarrow{k_1} \{O_m^-\} \tag{3.1}$$

$$X + \{O_m^-\} \xrightarrow{k_2} \{XO_m\} + e^- \tag{3.2}$$

First, vacant sites within the non-stoichiometric tin oxide lattice react with atmospheric oxygen to abstract electrons out of the conduction band of the tin oxide creating chemisorbed oxygen sites such as O^-, O_2^- , and so on (Eq. 3.1). Next, this reversible reaction is disturbed when the analyte molecule X reacts with the chemisorbed oxygen species to release electrons and promulgate further reactions (Eq. 3.2) [19]. Consequently, the conductivity is increased as a result of the increase in carrier concentration.

The schematic diagram in Figure 2 explains the conductivity increment due to the carrier mobility for SnO_2 gas sensors. In clean air, oxygen atoms that trap free electrons in the bulk of SnO_2, are adsorbed onto the SnO_2 particle surface, forming a potential barrier in the grain boundaries as shown in Fig. 2a. This potential barrier restricts the flow of electrons, causing the electrical conductivity to decrease, because the potential barrier acts as the scattering centre for electron conduction. When the sensor is exposed to an atmosphere containing reducible gases, e.g. combustible gases, CO, and other similar vapours, the SnO_2 surface adsorbs these gas molecules and causes oxidation. This lowers the potential barrier, allowing electrons to flow more easily, thereby increasing the electrical conductivity as shown in Fig. 2b.

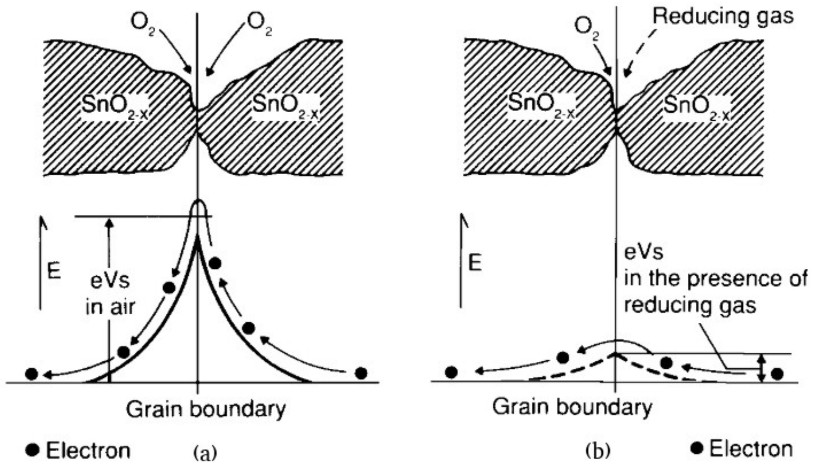


Figure 2: Schematic diagram explaining the conductivity increment caused by the carrier mobility increase in SnO2 gas sensors. (General Information for TGS Sensors, Figaro USA Inc.)

3.1.2 MOX Gas Sensing Devices

A schematic of a typical conductivity sensor design is shown in Figure 3. The sensing material is deposited over interdigitated or two parallel electrodes, which form the electrical connections through which the relative resistance change is measured. The heater is required because very high temperatures are required for effective operation of metal oxide sensors for several reasons [20]. First, and most important, the chemical reaction is more specific at higher temperatures, and second, the reaction kinetics are much faster, that is, the device responds in just a few seconds. Finally, operating the device well above a temperature of 100 °C ameliorates the effect of humidity upon its response – a critical factor for many chemical sensors [19].

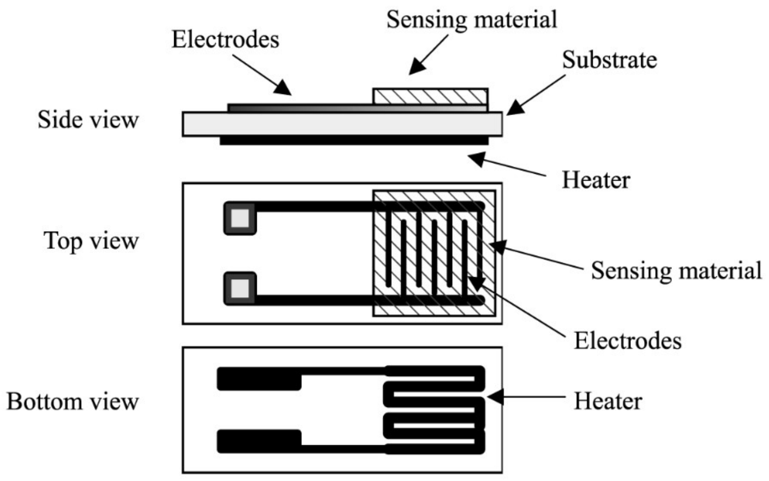


Figure 3: Typical structure of a conductivity sensor

Figure 4 illustrates a typical behavior of the sensor's resistance when the sensor is exposed to and then removed from a deoxidizing gas.

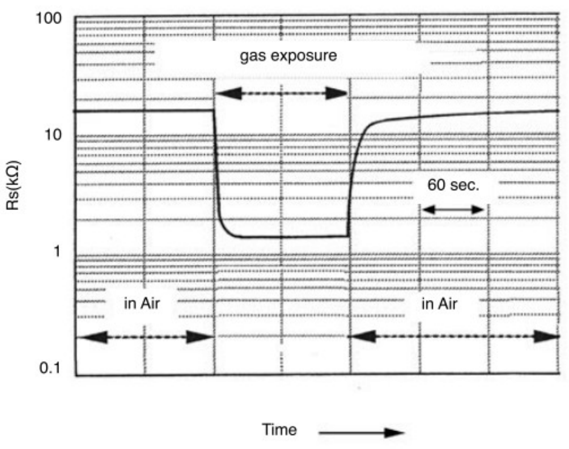


Figure 4: Typical sensor response when exposed to a deoxidizing gas (General Information for TGS Sensors, Figaro USA Inc.)

The relationship between sensor resistance and the concentration of deoxidizing gas can be expressed by the following equation over a certain range of gas concentration:

$$R_s = A\,[C]^{-\alpha} \tag{3.3}$$

where R_S is the electrical resistance of the sensor, A is a constant depending on the sensor's operating temperature and on the type of target gas, C is the gas concentration and α is the slope of R_S curve. As can be seen from the above equation, the relationship of sensor resistance to gas concentration is linear on a logarithmic scale within a practical range of gas concentration (from several ppm to several thousand ppm). Figure 5 shows a typical example of the relationship between sensor resistance and gas concentration. R_0 is the reading of the sensor's resistance upon exposure to a specific concentration of a target-gas of interest.

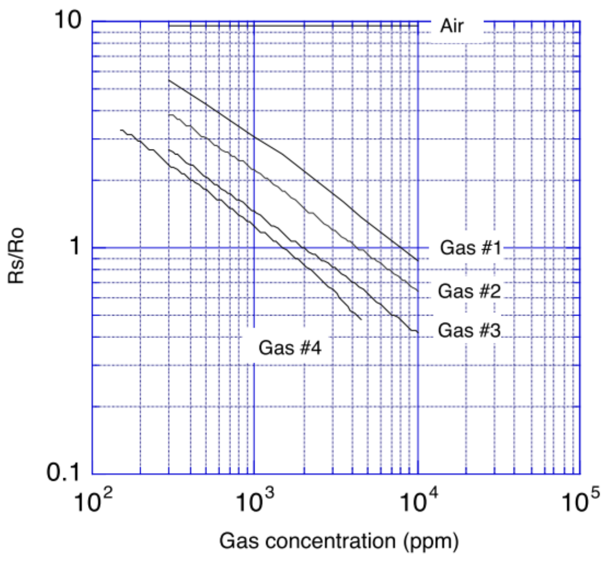


Figure 5: Typical sensitivity characteristics (Figaro, General Information for TGS Sensors, Figaro Engineering, Inc., Osaka, Japan, 1996.)

The sensor will show sensitivity to a variety of deoxidizing gases, with relative sensitivity to certain gases optimized by the formulation of sensing materials and operating temperature. Since actual sensor resistance values vary from sensor to sensor, typical sensitivity characteristics are expressed as a ratio of sensor resistance in various concentrations of gases (R_S) over resistance in a certain concentration of a target gas (R_O).

3.1.3 Gas Sensor Array

Due to cross sensitivity and low selectivity of commercial gas sensors, their applications are limited to alarms activation. In order to use sensors for the analysis of gases, an array of sensors needs to be used which consists of several sensors with different sensitivity characteristics. The exposure of a gas sensor array (GSA) to a volatile organic compound produces a unique fingerprint. An example is shown in Fig. 6 where a GSA consisting of eight MOX sensors is exposed to 75 ppm of Ethane [21]. Each of the sensors shows a different response to the gas stimulus. Through the interpretation of the acquired data, it is possible to extract meaningful insights about the nature of the gas or mixture.

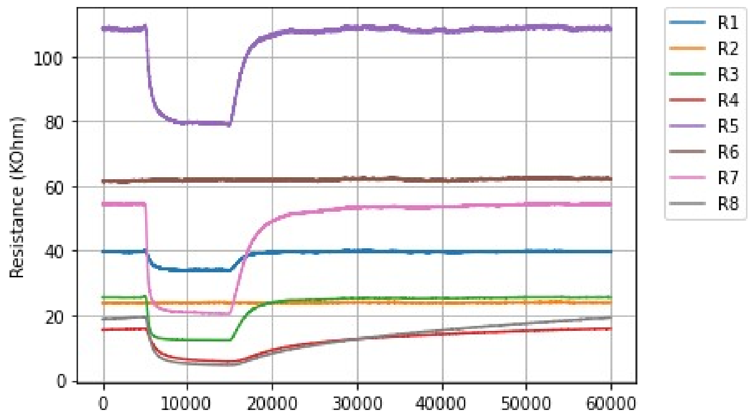


Figure 6: Response of an 8-sensor array to 75 ppm of ethanol

3.2 Signal processing

Signal processing is the first computational stage after the sensor array data has been sampled and stored into computer memory. The goal of this step is to extract relevant information from the sensor responses and prepare the data for multivariate pattern analysis.

3.2.1 Normalization

The first step in signal processing consists of normalizing our dataset in the 0–1 range. This was done by applying the following relation across each GSA fingerprint:

$$R_{norm} = \frac{R - R_{min}}{R_{max} - R_{min}} \tag{3.4}$$

where R is the acquired value, and R_{min} and R_{max} are the minimum and maximum value of each individual fingerprint of the GSA.

3.2.2 Downsampling

Downsampling produces an approximation of the sequence that would have been obtained by sampling the signal at a lower rate. We choose to downsample the signals (if necessary) by a factor of ten, that is, keeping one sample every tenth sample. So for example, if we choose to perform downsampling to a signal acquired at 100 Hz sampling rate, the resulting sampling rate is 10 Hz.

This procedure is useful as it reduces dramatically the size of the datasets, while keeping all the necessary informations in order to perform the analysis.
Fig. 7 shows the example depicted in Fig. 6 after normalization and downsampling:

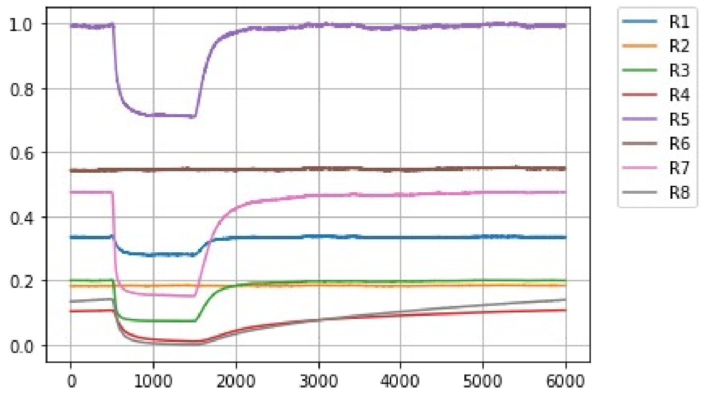


Figure 7: GSA response after normalization and downsampling

3.2.3 Response Matrix

Let us now consider an array of n discrete sensors, where each sensor i produces a time-dependent output signal $X_{ij}(t)$ in response to an odour j. The electrical signal depends on several physical parameters (fluid dynamics of odour delivery system, ambient pressure, temperature, humidity, etc.), but the output is expected to reach constant values when presented with a constant input stimulus. It has been common practice to use only the static values (i.e. steady-state) of the sensor signals rather than the dynamic (i.e. transient) response. In that case the response is a time-independent parameter, $X_{ij}(t) \longrightarrow X_{ij}$. [1].

In order to extract relevant key features from the data in terms of the static change in sensor parameter (e.g. resistance or conductivity), a good choice is to use a fractional difference model:

$$X_{ij} = \frac{X_{ij}^{odor} - X_i^0}{X_i^0} \tag{3.5}$$

where X_{ij}^{odor} is the response of sensor i to the sample odor j, and X_i^0 is the baseline signal, such as the value in ambient room air. The response generated by the *n*-sensor array to an odor *j* can then be represented by a time-independent vector:

$$X_j = (X_{1j}, X_{2j}, ..., X_{ij}, ..., X_{nj})^T \tag{3.6}$$

When the same array is presented to a set of *m* odors, the responses can be regarded as a set of *m* vectors, which are best represented by a response matrix $\mathbf{R}$:

$$\mathbf{R} = \begin{bmatrix} X_{11} & X_{12} & \cdots & X_{1m} \\ X_{21} & X_{22} & \cdots & X_{2m} \\ \vdots & \vdots & X_{ij} & \vdots \\ X_{n1} & X_{n2} & \cdots & X_{nm} \end{bmatrix} \tag{3.7}$$

Each column represents a response vector associated with a particular odour, whereas the rows are the responses of an individual sensor to the different measurands [1].

By the end of the data process stage, the final dataset is ready to be used for pattern recognition by utilizing learning algorithms. These techniques are non – parametric, in the sense that there is no need to assume any specific underlying probability density function for the sensor data.

3.3 Pattern Analysis

Data analysis provides a large number of available pattern recognition techniques that are being widely used in physical, chemical and engineering sciences. A pattern recognition method is the final stage in an intelligent olfactory system. This section aims at presenting the pattern recognition methods that we used in this thesis, for the discrimination of GSA examples.

3.3.1 Machine Learning

Machine learning is an application of artificial intelligence that involves the study of algorithms and statistical models that computer systems use to perform a specific task without using explicit instructions, relying on pattern recognition and inference instead. The process of learning begins with observations of data, such as examples, direct experience, or instructions, in order to look for patterns in data and make better decisions in the future based on the examples that the user provided. According to the definition by Mitchel (1997) "*A computer program is said to learn from experience E with respect to some class of tasks T and performance measure P, if its performance at tasks in T, as measured by P, improves with experience E*".

The algorithms that we used in this work can be characterized as supervised learning algorithms. These algorithms are able to build a mathematical model from a set of labeled data, that is, a set which contains both the inputs and the desired outputs. The precise form of this mathematical model is determined during the training phase, also known as the learning phase, on the basis of the set of the labeled data (we refer to this set as the *training set*). By the end of the training processes, the system can provide targets (predictions) for any new input. In order to evaluate the model's performance, we often measure its accuracy, which is the proportion of examples for which the model produces the correct output [22].

In this work we address a multiclass classification problem, that is, classifying sensor array fingerprints into several categories defined by the type and concentration of the gas stimulus that produced each fingerprint.

3.3.2 Multiclass Classification

A classification method always starts with a pair of variables $(\mathbf{x}, y)$. The first variable $\mathbf{x} \in \mathbf{X} \subset \mathbb{R}^d$ is an input vector and the set $\mathbf{X}$ is the input space. We assume that each input vector has a dimensionality d which is finite. The second variable $y \in \{1, ..., k\} = \mathbf{Y}$ denotes the class label of the classes $\{C_1, ..., C_k\}$. In order to train a classifier we assume that we have a training set D, which is a collection of paired variables $(\mathbf{x}_j, y_j)$, for $j = 1, ..., n$. The vector denotes the j-th sample input in , and denotes the corresponding class label. The relationship between $\mathbf{x}_j$ and y_j is specified by the target function $f : \mathbf{X} \to \mathbf{Y}$ such that $y_i = f(\mathbf{x}_i)$.

In this type of task the computer program is asked to specify which of k categories some input belongs to. To solve this task, the learning algorithm is usually asked to produce a function $f : \mathbb{R}^n \to \{1, ..., k\}$. When , the model assigns an input described by vector to a category identified by numeric code .

Many algorithms can perform multiclass classification and in this work we use some of them in order to select the one that has the highest accuracy. But first let us refer to the main idea behind each classifier.

3.3.2.1 Support Vector Machines (SVM)

The main idea of SVM is the construction of a hyperplane as the decision surface in such a way that the margin of separation between examples of different classes of a training set is maximized [23].

Considering the training example $\{(\mathbf{x}_i, y_i)\}_{i=1}^{N}$ we assume that the pattern represented by the subset $y_i = +1$ and the pattern represented by $y_i = -1$ are linearly separable. The equation of a decision surface in the form of a hyperplane that does the separation is

$$\mathbf{w}^T\mathbf{x} + b = 0 \tag{3.8}$$

where **x** is an input vector, **w** is an adjustable weight vector, and b is a bias. We may thus write:

$$\begin{aligned} \mathbf{w}^T\mathbf{x}_i + b &\geq 0 \text{ for } y_i = +1 \\ \mathbf{w}^T\mathbf{x}_i + b &< 0 \text{ for } y_i = -1 \end{aligned} \tag{3.9}$$

For a given weight vector **w** and bias b, the separation between the hyperplane defined in Eq. 3.8 and the closest data point is called the margin of separation, denoted by ρ. The goal of a support vector machine is to find the particular hyperplane for which ρ is maximized. Under this condition, the decision surface is referred to as the optimal hyperplane. Figure 8 illustrates the geometric construction of an optimal hyperplane for a two-dimensional input space.

If we denote by $\mathbf{w}_0$ and b_0 weight and bias values of the optimal hyperplane, it is proven that [23].

$$\rho = \frac{2}{\|\mathbf{w}_0\|} \tag{3.10}$$

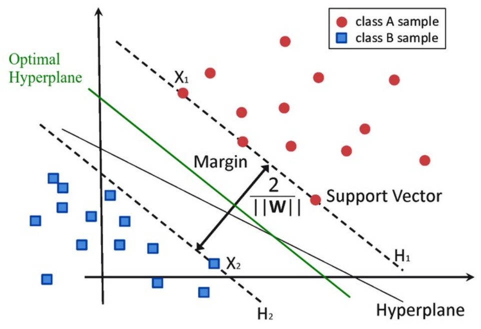


Figure 8: Support Vector Machine classification

SVM performs multiclass classification by splitting the dataset into multiple binary classification problems. For training the classifier, two different strategies are being used:

- *"One-versus-rest"*: A binary classifier is trained on each binary classification problem and predictions are made using the model that is the most confident

- *"One-versus-one"*: Splits the dataset into one dataset for each class versus every other class.

3.3.2.2 K-nearest-neighbors

Neighbors-based classification is a type of *instance-based learning* or *non-generalizing learning*: it does not attempt to construct a general internal model, but simply stores instances of the training data. Classification is computed from a simple majority vote of the nearest neighbors of each point: a query point is assigned the data class which has the most representatives within the nearest neighbors of the point. The algorithm implements learning based on the k nearest neighbors of each query point, where k is an integer value specified by the user. The optimal choice of the value k is highly data-dependent: in general a larger k suppresses the effects of noise, but makes the classification boundaries less distinct [24].

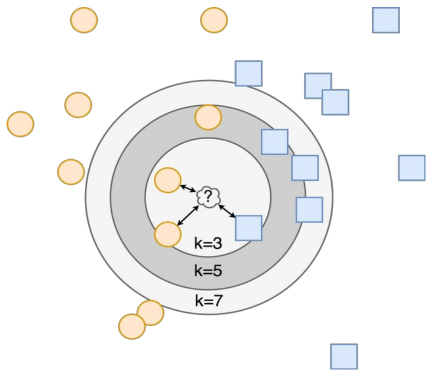


Figure 9: K-nearest neighbors classification

3.3.2.3 Decision trees

Decision trees are powerful and popular tools for classification and prediction. Decision trees represent rules, which can be understood by humans and used in knowledge system such as database. The goal is to create a model that predicts the value of a target variable based on several input variables. A decision tree is a flowchart like tree structure, where each internal node denotes a test on an attribute, each branch represents an outcome of the test, and each leaf node (terminal node) holds a class label. The topmost node in a tree is the root node. A tree can be “learned” by splitting the source set into subsets based on an attribute value test. This process is repeated on each derived subset in a recursive manner called recursive partitioning. The recursion is completed when the subset at a node all has the same value of the target variable, or when splitting no longer adds value to the predictions [25].

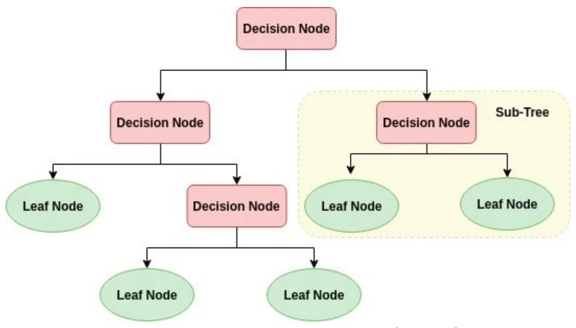


Figure 10: Decision Tree Algorithm

3.3.2.4 Random Forest

Random forest is an ensemble method as it uses multiple different decision trees estimators for making predictions. Each decision tree takes a different random sample from our set of training data and constructs a tree from it. Then each resulting tree can vote on the right result [26].

3.3.3 Artificial Neural Networks

Artificial neural networks (ANN) are information-processing mathematical models inspired by the biological neural networks that constitute the human brain. As its original counterpart, they are able to learn from observational data, that is, by considering examples without being programmed with any task-specific rules. The basic component of a NN is the artificial neuron. An artificial neuron, denoted with j is a processing unit which performs the following operations:

- It receives an input signal x_i from the synapse i
- It multiplies the signal by the synaptic weight w_{ji}
- It sums all input signals x_i with their respective weights w_{ji}, for all the synapses $i = 1, \ldots, n$ and adds a bias term b_i.
- It processes the sum of the input signals through an activation function $\varphi(\cdot)$ and outputs the result y_j.

Fig. 11 shows the graphical representation of a neuron.

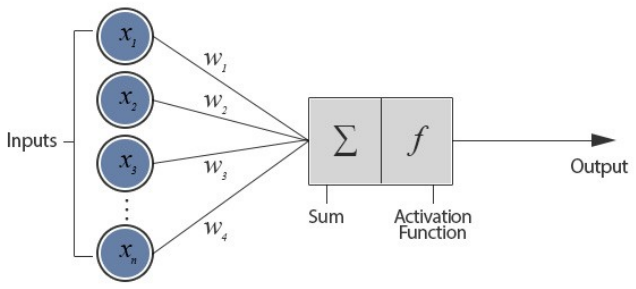


Figure 11: Graphical representation of a neuron

In mathematical terms, the neuron j can be described by the equation:

$$y_j = \varphi(v_j) \tag{3.11}$$

where

$$v_j = \sum_{i=1}^{n} w_{ji} x_i + b_j \tag{3.12}$$

The purpose of the activation function is to add the non-linearity needed in the network.

In this context, a neural network is an oriented graph with neurons being the nodes of the graph and the synapses being the oriented edges. The synaptic weights are calibrated through a training process based on observational data.

Depending on the interconnection of neurons, different types of neural networks arise. Among them, the most popular and widely applied type is the feed-forward neural network (FFNN), also known as a multilayer perceptron (MLP) (Fig. 12). In terms of the architecture, an FFNN consists of the input layer, the hidden layer(s) and the output layer. Neural networks with more than one hidden layer are referred to as deep neural networks. In terms of connectivity, in FFNN neurons from a layer can only be connected with neurons from the next layer towards the output layer. This means that the information moves in one direction, forward, from the input nodes, through the hidden nodes and to the output nodes [27].

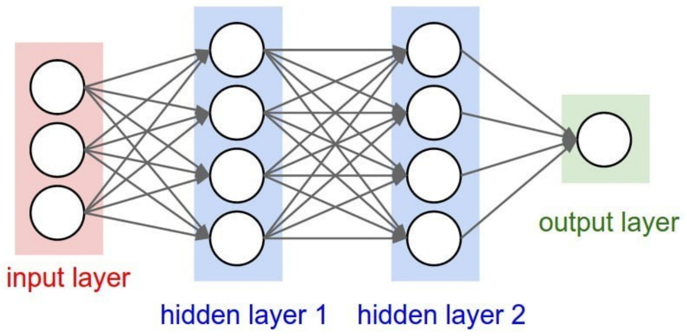


Figure 12: FFNN network architecture

The propagation of the information from the input to the output layer is called the forward pass. During this pass the weights remain unaltered throughout the network and the function signals of the network are computed on a neuron-by-neuron basis.

When the propagation reaches neuron j of the output layer, an error signal is computed as follows:

$$e_j = \hat{y}_j - y_j \tag{3.13}$$

where $\hat{y}_j$ is the desired output of neuron j. The desired output is provided along with the input data at the beginning of the training process, as a set of N_{tr} labelled data $\{\hat{\mathbf{x}}, \hat{\mathbf{y}}\}_{1 \leq k \leq N_{tr}}$. The output of Eq. 3.13 will be used by a cost function $\mathcal{E}$, such as the least mean-squared algorithm (LMS), in order to compute the instantaneous error energy $\mathcal{E}_j$ of neuron j, and then, the total instantaneous error energy of the whole network $\mathcal{E}$:

$$\mathcal{E} = \sum_{j \in C} \mathcal{E}_j \tag{3.14}$$

where set C includes all the neurons in the output layer.

Now the training of the network consists in finding the optimal weights that minimize the cost function $\mathcal{E}$. In order to solve this (non-convex) optimization problem we make use of the so-called back-propagation algorithm [27] [23]. During the backward pass, a correction Δw_{ji} is applied to the synaptic weight w_{ji}, that is proportional to the partial derivative $\partial \mathcal{E} / \partial w_{ji}$:

$$\Delta w_{ji} = -\eta \frac{\partial \mathcal{E}}{\partial w_{ji}} \tag{3.15}$$

where η is the learning rate of the back-propagation algorithm. After the computation of the weight correction Δw_{ji} by the back-propagation algorithm, this information is now available for other algorithms, such as gradient descent or adaptive moment estimation, to perform parameter update:

$$w_{ij} \longleftarrow w_{ij} + \Delta w_{ij} \tag{3.16}$$

By the end of the training process all weights have been adjusted to their optimal values, resulting to the global minimum of the loss function.

Based on the above, FFNNs essentially establish a non-linear map from the space of the input data to the space of the output data (Fig. 13).

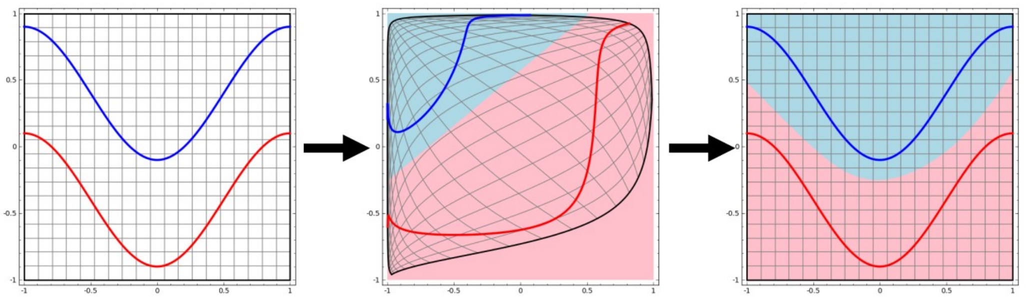


Figure 13: Mapping the space of the input data to the space of the output data

Even though FFNN are universal function approximators, yet, they do not provide any guidelines for selecting the exact network architecture, nor the number of samples required to train the network. In addition, to identify the network parameters only heuristics can be employed to solve the non-convex optimization problem. As a consequence, the optimal network architecture and parameters are achieved in practice via a trial-and-error process which can be quite cumbersome (if not intractable) for large-scale problems [28].

3.3.3.1 Convolutional Neural Networks

Convolutional Neural Networks (CNN) is another class of deep neural networks commonly applied to tasks such as pattern recognition in images and videos, image segmentation and classification, time-series analysis and more. The main advantage they have over FFNNs is that they can handle data with high dimensionality. In CNNs the input is a tensor (which is viewed as nD-array), such as a set of images, which can be given by the 4D-matrix: (number of images) × (image height) × (image width) × (input channels)[27]. Image height and width represents the number of pixels each dimension has. The number of input channels is 1 in the case of greyscale images, and 3 for coloured images (RGB channels).

The name "convolutional neural networks" indicates that the network employs a mathematical operation called "convolution" which is a specialized kind of linear operation. Convolutional networks are simply neural networks that use convolution in place of general matrix multiplication in at least one of their layers [22].

The input image in a CNN is represented as an array where the values correspond to the color of a pixel of the image. Then the convolution operation takes place between the input array and a pre-selected kernel. We call the output of the convolution operation the feature map. Fig. 14 provides an example of a discrete convolution where the input I and the kernel K produce the feature map $I * K$.

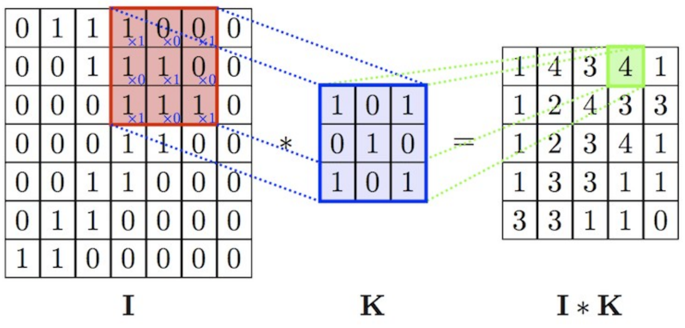


Figure 14: An example of 2D convolution operation

At each location, the product between each element of the kernel and the input element it overlaps, is computed and the results are summed up to obtain the output in the current location. If there are multiple input feature maps (as it is the case in coloured images), the kernel will have to be 3-dimensional – or, equivalently each one of the feature maps will be convolved with a distinct kernel – and the resulting feature maps will be summed up element-wise to produce the output feature map.

The output size of the convolution layer can be affected by the input size, the kernel size, the stride (the distance between two consecutive positions of the kernel in both horizontal and vertical directions) and the zero padding, which is the number of zeros concatenated at the beginning and at the end of both axes of the input [29] [22] [30].

- **Pooling layer**

In addition to discrete convolutions themselves, pooling operations make up another important building block in CNNs. Pooling operations reduce the size of feature maps by using some function to summarize sub-regions, such as taking the average or the maximum value.

Pooling works by sliding a window across the input and feeding the content of the window to a pooling function. In some sense, pooling works very much like a discrete convolution, but replaces the linear combination described by the kernel with some other function [30].

- **Fully connected layer**

This layer forms the last block of the CNN architecture, related to the task of classification. The output matrix of the convolution block (convolution layer and pooling layer) is a 3-D matrix in the case where the input is a coloured image. Fully connected layer flattens the shape of the output into a 1-D matrix. It is essentially a fully connected simple neural network, consisting of two or three hidden layers and an output layer that performs the work of classification.

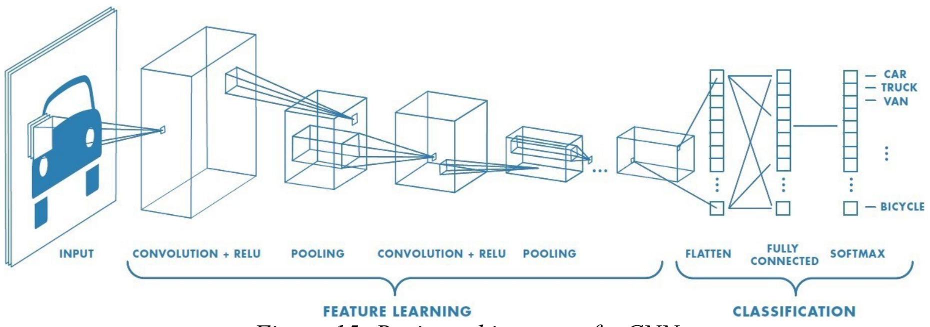


Figure 15: Basic architecture of a CNN

3.3.3.2 Transfer Learning

It is a popular approach in deep learning to reuse a model developed for a task as the starting point for a model on a second task. Transfer learning and domain adaption refer to the situation where what has been learned in one setting is exploited to improve generalization in another setting [22].

A pre-trained model is typically trained one a huge dataset and it is capable of classifying data into a large number of categories. By using those models we have the freedom to choose weather or not we want to use the adjusted weights of the pre-trained network, the layers we want to train, and the number of classes.

The use of CNN pre-trained models for pattern-recognition of GSA data is being investigated by Han et al. [15]. In that paper the authors compare the performance of various models in order to select the one which discriminates volatile compounds with the highest accuracy. Bellow we briefly present one of those models which we also used (VGG-16) for classifying GSA data.

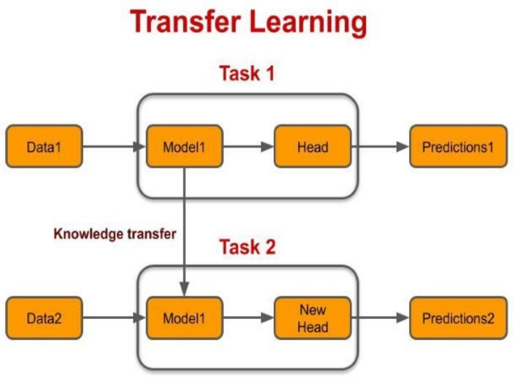


Figure 16: Transfer Learning

- **VGG-16**

VGG-16 is a convolutional neural network model proposed by K. Simonyan and A. Zisserman [31]. The model achieves 92.7% test accuracy in ImageNet dataset [32], which is a dataset of over 14 million images belonging to 1000 classes. The architecture of the network is depicted in Fig. 17.

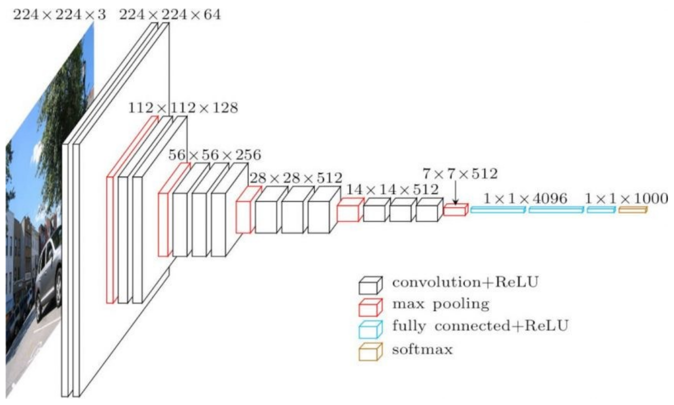


Figure 17: VGG-16 architecture

The input layer is of fixed size 224×224 RGB image. The image is passed through a stack of convolutional layers, where the filters were used with a 3×3 receptive field. The convolution stride is fixed to 1 pixel; the spatial padding of convolutional layer input is such that the spatial resolution is preserved after convolution, i.e. the padding is 1-pixel for 3×3 convolutional layers. Spatial pooling is carried out by five max-pooling layers, which follow some of the convolutional layers. Max-pooling is performed over a 2×2 pixel window, with stride 2.

Three fully-connected layers follow a stack of convolutional layers: the first two have 4096 channels each, the third performs 1000-way classification and thus contains 1000 channels. The final layer uses the softmax activation function to normalize the output of a network to a probability distribution over predicted output classes. The configuration of the fully connected layers is the same in all networks. All hidden layers are equipped with the rectification (ReLU) non-linearity.

4 *Experimental Process*

4.1 Sensors – GSA

The experiments were performed using three *MiCS 3110* sensors which are designed for ethanol detection (*Micro Chemical Systems MiCS-3110*). The sensor response is shown in Fig. 18 . The mode of operation for the heater resistance R_H is a constant voltage mode at V_H=2.4 V. This causes the sensing resistor (R_S) temperature to reach about 430 ºC. V_H was applied to each of the sensors using the device shown in Figure 19. The measurement circuit is shown in Fig. 20.

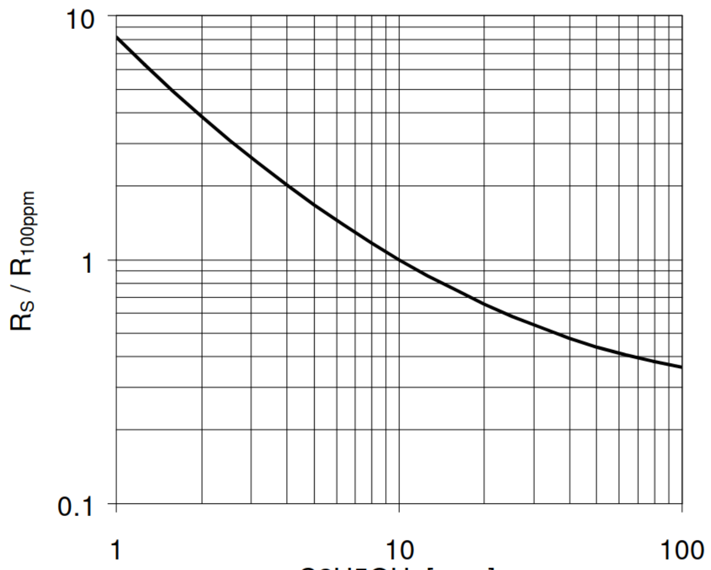


Figure 18: R_S / R_{10ppm} as a function of gas concentration at 23°C.
(MicroChemical Systems MiCS-3110 Data Sheet)

Figure 19: Heater Voltage supplier

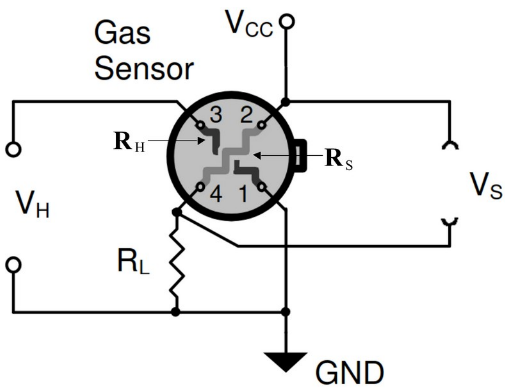


Figure 20: Measurement circuit of MiCS 3110 (top view).(MicroChemical Systems MiCS-3110 Data Sheet)

A GSA was constructed by attaching each of the sensors to a circuit as shown in Figure 21. During the experiments the sensing resistance R_S is being measured. For this purpose we used the Keithley 2400 Source Meter Unit (SMU) (Fig. 22) and performed I-V measurements while applying a constant voltage $V_S = 5$ V.

In order to measure R_S for each of the sensors in the array it was necessary to construct a circuit which allows V_S to be applied to a different sensor each time. This was done by using electrically operated switches (relays). When a relay is 'open', gives access to SMU to apply V_S to its corresponding sensor, while the rest relays remain 'closed'. This procedure was controlled by a microcontroller (Arduino – Fig. 23) which 'opens' and 'closes' each relay in a circular manner. The delay time between two consecutive measurements was 100 ms, resulting to a delay time of 300 ms between two measurements of the same sensor. This leads to a sampling rate of 3.33 Hz.

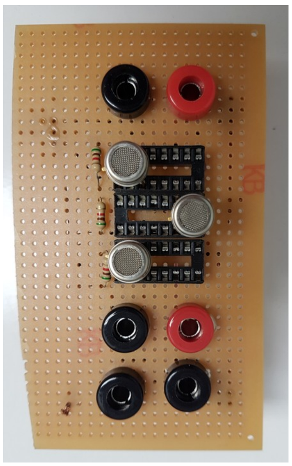


Figure 21: Gas Sensor Array

Figure 22: Keithley 2400 Source Meter Unit

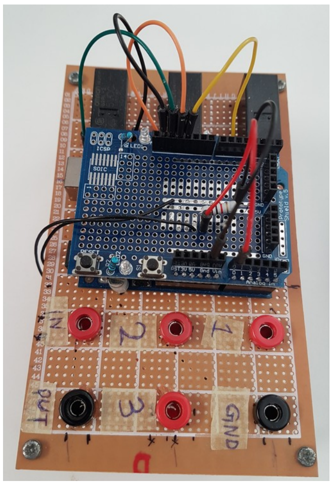


Figure 23: Arduino board and circuit for the control of the acquisition

4.2 Odor delivery system

The GSA was placed in an airtight test chamber of known inner volume. The sample gas was taken from the gas cylinder using a syringe and then injected into the chamber. In order to ensure that the gas was uniformly mixed, a compact fan was placed inside the test chamber.

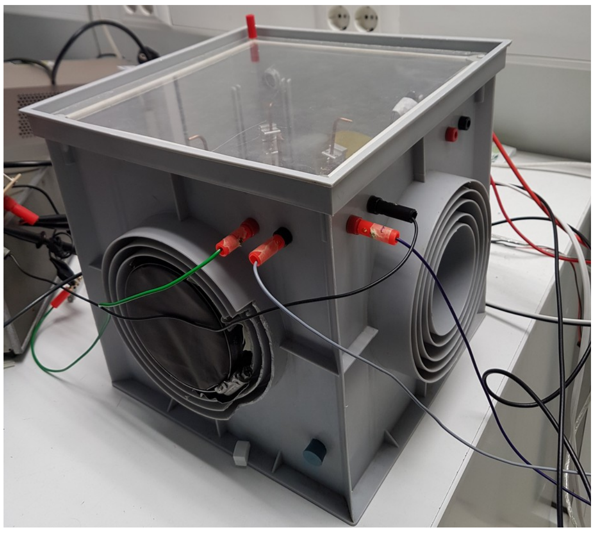


Figure 24: Test chamber

4.3 Experimental Protocol

The experimental protocol includes three stages: the array is first exposed to a gas reference (ambient air), then to the gas sample, and finally to the reference again to recover the initial state. The duration of each stage varies for the datasets we used.

4.4 Datasets

Along with the data collected from the experiments, we also used some freely available datasets from the UCI Machine Learning Repository. In this section we briefly present all the datasets that we used.

4.4.1 Dataset 1 – Experimental Dataset

The dataset that was created from the experiments consists of 252 experiments (examples). The duration of each experiment was 5 minutes. During the first minute the response of the GSA in ambient air was recorded (baseline). Then the gas under test was injected into the chamber and for the next three minutes the response of the GSA was recorded. Next the gas was released from the chamber and for the remaining minute we recorder the recovery response of the array.

Due to some unexpected technical limitations we had to stop the experiments long before we came up with the initially designed dataset. As a consequence the final dataset is too unbalanced in terms of gas concentrations. Hence, the examples in the dataset can be distributed into three classes depending only on the target gas: "Hydrogen", "Butane", "Mixture". The number of examples each class contains are 78 for "Hydrogen", 95 for "Butane" and 79 for "Mixture". Each example has three features (sensor readings) and 1000 rows (300 sec duration of experiment with 0.3 sec sampling rate). An example of the acquired time-series can be seen in Fig. 25.

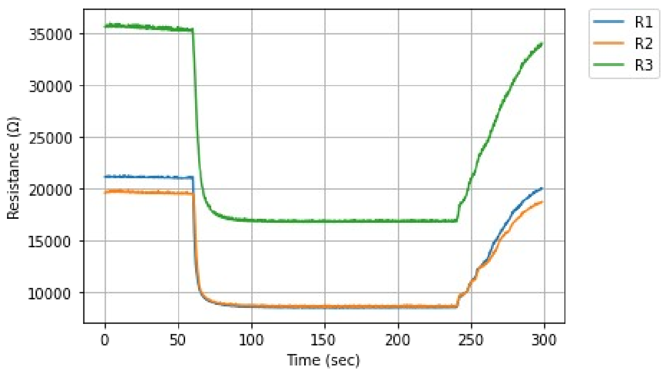


Figure 25: Response of the GSA when exposed to Butane

Fig. 26 shows the response of each individual sensor to the three volatiles used in this dataset.

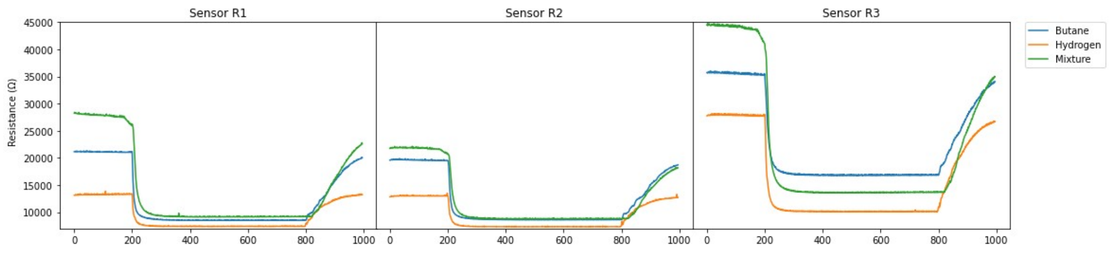


Figure 26: Response of each sensor to different volatile compounds

4.4.2 Dataset 2 – "Gas sensor array exposed to turbulent gas mixtures"

This dataset [33] provides the acquired time series from an array of 8-MOX gas sensors exposed to turbulent gas mixtures. The array was exposed to mixtures of Ethylene with Methane or Carbon Monoxide. Each volatile was released at four different concentrations: zero (non-mixture example), low, medium and high providing up to 30 different mixture configurations: 15 mixtures of Ethylene with CO and 15 mixtures of Ethylene with Methane. Each configuration was repeated 6 times. Hence, the complete dataset was composed of 180 measurements. Each measurement had a total duration of 300 seconds: clean air for 60 sec, gas exposure for 180 sec and 60 sec recovery time. The sampling rate was 10 Hz resulting to approximately 3000 rows for a typical dataframe.

We assumed that each example belongs to a class defined by the type of the volatile. As a consequence, we addressed a classification problem with five classes: "CO", "Ethylene", "Methane", "CO+Ethylene", "Methane+Ethylene". Figure 27 illustrates the response to mixture of ethylene and carbon monoxide.

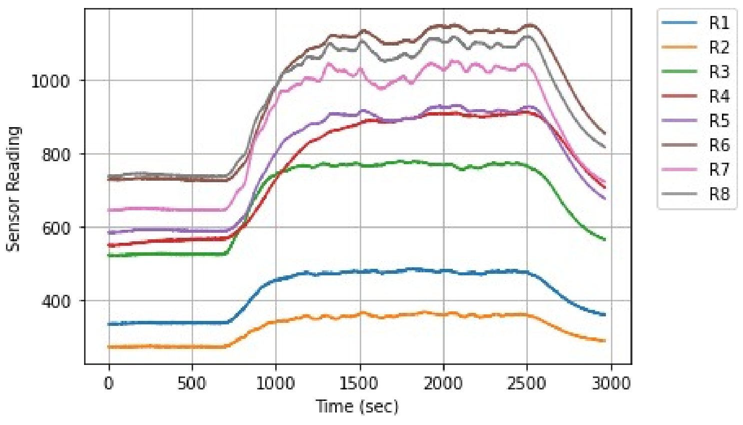


Figure 27: Response of the GSA to mixture of Ethylene and CO

4.4.3 Dataset 3 – "Twin gas sensor arrays"

Next we used the dataset mentioned in [21]. The dataset includes the recordings of five replicates of an 8-sensor array. The units were exposed to ten concentration levels of Ethanol, Methane, Ethylene, and Carbon Monoxide. For each target-gas we assumed the existence of three concentration categories (Table 4.1): Low (concentrations 1 – 3), Medium (concentrations 4 – 7) and High (concentrations 8 – 10). Each concentration level consists of 16 examples, thus, the number of instances each concentration category has are: Low: 192, Medium: 256, High: 192. Thus the dataset consists of 640 examples.

For the classification task we assume the existence of 12 classes; three (low, medium, high) for each of the four volatiles.

Table 4.1 Concentration levels in Dataset 3 (in ppm)

	Low			Medium				High		
Ethanol	12.5	25	37.5	50	62.5	75	87.5	100	112.5	125
CO	25	50	75	100	125	150	175	200	225	250
Ethylene	12.5	25	37.5	50	62.5	75	87.5	100	112.5	125
Methane	25	50	75	100	125	150	175	200	225	250

The duration of each experiment was 600s; 0 – 50s exposure to clean air, 50s – 150s exposure to gas stimuli, 150s – 600s recovery time. The sampling rate was 100 Hz and we down sampled it at 10 Hz.

A typical response of the GSA for this dataset can be seen in Figure 28.

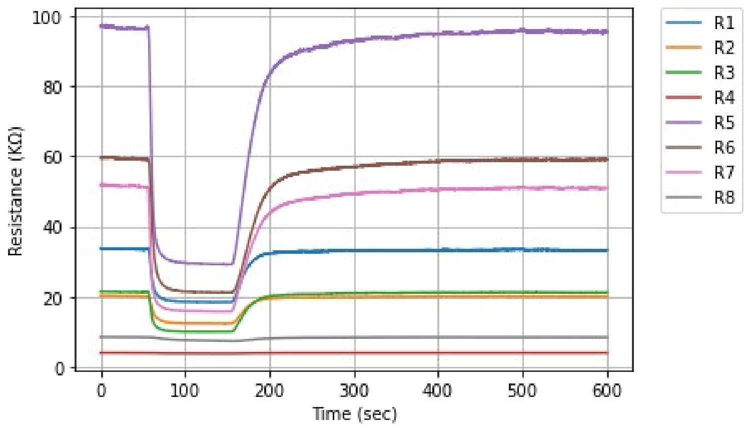


Figure 28: Response of 8-sensors array to 100 ppm of Methane

4.5 Data Augmentation

Data augmentation are techniques used to increase the amount of data by adding slightly modified copies of already existing data. We used three techniques in order to extend the number of examples:

- *Jitter*: addition of gaussian noise on existing time-series
- *Scale:* slightly change the position of each time-series on the vertical axis
- *Interpolation:* cubic spline for generating random curves

We should mention that the augmentation is only applied on the training sets. It is important to use only original samples for validation and test sets. The above functions were retrieved from [34].

4.6 Pictures creation & processing

Each of the aforementioned datasets were further processed in order to create the images that we fed into the pre-trained convolutional neural network for image classification (VGG-16). The first step in this process was the illustration of the response of the GSA to each stimulus. Next, each figure was resized to (224, 224) pixels and saved to computer. In order to create inputs for the VGG-16, a 3-D matrix was extracted from each image. Each matrix has (224, 224, 3) size which corresponds to the number of pixels in horizontal and vertical axis, and 3 is the number of channels a coloured image has: red – green – blue. Each entry in these tensors is a number between 0–255; 0 being black and 255 white. The final step consists of scaling these values to range 0–1.

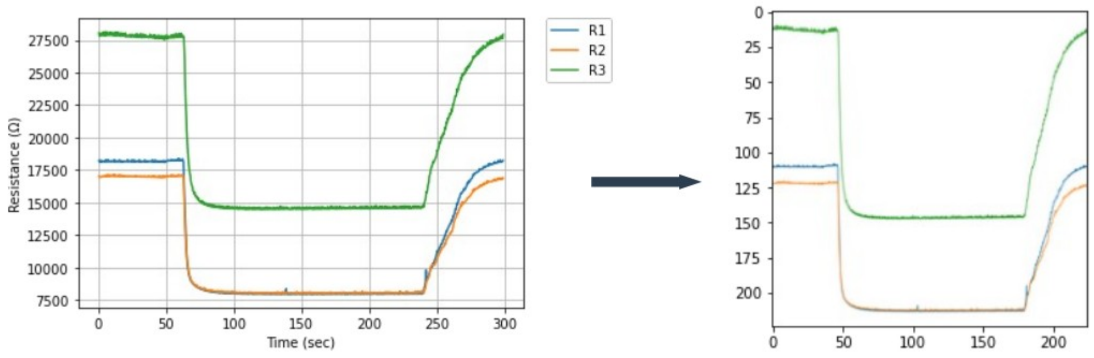


Figure 29: Initial plot and resized image – input for the VGG-16

4.7 Frameworks and libraries

The programming language used for the analysis was Python 3.7. There are many open source frameworks available for implementing ML and DL algorithms. In the present work the ones we used was Tensorflow 1.14.0 [35] and Scikit Learn 0.23.2 [24]. Other python libraries used are: Numpy 1.19.2 for numerical computations and Pandas 1.1.3 for data management.

5 *Results*

5.1 Experimental Dataset

We first examine the performance of VGG-16 on the experimental dataset. The dataset was augmented according to section 4.5 and the extended dataset consists of 618 examples. The dataset was split into training, validation and test set:

- Training set examples: 488 (78.9%)
- Validation set examples 65 (10.55%)
- Test set examples: 65 (10.55%)

We should note that all synthetic examples were used only in the training set.

The output of the network is a probability distribution over three classes: 'Butane', 'Hydrogen' and 'Mixture'. The network was trained over the training set and we used the validation set for parameter selection. The network's hyper-parameters were:

- Epochs: 100
- Initial Learning Rate: 5e-3
- Batch Size: 256
- Kernel Regularization: L1 = 1e-3, L2 = 1e-3
- Dropout: 0.1

The learning rate was programmed to decay as a function of epochs. As a result the last value of the learning rate was 0.0185. The selection of hyper-parameters was done by trial-and-error strategy. The trained network was evaluated over the test set and an accuracy of 87.7% was obtained. Below we present the learning curves and the confusion matrix which sums up the classification results over the test set.

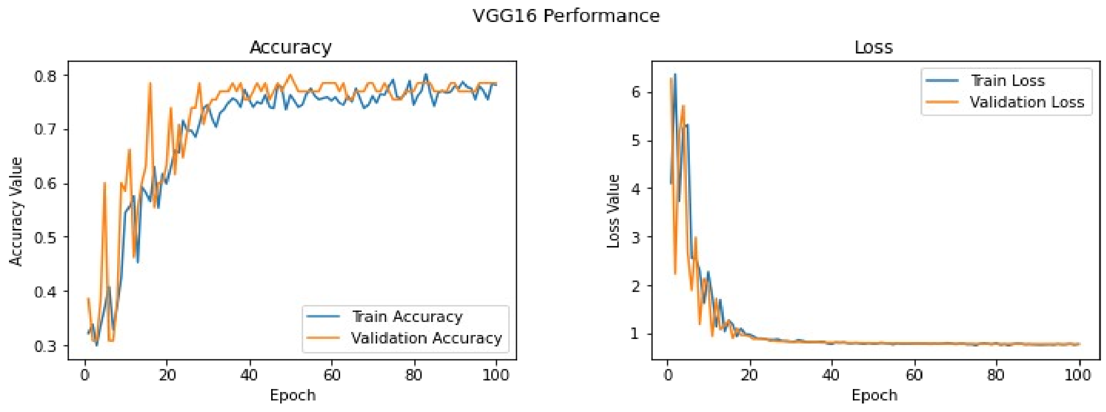


Figure 30: Learning curves of the VGG-16 for dataset 1

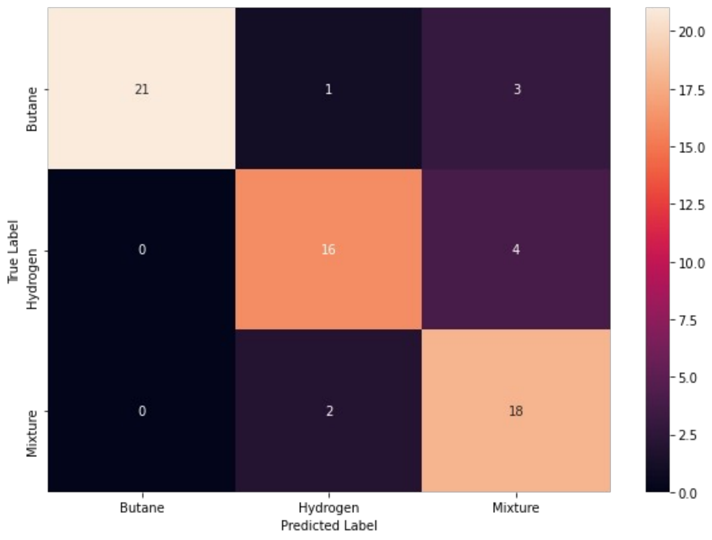


Figure 31: Classification result on the test set of dataset 1

5.2 Dataset 2

We also tested the performance of VGG-16 on the dataset–2. The dataset was split as follows:

- Training set examples: 360 (66.6 %)
- Validation set examples 90 (16.6 %)
- Test set examples: 90 (16.6 %)

The network classifies the examples into 5 categories: “Methane”, “CO”, “Ethylene”, “CO+Ethylene”, “Methane+Ethylene”. The selected hyper-parameters were:

- Epochs: 150
- Learning Rate: 0.01
- Batch Size: 128
- Kernel Regularization: L2 Regularizer(0.01)

The learning rate was not changed during the training process. The accuracy obtained was 94.4 %.

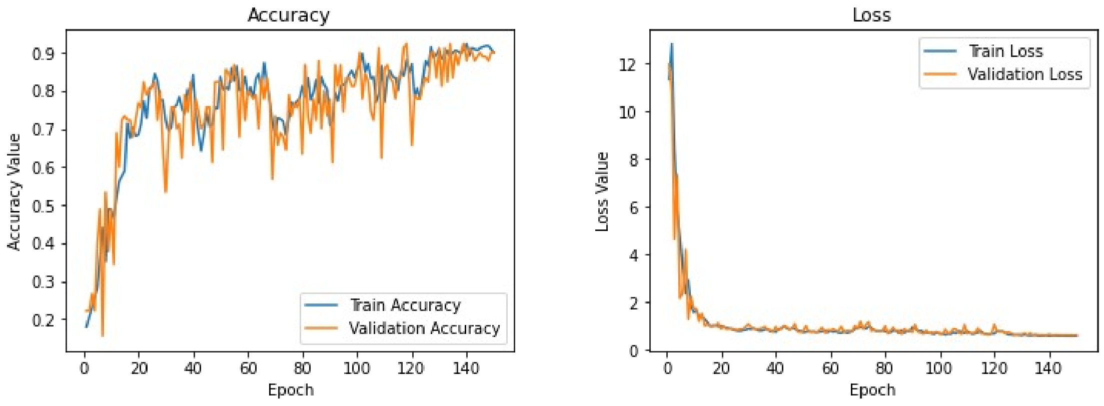
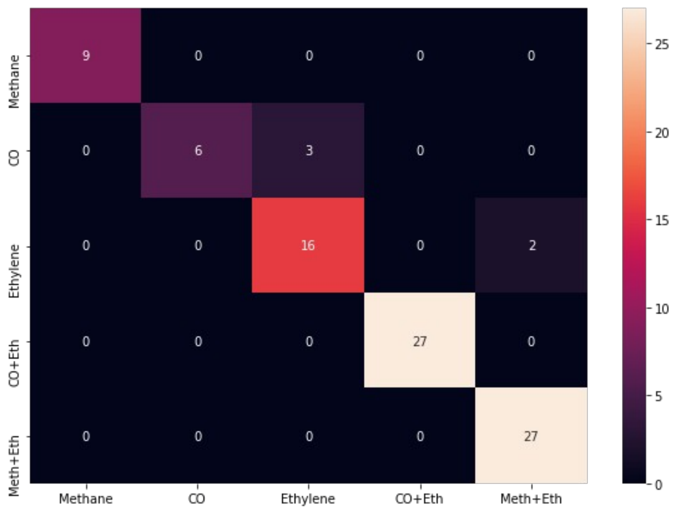


Figure 32: Learning curves of the VGG-16 for dataset 2



	Methane	CO	Ethylene	CO+Eth	Meth+Eth
Methane	9	0	0	0	0
CO	0	6	3	0	0
Ethylene	0	0	16	0	2
CO+Eth	0	0	0	27	0
Meth+Eth	0	0	0	0	27

Figure 33: Classification result on the test set of dataset 2

5.3 *Dataset 3*

➢ **VGG-16**

Dataset 3 was also used to evaluate the performance of VGG-16. The dataset was split as follows:

- Training set examples: 432 (67.5 %)
- Validation set examples 104 (16.25 %)
- Test set examples: 104 (16.25 %)

The network classifies the examples over 12 categories (see §4.4.3). The selected hyper-parameters were:

- Epochs: 300
- Initial Learning Rate: 5e-3 – Final learning rate: 2.4768e-4

- Batch Size: 256
- Kernel Regularization: L1 = 1e-3, L2 = 1e-4

The accuracy achieved was 55.8% on the test set. Bellow are the plots of the learning curves and the corresponding confusion matrix.

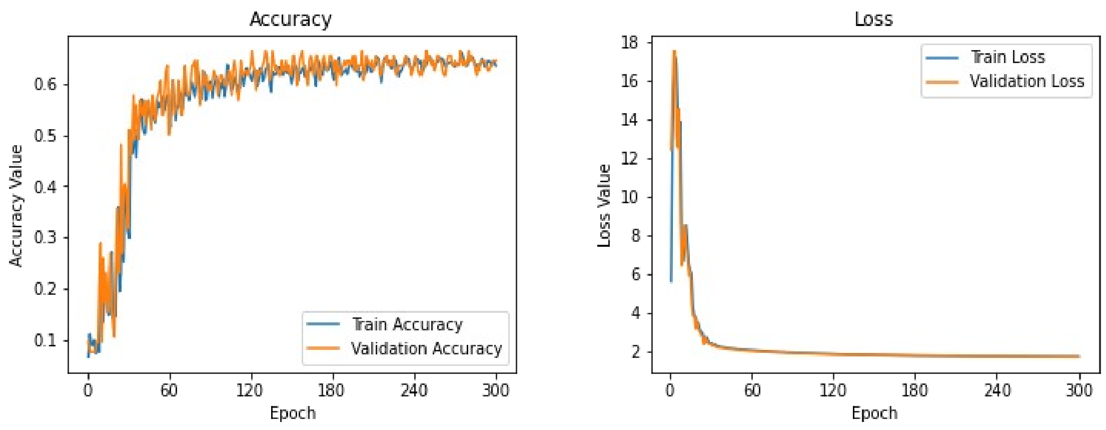


Figure 34: Learning curves of VGG-16 on dataset 3

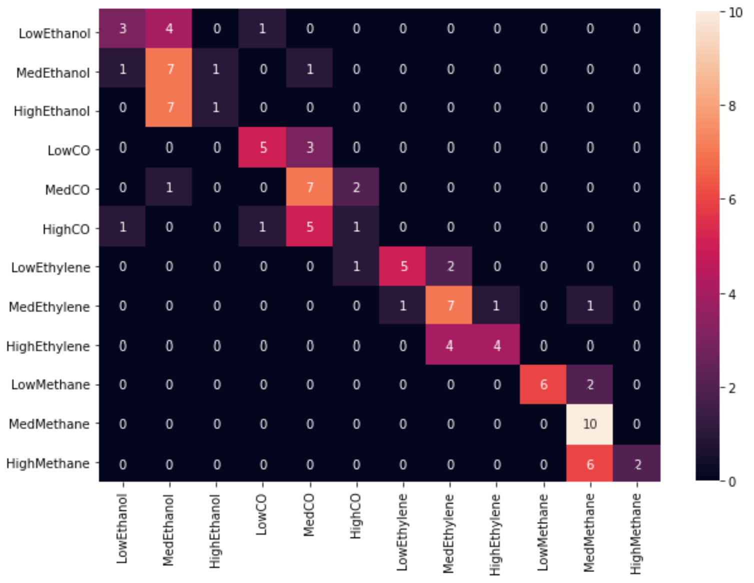


Figure 35: Classification results of VGG-16 on the test set

We then tried to increase the accuracy of the model by generating more examples. For this purpose we used the "Jitter" and "Scale" functions described in section 4.5. The augmented dataset consists of 1920 examples and it was split as follows:

- Training set examples: 1332 (70 %)
- Validation set examples 292 (15 %)
- Test set examples: 292 (15 %)

We trained the network for 300 epochs while decaying its learning rate from 1e-2 to 1.4e-3. The batch size was 256. The network now seems to classify the given examples more accurately as the accuracy on the test set is 67.1%. Bellow we present the performance measures:

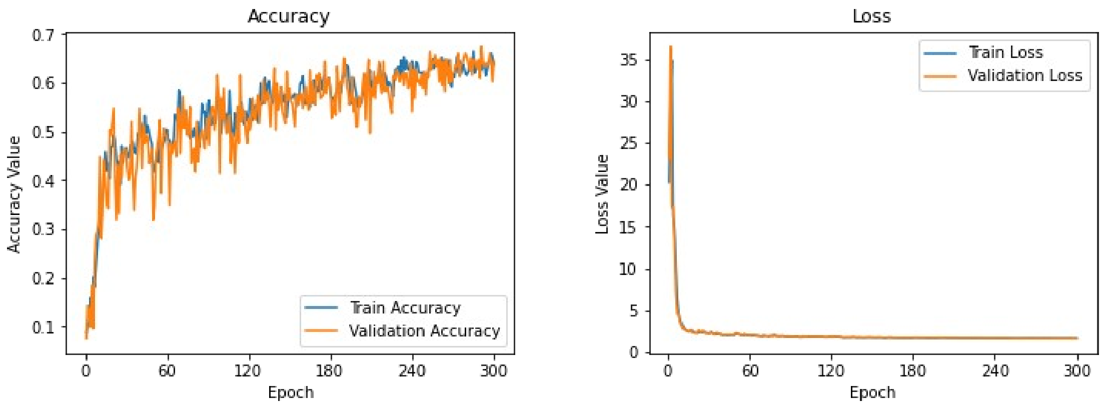
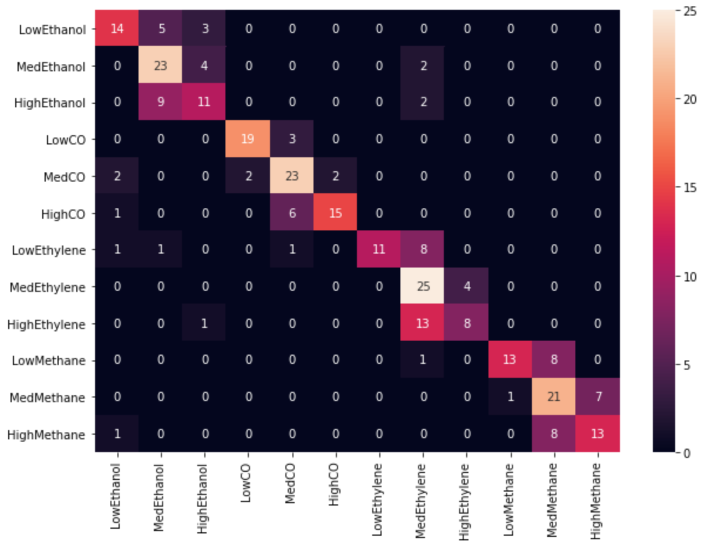


Figure 36: Learning curves of VGG-16 on the augmented dataset



	LowEthanol	MedEthanol	HighEthanol	LowCO	MedCO	HighCO	LowEthylene	MedEthylene	HighEthylene	LowMethane	MedMethane	HighMethane
LowEthanol	14	5	3	0	0	0	0	0	0	0	0	0
MedEthanol	0	23	4	0	0	0	0	2	0	0	0	0
HighEthanol	0	9	11	0	0	0	0	2	0	0	0	0
LowCO	0	0	0	19	3	0	0	0	0	0	0	0
MedCO	2	0	0	2	23	2	0	0	0	0	0	0
HighCO	1	0	0	0	6	15	0	0	0	0	0	0
LowEthylene	1	1	0	0	1	0	11	8	0	0	0	0
MedEthylene	0	0	0	0	0	0	0	25	4	0	0	0
HighEthylene	0	0	1	0	0	0	0	13	8	0	0	0
LowMethane	0	0	0	0	0	0	0	1	0	13	8	0
MedMethane	0	0	0	0	0	0	0	0	0	1	21	7
HighMethane	1	0	0	0	0	0	0	0	0	0	8	13

Figure 37: Classification results on the augmented dataset

In an attempt to further increase the accuracy of the model, we tried to illustrate the GSA responses in a different way. Considering that a typical response contains a lot of useless information, we only included the transient recordings between 50 – 200 seconds. The transformation for a typical picture is illustrated in Fig. 38.

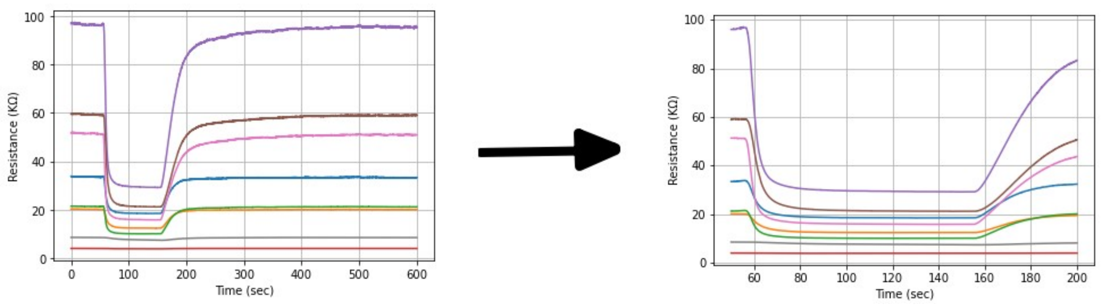


Figure 38: Modification on the pictures of dataset 3

For the modified pictures, we first tested the performance of the network on the original dataset and then on the augmented dataset. The accuracy achieved for the original dataset was 63.5%. The corresponding learning curves and confusion matrix are depicted bellow.

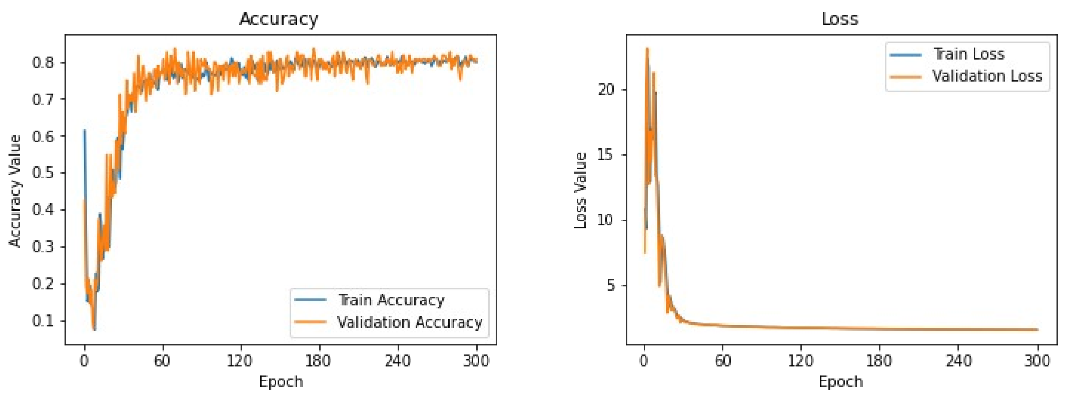


Figure 39: Learning curves of VGG-16 on the modified pictures of the original dataset

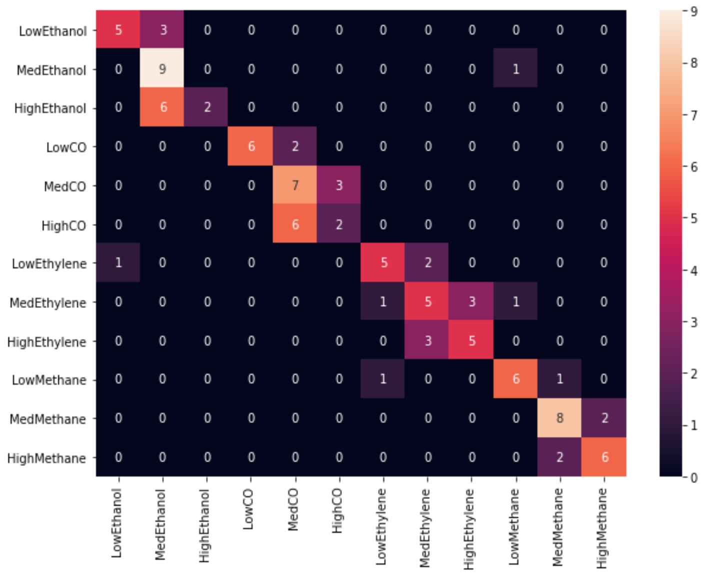


Figure 40: Classification results of VGG-16 for modified images

We then used the same procedure for the augmented dataset. Here we observe a significant increase of the model's accuracy, since the accuracy now is 79.5%. The performance measures are depicted bellow:

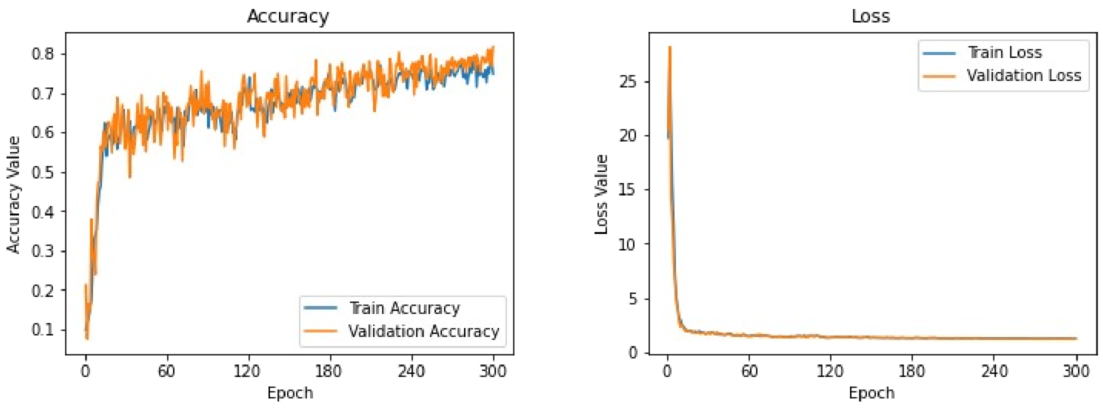


Figure 41: Learning curves of VGG-16 on the modified-augmented dataset

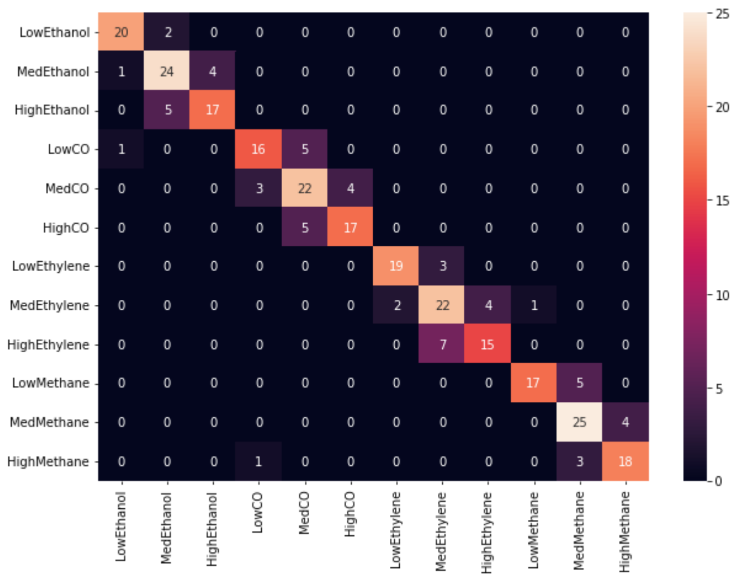


Figure 42: Classification result for the modified images of the augmented dataset

All the above results can being summarized in the following table:

	Original Dataset		Augmented Dataset	
	Normal Illustration	Modified Illustration	Normal Illustration	Modified Illustration
Train Accuracy (%)	55.8	63.5	67.1	**79.5**

➢ **ML algorithms**

Apart from VGG-16, we also used Machine Learning algorithms to perform the classification task. We used some of the ML algorithms most commonly referred to the bibliography as pattern-recognition methods applied to gas sensor array data.

- Support Vector Classifier (SVC)
- K-Nearest Neighbors (KNN)
- Random Forest (RF)
- Decision Tree (DT)

Now each example is not represented as an image, but as the fractional difference between the baseline response of each sensor, and its steady-state response. Therefore the dataset was organized in a matrix similar to Eq. 3.7. For the implementation of the ML algorithms we use the non-augmented dataset. We split the dataset into a training set (80% – 512 instances) and a test set (20% – 128 instances). In order to select the parameters for each classifier, we used a brute-force strategy in which we kept one parameter constant and alter all the others. In order to avoid overfitting, we selected the combination of parameters that minimize the difference between the train and the test accuracy. The results were validated using k-fold cross validation with *k = 5* (i. e. the split of the dataset into *k* different sets (folds) and selecting each time a different fold to use as the test set, while the rest *k - 1* folds were used for the training). Bellow we present the performance of each classifier.

- **Support Vector Classifier**

We altered the regularization parameter *C* of SVC. All the other parameters were set to their default vales. The minimum difference between the train and the test accuracy was obtained for *C* = 0.1. The results of the 5-fold cross validation are:

	Train Accuracy	Test Accuracy
k = 1	0.4	0.41
k = 2	0.44	0.41
k = 3	0.43	0.45
k = 4	0.44	0.41
k = 5	0.43	0.43
Mean	**0.43**	**0.42**

The results for the rest of the selected classifiers are:

- **k-Nearest Neighbors**

	Train Accuracy	Test Accuracy
k = 1	0.68	0.66
k = 2	0.65	0.62
k = 3	0.66	0.63
k = 4	0.68	0.64
k = 5	0.66	0.66
Mean	**0.67**	**0.64**

- **Random Forest**

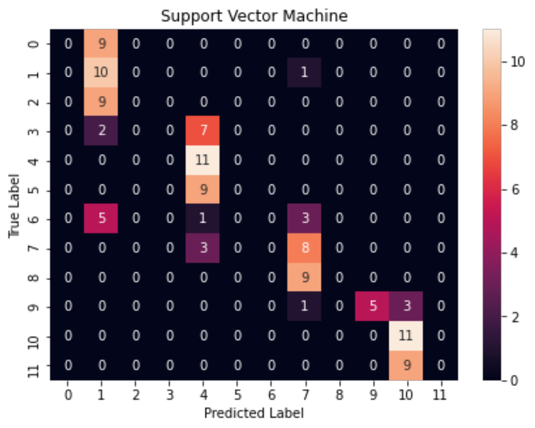
	Train Accuracy	Test Accuracy
k = 1	0.66	0.66
k = 2	0.65	0.60
k = 3	0.64	0.59
k = 4	0.58	0.54
k = 5	0.60	0.59
Mean	**0.63**	**0.60**

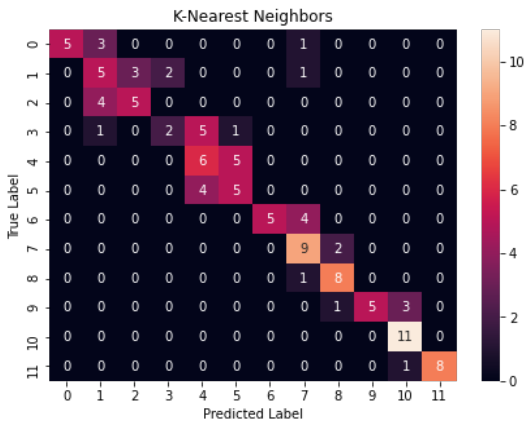
- **Decision Tree**

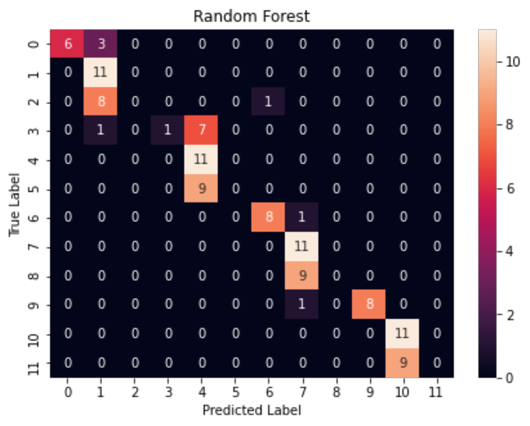
	Train Accuracy	Test Accuracy
k = 1	0.63	0.63
k = 2	0.68	0.66
k = 3	0.62	0.60
k = 4	0.68	0.65
k = 5	0.57	0.56
Mean	**0.64**	**0.62**

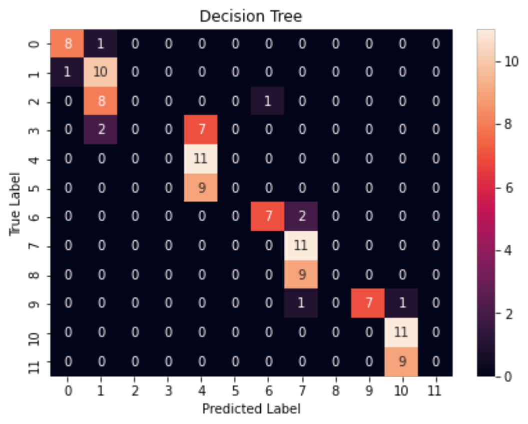
Three of the four selected classifiers exhibit similar test accuracy. The k-NN algorithm shows the highest accuracy of 64%, while SVC shows the lowest accuracy of 42%.

In order to visualize the classification results for each classifier, we present the corresponding confusion matrices. The numbers from 0 to 11 correspond to labels: “Low CO”, “Medium CO”, High CO”, “Low Ethanol”, “Medium Ethanol”, High Ethanol”, “Low Ethylene”, “Medium Ethylene”, High Ethylene”, “Low Methane”, “Medium Methane”, High Methane” respectively.









6 *Conclusion*

6.1 Discussion

In this work we tried to establish a common basis between the fundamental elements and operating principles of MOX gas sensors, and the computational methods that are being used for the development of an intelligent system capable of discriminating odors in complex sensing scenarios.

For this purpose we first constructed an experimental setup for the acquisition of the response signal of each one of the sensors in a gas sensor array. Due to some technical issues we were not able to construct the dataset we had originally planned. As a consequence the experimental dataset includes only recordings that are being characterized by the type of the gas under investigation, and not by its concentration. In order to examine the classification capabilities of the models we tested, on recordings that are being labeled by the gas type and also by its concentrations, we used two other datasets available at the UCI Machine Learning Repository.

Regarding the models we tested for the discrimination of volatile organic compounds and their binary mixtures, we utilized various machine learning algorithms and a pre-trained 2D convolutional neural network (VGG-16) used for image classification.

As the results indicate for the experimental dataset and dataset 2, VGG-16 performs well when classifying examples into categories defined by the type of the target-gas; whether it is a pure gas or a mixture. The accuracy achieved when we evaluated the network on a test set was 87.7% and 94.4% for datasets 1 & 2 respectively. A high accuracy for both classification tasks was somehow expected if we consider the low complexity of both datasets and the fact that VGG-16 is a sophisticated network, trained on a huge dataset and capable of classifying images into 1000 categories. Nevertheless, we should note that, despite the fact the the classification of the examples in dataset 1 is simpler than the one in dataset 2 (since we have fewer classes), the accuracy of VGG-16 in dataset 2 is higher. One explanation to this may be that the GSA of dataset 2 consists of eight sensors and thus, more discriminatory informations are available.

Dataset 3 was also used for the evaluation of VGG-16. The task now is more complicated since it consists of classifying instances into 12 categories defined by the type of target-gas (no mixtures) and the concentration in which each gas is detected. In order to evaluate the performance of VGG-16 to the given task, we used the original dataset and the augmented version of it. We also tried two different ways to illustrate the pictures of the GSA fingerprints. First, each picture illustrates the response of the GSA from the beginning to the end of the experiment. The obtained accuracy was 55.77%. Then, we tried to get rid of the useless information that a picture contains such as, the response during the second half of the experiment where no significant changes occur, and zoom-in to the transient response. The accuracy now is 63.46%. The increment to the accuracy of the model is significant in the case of the extended dataset, where we first achieved 67.1% accuracy and for the modified illustration of the pictures the obtained accuracy was 79.5 %. The impact of image modification and of data augmentation is clear in this case as we can see an increment to the accuracy of the model from 55.77% to 79.5%. Another way to further increase the accuracy could be a more efficient hyper-parameter tuning of the network. Since this can be achieved only by trial-and-error strategy, the implementation of this process could require a lot of time.

Finally we tested some machine learning algorithms to perform classification on dataset 3. This time the dataset was organized in a matrix similar to Eq. 3.7. We used four different classifiers and in order to select the right parameters for the models we used a brute-force strategy. Brute-force could require a lot of time to implement since it solves the classification problem for every possible combination of parameters for a finite parameter space. We tried to focus on the parameters that have the biggest impact on the final result. The highest accuracy achieved was 64% for k-Nearest Neighbors classifier which is higher than the ones of Decision Tree (DT) (62%), Random Forest (RF) (60%) and SVM (42%). The accuracy of RF and DT may be close with k-NN, but looking at the confusion matrices, we can see that the examples are gathered around the diagonal in the case of the k-NN. This is interpreted as the ability of k-NN to classify better the examples into concentration-related class, unlike RF and DT where they perform well only for the classification of examples to gas-related category.

At the end of the day, the choice of model depends on the application and the trade-off between the computational cost and the desired performance. As the results indicate, the neural network we put on the test has better performance (79.5% accuracy) for the classification task of dataset 3, than k-NN algorithm (64% accuracy). Of course the training of the neural network is a much more time-consuming process compared to the one of selecting the right parameters for k-NN.

6.2 Future work

Some thoughts about the future of the investigation, include the extension of the experimental dataset, by adding more sensors with different sensitivity characteristics, exposing them to more odors and their binary or ternary mixtures across a larger range of concentrations. We should design an experimental process that simulates the prevailing conditions of the working environment of the sensors as precise as possible. In this context we could add more features to the acquired data, such as, humidity, environmental temperature and pressure, and record the response of the GSA during the variations of those conditions. Another factor we should take under consideration is the fluid dynamics of the odor delivery system because the way a sensor detects the existence of an odor varies from one application to another.

In addition to the precise simulation of the environment of the sensors, we should take into account the effect of drift, that is, the gradual and unpredictable variation of the chemo-sensory signal responses when exposed to the same analyte under identical conditions, caused by aging (e.g. the reorganization of the sensor surface over long periods of time) and poisoning (e.g. irreversible binding due to external contamination) [9].

On top of the approaches for the construction of a representative dataset, we should also have in mind that a more extensive preprocessing stage may be necessary. This could include compressing and normalizing the acquired signal with local methods (operate across the sensor array on each individual fingerprint) and global methods (operate across the entire database for a single sensor), as well as reducing the dimensionality of the data utilizing a decomposition method such as the principal component analysis which is being widely used in smart gas sensing applications.

In this present work, we used the static descriptors (baseline and steady-state responses) for the characterization of the pattern of the GSA as well as the transient responses. It has been reported that the dynamic response to an odor carries a wealth of odor-discriminatory information that cannot always be captured with a single parameter [36]. Therefore, the use of the sensor transients as dynamic fingerprints has multiple advantages such as the improvement of selectivity by pattern-recognition means and reduction of the acquisition time (it is not necessary to reach the steady-state of the sensor) which yields to an increment of the sensor’s lifetime.

Regarding the pattern recognition methods, in this thesis we address a classification problem. In order to give a different approach to the analysis of volatile organic compounds and their mixtures, we could perform a regression analysis in which the final output is not a discrete class, but a value in a continuous domain such as the concentration of individual components in a mixture. In this context, along with other ML algorithms, we may use a 1-D convolutional neural network which will perform the convolution operation on 1-d data, such as time series, and not on 2-D images.

References

[1] E. L. Hines, E. Llobet, and J. W. Gardner, “Electronic noses: a review of signal processing techniques,” *IEE Proceeding online*, vol. 146, no. 6, pp. 297–310, 1999.

[2] J. W. Gardner, E. L. Hines, and C. Pang, “Detection of vapours and odours from a multisensor array using pattern recognition: Part 1. Principal Component and Cluster Analysis,” *Sensors Actuators B*, vol. 4, no. 4, pp. 109–115, 1991, doi: 10.1177/002029409602900603.

[3] J. W. Gardner, E. L. Hines, and H. C. Tang, “Detection of vapours and odours from a multisensor array using pattern-recognition techniques Part 2. Artificial neural networks,” *Sensors Actuators B. Chem.*, vol. 9, no. 1, pp. 9–15, Jul. 1992, doi: 10.1016/0925-4005(92)80187-3.

[4] M. Pardo and G. Sberveglieri, “Classification of electronic nose data with support vector machines,” *Sensors Actuators, B Chem.*, vol. 107, no. 2, pp. 730–737, 2005, doi: 10.1016/j.snb.2004.12.005.

[5] Y. Luo, W. Ye, X. Zhao, X. Pan, and Y. Cao, “Classification of Data from Electronic Nose Using Gradient Tree Boosting Algorithm,” 2017, doi: 10.3390/s17102376.

[6] V. Krivetskiy *et al.*, “Selective detection of individual gases and CO/H2 mixture at low concentrations in air by single semiconductor metal oxide sensors working in dynamic temperature mode,” *Sensors Actuators, B Chem.*, vol. 254, pp. 502–513, 2018, doi: 10.1016/j.snb.2017.07.100.

[7] J. Yang, Z. Sun, and Y. Chen, “Fault detection using the clustering-kNN rule for gas sensor arrays,” *Sensors (Switzerland)*, vol. 16, no. 12, pp. 1–21, 2016, doi: 10.3390/s16122069.

[8] S. Brahim-Belhouari, A. Bermak, M. Shi, and P. C. H. Chan, “Fast and Robust gas identification system using an integrated gas sensor technology and Gaussian mixture models,” *IEEE Sens. J.*, vol. 5, no. 6, pp. 1433–1444, 2005, doi: 10.1109/JSEN.2005.858926.

[9] A. Vergara, S. Vembu, T. Ayhan, M. A. Ryan, M. L. Homer, and R. Huerta, “Chemical gas sensor drift compensation using classifier ensembles,” *Sensors Actuators, B Chem.*, vol. 166–167, pp. 320–329, 2012, doi: 10.1016/j.snb.2012.01.074.

[10] X. Zhao, Z. Wen, X. Pan, W. Ye, and A. Bermak, “Mixture Gases Classification Based on Multi-Label One-Dimensional Deep Convolutional Neural Network,” *IEEE Access*, vol. 7, pp. 12630–12637, 2019, doi: 10.1109/ACCESS.2019.2892754.

[11] P. Peng, X. Zhao, X. Pan, and W. Ye, “Gas classification using deep convolutional neural networks,” *Sensors (Switzerland)*, vol. 18, no. 1, pp. 1–11, 2018, doi: 10.3390/s18010157.

[12] Z.-X. C. and C.-Y. Y. Chao-Lung Yang, “Sensor Classification Using Convolutional Neural Network by Encoding Multivariate Time Series as Two-Dimensional Colored Images,” *Sensors (Switzerland)*, no. 1, 2020.

[13] G. Wei, G. Li, J. Zhao, and A. He, “Development of a LeNet-5 gas identification CNN structure for electronic noses,” *Sensors (Switzerland)*, vol. 19, no. 1, 2019, doi: 10.3390/s19010217.

[14] Y. LeCun, L. Bottou, P. Haffner, and Y. Bengio, “Gradient-Based Learning Applied to Document Recognition,” *Proc. IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998, doi: 10.1016/j.bbrc.2005.03.111.

[15] L. Han, C. Yu, K. Xiao, and X. Zhao, “A new method of mixed gas identification based on a convolutional neural network for time series classification,” *Sensors (Switzerland)*, vol. 19, no. 9, 2019, doi: 10.3390/s19091960.

[16] B. Schilling, *Springer Handbook of Odor*. 2017.

[17] G. Eranna, *Gas Sensing Devices Metal Oxide*. 2012.

[18] J. W. Gardner, *Microsensors : principles and applications*. Chichester: Wiley, 1994.

[19] Julian W. Gardner, V. K. Varadan, and O. O. Awadelkarim, *Microsensors, MEMS, and Smart Devices*. .

[20] K. Arshak, E. Moore, G. M. Lyons, J. Harris, and S. Clifford, “A review of gas sensors employed in electronic nose applications,” *Sens. Rev.*, vol. 24, no. 2, pp. 181–198, 2004, doi: 10.1108/02602280410525977.

[21] J. Fonollosa, L. Fernández, A. Gutiérrez-Gálvez, R. Huerta, and S. Marco, “Calibration transfer and drift counteraction in chemical sensor arrays using Direct Standardization,” *Sensors Actuators, B Chem.*, vol. 236, pp. 1044–1053, 2016, doi: 10.1016/j.snb.2016.05.089.

[22] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. Cambridge, MA, USA: MIT Press, 2016.

[23] S. Haykin, *Neural Networks and Learning Machines*, vol. 3. 2008.

[24] V. Pedregosa, F. and Varoquaux, G. and Gramfort, A. and Michel, P. and Thirion, B. and Grisel, O. and Blondel, M. and Prettenhofer, and E. and Weiss, R. and Dubourg, V. and Vanderplas, J. and Passos, A. and Cournapeau, D. and Brucher, M. and Perrot, M. and Duchesnay, “Scikit-learn: Machine Learning in Python,” *J. Mach. Learn. Res.*, vol. 12, pp. 2825--2830, 2011.

[25] R. A. Devi and K. Nirmala, “Construction of Decision Tree : Attribute Selection Measures,” *Int. J. Adv. Res. Technol.*, vol. 2, no. 4, pp. 343–347, 2018, [Online]. Available: http://www.ijoart.org/docs/Construction-of-Decision-Tree--Attribute-Selection-Measures.pdf.

[26] K. Frank, *Hands-On Data Science and Python Machine Learning*. 2017.

[27] S. Nikolopoulos, I. Kalogeris, and V. Papadopoulos, “Non-intrusive surrogate modeling for parametrized time-dependent PDEs using convolutional autoencoders,” 2021, [Online]. Available: http://arxiv.org/abs/2101.05555.

[28] H. Sildir, E. Aydin, and T. Kavzoglu, “Design of feedforward neural networks in the classification of hyperspectral imagery using superstructural optimization,” *Remote*

Sens., vol. 12, no. 6, 2020, doi: 10.3390/rs12060956.

[29] A. Zhang, Z. C. Lipton, M. Li, and A. J. Smola, *Dive Into Deep Learning*. 2021.

[30] V. Dumoulin and F. Visin, “A guide to convolution arithmetic for deep learning,” pp. 1–31, 2018, [Online]. Available: http://arxiv.org/abs/1603.07285.

[31] K. Simonyan and A. Zisserman, “Very deep convolutional networks for large-scale image recognition,” *3rd Int. Conf. Learn. Represent. ICLR 2015 - Conf. Track Proc.*, pp. 1–14, 2015.

[32] O. Russakovsky *et al.*, “ImageNet Large Scale Visual Recognition Challenge,” *Int. J. Comput. Vis.*, vol. 115, no. 3, pp. 211–252, 2015, doi: 10.1007/s11263-015-0816-y.

[33] J. Fonollosa, I. Rodríguez-Luján, M. Trincavelli, A. Vergara, and R. Huerta, “Chemical discrimination in turbulent gas mixtures with MOX sensors validated by gas chromatography-mass spectrometry,” *Sensors (Switzerland)*, vol. 14, no. 10, pp. 19336–19353, 2014, doi: 10.3390/s141019336.

[34] T. T. Um *et al.*, “Data augmentation of wearable sensor data for Parkinson’s disease monitoring using convolutional neural networks,” *ICMI 2017 - Proc. 19th ACM Int. Conf. Multimodal Interact.*, vol. 2017-Janua, pp. 216–220, 2017, doi: 10.1145/3136755.3136817.

[35] M. and B. Abadi and M. and others Barham, Paul and Chen, Jianmin and Chen, Zhifeng and Davis, Andy and Dean, Jeffrey and Devin, Matthieu and Ghemawat, Sanjay and Irving, Geoffrey and Isard, “Tensorflow: A system for large-scale machine learning,” in *12th Symposium on Operating Systems Design and Implementation*, 2016, pp. 265--283.

[36] E. Llobet, J. Brezmes, X. Vilanova, J. E. Sueiras, and X. Correig, “Qualitative and quantitative analysis of volatile organic compounds using transient and steady-state responses of a thick-film tin oxide gas sensor array,” *Sensors Actuators, B Chem.*, vol. 41, no. 1–3, pp. 13–21, 1997, doi: 10.1016/S0925-4005(97)80272-9.