Machine Learning and Signal Processing Algorithms for a CPS with Chemical and Infrared Sensors

BY

DIAA H J BADAWI
M.Sc. in Electrical and Electronics Engineering, Bilkent University, Turkey
B.Sc. in Communications Engineering, An-Najah National University, Palestine

DISSERTATION

Submitted as partial fulfillment of the requirements
for the degree of Doctor of Philosophy in Electrical and Computer Engineering
in the Graduate College of the
University of Illinois at Chicago, 2022

Chicago, Illinois

Defense Committee:
Rashid Ansari, Chair
Ahmed Enis Cetin, Advisor
Pai-Yen Chen
Natasha Devroye
Erdem Koyuncu
Sule Ozev, Arizona State University

dedicated to my family

ACKNOWLEDGMENTS

I would like to thank the National Science Foundation NSF for providing funding for my research via Grant 1739396. I would like to thank Professor Sule Ozev of ASU and her team for providing us with sensor data that was used throughout this work. I would like to thank my advisor Professor Enis Cetin for his support and invaluable feedback throughout my doctoral journey.

DB

CONTRIBUTION OF AUTHORS

In the work "Computationally Efficient Spatio-Temporal Dynamic Texture Recognition for Volatile Organic Compound (VOC) Leakage Detection in Industrial Plants", Diaa Badawi and Ahmet Enis Cetin developed the algorithms. Diaa Badawi collected most of the data and processed the data and carried out most of the experiments. Hongyi Pan and Sinan Cem Cetin helped in collecting the data and carrying out parts of the experiments. Ahmet Enis Cetin was the supervisor of the work. Diaa Badawi wrote the paper, with help from Ahmet Enis Cetin.

In the work "Real-Time Low-Cost Drift Compensation for Chemical Sensors Using a Deep Neural Network with Hadamard Transform and Additive Layers", Diaa Badawi and Ahmet Enis Cetin developed the framework. Diaa Badawi carried out the experimentations. Agamyrat Agambayev collected part of the data used in the work. Ahmet Enis Cetin and Sule Ozev supervised the work. Diaa Badawi wrote most of the paper, with help from the other co-authors.

In the work "Detecting Anomaly in Chemical Sensors via Regularized Contrastive Learning", Diaa Badawi developed the idea. Ishaan Bassi collected part the data used in the work, under the supervision of Sule Ozev. Ahmet Enis Cetin supervised the work. Diaa Badawi wrote the paper, with help from Ahmet Enis Cetin.

TABLE OF CONTENTS

TABLE OF CONTENTS (Continued)

CHAPTER **PAGE**

TABLE OF CONTENTS (Continued)

CHAPTER **PAGE**

LIST OF TABLES

LIST OF TABLES (Continued)

LIST OF TABLES (Continued)

TABLE **PAGE**

LIST OF FIGURES

LIST OF FIGURES (Continued)

LIST OF FIGURES (Continued)

FIGURE **PAGE**

LIST OF FIGURES (Continued)

FIGURE **PAGE**

LIST OF FIGURES (Continued)

FIGURE **PAGE**

LIST OF ALGORITHMS

ALGORITHM **PAGE**

SUMMARY

In this dissertation, we present our deep learning-based solutions to crowd-sourced cyber-physical systems (CPS) with low-cost chemical sensors to detect ammonia and methane gas leakages. The first algorithm is a gas-leak detection method using uncalibrated sensors. The idea is to construct one-dimensional time-varying waveforms that represent the temporal history of sensor measurements at a location and feed the one-dimensional signals to a convolutional neural network. Our main observation is that while ammonia and other volatile organic compounds can be detected using chemical sensors, sensor measurements decrease over time due to deterioration in the sensor conditions known as sensor drift. We show that it is possible to detect gas leaks even with "below-the-threshold values" using data-driven machine learning algorithms. Furthermore, we show that the detection process can be improved by using the data from multiple sensors. We develop a multiplication-free neural network that is more suitable for energy-constrained devices than the vanilla network, and we show it can achieve good detection and drift correction accuracy. We also show that integrating Hadamard-transform-based layers into the deep learning structure achieves better results, thanks to the regularization effect of the transform. We also investigate a novel ℓ_1-inducing kernel metric for drift correction in a multiple-sensor system in unsupervised settings. Our results show that the kernel-based approach achieves better robustness than the baseline methods. The second algorithm aims to detect malfunctioning units in a sensor array system. The deep-learning algorithm aims at learning contrast between normal and abnormal sensors to achieve better sensitivity than

direct similarity comparison. We also devise a similarity score based on the aforementioned ℓ_1-inducing operator. Our results show that learning new representations via a contrastive learning scheme improves the ROC score from direct methods by 3.5%. Our third algorithm estimates the location of a gas leak source using sparse unreliable spatio-temporal chemical sensor data. This algorithm is based on deep learning and classical inverse problem methods. The neural network has a Fourier-domain layer that models the smoothness of the gas plume. In the transform domain, we project the feature map values onto a low-pass region whose boundary is determined during training using the backpropagation algorithm. This operation is equivalent to making a projection onto a convex set representing the smoothness of the data, and it is embedded into the non-linear structure of the convolutional neural network. We considered both VOC source leak detection and the ammonia vapor leak detection problems. In practice, we use the Discrete Cosine Transform (DCT) instead of the Fourier transform to take advantage of real arithmetic. Our method produces better results than the classical Papoulis-Gerchberg-based interpolation and Gaussian-mixture model-based interpolation methods.

CHAPTER 1

INTRODUCTION & OVERVIEW

We developed deep machine learning-based solutions to a crowd-sourced Cyber-Physical System (CPS) with low-cost sensors detecting leaking ammonia and methane gas plumes. These machine learning algorithms are (i) gas detection methods without using a system-level fixed threshold, (ii) anomalous sensor detection, and (ii) estimation of the gas leak source location using multiple sensors. Our machine learning algorithms are computationally efficient and can be used in edge devices.

(i) We developed Machine Learning (ML) based gas leakage detection algorithms using uncalibrated sensors. We tested the algorithm using ammonia and Volatile Organic Compounds (methane, ethane, butane, etc.). We collected our own Volatile Organic Compound (VOC) data and used data available on the Internet in our paper [3]. The sensor response to ammonia and VOC's are similar [4]. Our main idea is to construct a one-dimensional (1-D) time-varying waveform representing the temporal history of sensor measurements and feed the 1-D signals obtained from several sensors to 1-D convolutional neural networks. Generally speaking, sensors generate random values close to zero when there is no gas leak, but whenever ammonia and VOC chemicals are detected, the sensor values increase. The sensor values may fluctuate and decrease after some time in open air due to wind and chemical sensor drift. Nevertheless, gas leaks can be detected even when "below a fixed-threshold value" using a machine learning algorithm that was trained using similar sensor data.

Drift correction is a crucial pre-processing step for reliable and accurate gas analyte detection and identification in chemical sensors and Electronic nose (E-nose) systems [5]. Sensor drift causes the characteristics of a chemical sensor's response to change over time. Our Deep-Learning (DL) algorithm can estimate the baseline level of the sensor using an interpretable multiplication-free Hadamard transform-based layer. We apply data (feature-map) smoothing in the transform domain [6–8]. As a result, we showed that detecting gas molecules below system resolution is possible using uncalibrated sensors using DL. We also showed that the detection process is more reliable than a single sensor by taking advantage of a large number of deployed sensors [4]. The developed ML algorithms allow us to detect gas leaks at "below-the-threshold" levels because the server-based decision-making framework improves the accuracy significantly compared to a single sensor. We not only used a regular 1-D temporal convolutional neural network but an Additive neural Network (AddNet) [3, 4, 9]. AddNet uses the Hadamard transform and additive operators which adds the absolute values of two operands and determines the sign of the result as the sign of multiplication. The resulting dot-product "induces" the ℓ_1 norm. AddNet performed better than the binary neural network in our dataset. The AddNet allows us to implement the "below-the-threshold" gas leak detection algorithms at the edge in simple architectures which do not have the GPU processing capability. This approach when combined with server-based decision making will provide a very reliable solution for ammonia leak-based explosive detection in large-area public gatherings.

In summary, the proposed DL-based algorithm utilizes both the spatial correlation between the sensors and the temporal correlation between sensor measurements to make decisions, and

it can be used in other crowd-sourced CPS systems with low-cost sensors. In addition, the proposed ML-based 'below-the-threshold" detection scheme can be used in Infrared (IR) sensors and cameras as well [3]. VOC and ammonia plumes appear as dark clouds in IR cameras. In [3] we describe a two-stage deep neural network structure, taking advantage of both spatial and temporal structure of the dynamic texture regions created by the leaking VOC plume. We first detect moving pixels that are darker than their neighboring pixels. We extract one-dimensional (1-D) signals representing the temporal history of such pixels from video and feed the 1-D time-history waveforms to a 1-D convolutional neural network. If those pixels are near the edge of a VOC plume, their 1-D temporal signals exhibit high-frequency behavior and eventually they become dark pixels. A properly trained ML algorithm such as a neural network generates high probability estimates for such pixels. If the 1-D neural network generates high confidence values, the final decision is reached using a deep convolutional neural network (CNN) which processes image frames. The proposed algorithm uses both spatial and temporal correlations to reach the final decision. As a result, the system does not rely on (a) any fixed thresholds, and (b) it is more robust compared to an ML algorithm using only spatial information (similar to an image). The computational complexity of the system is also lower compared to a system using multiple image frames or the three-dimensional data cube for decision making.

(ii) Some of the low-cost sensors may malfunction due to sensor drift and chemical poisoning or they may simply be dead. We developed a method for detecting anomalous chemical sensors using a deep neural network using the contrastive learning-based framework [10]. In many practical systems including our CPS, an array of multiple chemical sensors is used. In standard

contrastive learning, the aim is to learn representations that will have maximum agreement among data samples of the same concept while having a minimal agreement with data samples from other concepts. We adapted standard contrastive learning to find useful representations for out-of-distribution sample detection. Furthermore, we compare the proposed framework with the cosine similarity measure and a novel similarity measure that we developed based on the ℓ_1 norm. The measure uses the operator that we use in the AddNet structure. Our experimental results show that our approach achieves higher AUC scores than baseline methods in our dataset. The low-cost ℓ_1 norm-based similarity measure can be used in systems requiring computational efficiency.

(iii) Our third algorithm estimates the location of a gas leak source using sparse unreliable spatio-temporal chemical sensor data. This algorithm is also based on deep learning and it has transform domain layers similar to our networks described in [3, 4, 6, 7, 9]. The neural network has a Fourier domain layer that models the smoothness of the gas plume. In addition, we have a novel projection-based operation for estimating the underlying gas leak signal and, henceforth, predicting the source location. In the transform domain, we project the feature map values onto a low-pass region whose boundary is determined during training using the backpropagation algorithm. This operation is equivalent to making a projection onto a convex set representing the smoothness of the data and it is embedded into the non-linear structure of the convolutional neural network. We considered both VOC source leak detection and the ammonia vapor leak detection problems. In practice, we use the Discrete Cosine Transform (DCT) instead of the Fourier transform to take advantage of real arithmetic. Our method produces better results

than the classical Papoulis-Gerchberg based interpolation and Gaussian-mixture model-based interpolation methods.

1.1 Specific Objectives

- We developed an adaptive chemical sensor drift estimation algorithm. The algorithm estimates the chemical sensor's baseline response. Our sensor drift estimation algorithm is computationally efficient and it can be an embedded part of the low-cost smart sensors. The estimation algorithm can be implemented in the server as well. The server can subtract the estimated baseline level from the sensor response to determine gas leaks that are "below-the-threshold" level.
- We developed anomalous sensor detection algorithms. One of the algorithms is based on deep learning. The second algorithm is based on classical principal component analysis (PCA) using a novel correlation operator which induces the L1 norm. Since the new correlation operator is a Mercer-type kernel. The method is a kernel-PCA method.
- We developed a deep-learning based method pinpointing the location of the explosive device producing the gas vapor from sparsely located sensor data. Our method combines a Fourier domain interpolation method with deep learning. We tested our algorithm using VOC methane gas data.

1.2 Significant Results

- We showed that using thresholds in chemical sensors may lead to incorrect detection results due to sensor poisoning. We observe the temporal waveform (time-varying signal)

that the sensor produces, and decide if the sensor is detecting ammonia or the VOC compounds including methane. The time-varying signal is very noisy in practice. We developed a neural network-based algorithm to monitor the temporal sensor waveform and the network makes the detection decision.

- We developed computationally efficient neural networks using binary operations. The neural networks can be used in other edge computing applications. The correlation operation that we use in these neural networks is related to the ℓ_1 norm.
- One of our computationally efficient correlation operations turns out to produce positive semi-definite covariance matrices. Therefore it is a Mercer-type kernel. We perform the vector product of two vectors as follows: $K(x,w) =< x,w >:= \sum_i \text{sign}(x_i w_i)\min(|x_i|,|w_i|)$ which turns out to induce the ℓ_1 norm: $< x,x >= ||x||_1$. This kernel is novel and it can be used in other applications [11].
- We introduced interpretable Fourier, DCT, and Hadamard transform domain layers to neural networks. In the transform domain, we impose a "band-limitedness" (or smoothness) constraint on the sensor signal. The gas vapor signal should have a smooth texture and be low-pass in nature. We observe this behavior in infrared videos of methane and ammonia leaks. We can impose this information on the feature maps of the neural network by using transform domain layers. We do not assume any specific bandwidth value as in classical Projections Onto Convex Sets (POCS)-based interpolation algorithms. We also denoise the feature maps in the transform domain using soft-thresholding (or smooth thresholding).

- Our gas source location estimation algorithm from sparsely located sensor data can be used in other applications. Source location detection problem is similar to reconstructing a physical field from sparse sensors or 3D scene reconstruction from point clouds.

1.3 Key outcomes or Other achievements

We developed a crowdsourced cyber-physical system made up of large numbers of ordinary people using small and low-cost sensors which respond to gas leaks. We developed ML algorithms using the temporal waveforms that the sensors produce. Our data is spatio-temporal in nature because the sensors are not in fixed locations. We showed that the source of a gas leak can be determined using this framework [3–8].

The proposed ML-based approaches can be also used in VOC (methane) leak, air pollution detection, and wildfire smoke detection [3, 4]. Since our algorithms use time-history of sensor values, they do not depend on fixed thresholds and the use of multiple sensors allows us to detect gas leaks below a system-level fixed threshold.

Our algorithms are computationally and energy-efficient. They are based on fast algorithms such as discrete cosine transform (DCT) and a Mercer-type kernel related to the ℓ_1 norm. Therefore they can be implemented at the edge in smartphones and/or in low-cost hardware.

1.4 Organization of This Thesis

in Chapter 2, we present our gas source leak localization algorithm in a cyber-physical system. In Chapter 3, we present our energy efficient DL system for VOC leak detection. In Chapter 4, we present our Hadmard-based drift correction system. In Chapter 5, we present our snsor anomaly detection system.

CHAPTER 2

GAS LEAK SOURCE LOCALIZATION USING SPARSE UNRELIABLE SENSOR DATA

Gas leak detection is crucial for environmental and security purposes. The rising levels of organic gases such as methane in the atmosphere are very concerning for the environment because of their strong greenhouse effect. This makes developing leak gas detection technology crucial to reducing emissions and, consequently, combating climate change. On another note, gases such as Ammonia (NH3) are by-products of improvised explosive devices (IEDs) that can wreak havoc on life and property. In light of this, detecting ammonia in these scenarios can immensely help in early identification of such dangers.

VOC gas detection technologies encompasses methods such as satellite imagery [12–14], commercially available bolometer-type IR cameras [15], spectrometry [16], and chemical olfactory technology [5, 17]. While these technologies vary vastly based on the application, chemical sensors are the cheapest and most flexible to use in cyber physical systems for indoor and outdoor environments. Chemical sensors can detect a wide array of gas analytes such as Ammonia, Methane, Acetone, Ethylene, Ethanol, and Toluene [18]. Nonetheless, one of the most important challenges facing the electronic nose technology is the sensor-to-sensor response variation due to aging and environmental factors such as, temperature and humidity, which is known as *sensor drift*. Furthermore, sensor calibration is costly and impractical.

Ammonia can be detected by low-cost Chemically Field Effect Transistors (ChemFET) [5, 19]. The low cost of this technology and its relative ease to connect to processing units have spurred interest in developing cyber-physical systems (CPS) for dangerous gas detection and source targeting [5]. In [5], A crowd-based CPS system for IED detection in public large events is envisioned. In their system, a large number of mobile sensory units are hosted by volunteers among the crowd. These units are attached to the participants' smartphones, and the data is sent to a central server to be analyzed. given the limited accuracy of the sensors, the CPS employs a two-level feedback logic.

2.0.1 Organization

The organization of this chapter is as follows: In Sec. 2.1 we provide a background on the problem of estimating gas leak source localization. In Sec. 2.2, we provide a background on bandlimited interpolation. In Sec. 2.3 and Sec. 2.4, we discuss the use of deep learning to solve inverse problems, and present our Deep Projection-onto-Convex-Sets (POCS) algorithms. In Sec. 2.5 we discuss the IR-based mathane leak data and its preprocessing, and we present our experimental results in Sec. 2.6. In Sec. 2.7 we discuss the Isopropyl alcohol data and its preprocessing, and present our experimental results in Sec. 2.8. Finally, we present our conclusion in Sec. 2.9

2.1 Gas Source Localization

While the inherent inaccuracies of sensor measurements is an issue in its own right, determining the source location is very challenging even with ideal sensory measurements. Many methods have been suggested to estimate the location of gas source leak indoor and outdoor.

Traditional approaches rely on expert-knowledge modeling and analyses. One of the most popular models is the Gaussian plume model [20, 21], thanks to its simplicity. In [20], the authors estimate the model parameters using different optimization algorithms and determine the source location accordingly. In [22] the authors maximize the likelihood function of a Gaussian plume model in an indoor setting and perform tracking by moving towards the predicted source location. In [23], the authors adopt a wave dispersion model to locate the gas source based on computational fluid dynamics (CFD). In that work, the authors argue that the downwind distance to the source is linearly correlated with the time it takes gas concentration to stabilize. Other approaches include the use of non-parametric models such as Gaussian processes regressions [24]. In [24], a bayesian prior of Gaussian processes is assumed and based upon the density construction, tracking is conducted using simulated annealing to balance between exploration and exploitation. In [25], the authors optimize a Eulerian simulator model given with the observed measurements of the sensors and the wind information as the initial conditions. The sensors are mounted on unmanned autonomous vehicles (UAVs), and the UAV are then flown to the location that maximizes a likelihood score.

While the previous body of work explores a wide array of approaches, Simulation-based models are complicated and require thorough domain-based fluid dynamics or meteorological knowledge. On the other hand, semi-analytical plume-based morphology models assume knowledge of the wind information and strong homogeneity of the wind field, which may not be the case in many real-world scenarios. Furthermore, selecting the best model is non-trivial and requires expert knowledge [26]. This motivates us to use deep learning to perform the task of

source locating, given sparse and noisy low-cost sensor measurements. For this purpose, we pose the problem as a spatio-temporal signal completion task.

Generally speaking, signal completion problem is an ill-posed inverse problem that. Without any prior assumptions, there exists an infinite number of solutions. Only a very small set of these solutions is meaningful. In this paper, we pose the problem as a generic image completion task. Unlike previous work, we do not incorporate any expert knowledge assumption but rather make use of natural assumptions on the underlying gas density signal. The main assumption we make is that the gas density signal is bandlimited.

2.2 Bandlimited Interpolation

The bandlimitedness assumption is wildly used in many signal processing applications such as image super-resolution [27], Direction-of-arrival (DoA) estimation [28], and medical imaging [29]. While signals cannot simultaneously be bandlimited and have finite support, a large class of natural finite-support signals has most of their energy concentrated in a finite bandwidth. This means that one can obtain a "good" bandlimited approximation of such signals.

In the classic image interpolation (completion) problem, one is presented with a subset of image pixel intensities, and the task is to find the missing points from the given point. The interpolated points should conform to a certain assumption made a priori. In our case, we use the bandlimited assumption. We pose the problem as a constrained optimization problem, i.e., finding a bandlimited two-dimensional signal whose pixel intensity at the given pixel locations is identical or as close as possible to the intensity of these given pixel values. This problem has been studied well over the past, and there exist many algorithms to solve it. One of the earliest

and most simple algorithms is Papoulis-Gerchberg (PG) algorithm [30] which falls under the more general framework of Alternating projection onto convex sets (POCS) [31–33].

2.2.1 Iterative Projections Onto Convex Sets

2.2.1.1 Background

Here, we provide some important mathematical facts on operator theory

Definition 1. *[35] Let (X,d) be a complete metric space. A mapping $T : X \mapsto X$ is called non-expansive if*

$$d(T(x), T(y)) \leq \alpha d(x, y), \quad \alpha \in [0, 1] \quad \forall x, y \in X$$

The mapping is called contractive if α is strictly less than one.

Definition 2. *Let $\mathcal{H}$ be a Hilbert space. A mapping $T : \mathcal{H} \mapsto \mathcal{H}$ is called firmly non-expansive [35] if*

$$||T(x) - T(y))||_2^2 + ||(I - T)(x) - (I - T)(y)||_2^2 \leq ||x - y||_2^2, \quad \forall x, y \in \mathcal{H}$$

where I is the identity mapping.

Examples of firmly nonexpansive operators include orthogonal projectors, soft-thresholding, and hard thresholding. Furthermore, we can obtain a firmly nonexpansive operator from an expansive operator T through the following transform.

$$Tx = \frac{(T + I)}{2} x \tag{2.1}$$

Example of nonexpansive but not firmly nonexpansive operators is $T = -I$.

Theorem 3. *(Contraction Mapping Theorem) [34] Let $C \neq \emptyset$ be a closed subset in a complete normed space $(X, ||.||)$. Let $f : C \mapsto C$ be a contractive operator. Then f has a unique fixed point in C.*

Theorem 4. *[31, 35] Let $\mathcal{H}$ be a Hilbert space. Let $T : \mathcal{H} \mapsto \mathcal{H}$ be firmly nonexpansive operator. Let Fix $T \neq \emptyset$. The iteration $x^{n+1} = Tx^n$ converges weakly to a point in Fix T.*

Theorem 5. *(Iterative Projections onto Affine Subspaces) [31, 35] Let $\mathcal{H}$ be a Hilbert space. Let $\{\mathcal{C}_1, \mathcal{C}_2, \ldots \mathcal{C}_\mathcal{N}\}$ be closed affine subspaces in $\mathcal{H}$. Let $\mathcal{C}_0 = \cap_{i=1}^{N} \mathcal{C}_i \neq \emptyset$. Let $P_i : \mathcal{H} \mapsto \mathcal{C}_i$ be the orthogonal projection operator onto subspace $\mathcal{C}_i$. The alternating projection:*

$$x^{n+1} := P_1 P_2 \ldots P_N x^n$$

converges strongly to $P_0 x$ for any $x \in \mathcal{H}$

Theorem 6. *(iterative Projections onto Convex Sets) [31, 35] Let $\mathcal{H}$ be a Hilbert space. Let $P_1, P_2, \ldots, P_N$ be a family of firmly non-expansive operators from D to D, where $D \subset \mathcal{H}$ is a closed and convex subset. Let Fix$(P_1 P_2 \ldots P_N) \neq \emptyset$. Then the iteration*

$$x^{n+1} = P_1 P_2 \ldots P_N x^n \tag{2.2}$$

converges weakly to a point $y \in$ Fix$(P_1 P_2 \ldots P_N)$.

2.2.1.2 Papoulis-Gerchberg (PG) Algorithm

In the PG algorithm, the problem is posed in a Hilbert-space framework and it is assumed that the discrete-space image $x_g[m,n]$ is bandlimited in the Fourier domain. This means that the signal lies in the intersection of two convex sets in $l^2(\mathbb{Z}^2)$.

$\mathcal{C}_1 := \{x \in l^2(\mathbb{Z}^2) |\ X(e^{j\omega_1}, e^{j\omega_2}) = 0,\ (\omega_1, \omega_2) \notin \text{BW}\}$, where $X(e^{j\omega_1}, e^{j\omega_2})$ is the Fourier transform, and $\mathcal{C}_2 = \{x \in \mathbb{R}^{M\times N} \mid x(m,n) = x_g(m,n) \text{ for } (m,n) \in S\}$. In this chapter $\mathcal{F}$ and $\mathcal{F}^{-1}$ denote the discrete-time Fourier transform (DTFT) and the inverse discrete-time Fourier transform (IDTFT), respectively. The bandwidth $\text{BW} \in [0, 2\pi] \times [0, 2\pi]$ is the bandwidth set defining the space of bandlimited signals. We use the upper case X to denote the DTFT of a signal x. The two sets $\mathcal{C}_1$ and $\mathcal{C}_2$ are known to be closed and convex sets in $\mathbb{R}^{N\times N}$. The PG algorithm is an iterative algorithm since it performs successive orthogonal projections onto $\mathcal{C}_1$ and $\mathcal{C}_2$. The orthogonal projection onto a set $\mathcal{C}$ is defined as

$$P_{\mathcal{C}}x = \inf_{z\in\mathcal{C}} ||x - z|| \tag{2.3}$$

The algorithm generates a sequence of iterates

$$x^{k+1} = P_1 \circ P_2(x^k), \quad k = \{0, 1, 2, \dots\}, \tag{2.4}$$

where P_1 and P_1 are the orthogonal projection operators onto $\mathcal{C}_1$ and $\mathcal{C}_2$, respectively. If the intersection $\mathcal{C}_1 \cap \mathcal{C}_2$ is non-empty, the iteration converges to a solution $x_c \in \mathcal{C}_1 \cap \mathcal{C}_2$. It is assumed

that the intersection is nonempty. Since the sets $\mathcal{C}_1$ and $\mathcal{C}_2$ are closed and convex, the projection operators are firmly nonexpansive operators and so is the composite operator $P_1 \circ P_2$ [31].

The projection operation onto the set $\mathcal{C}_1$ can be implemented in the Fourier domain. First, the DFT of x^k is computed and all the high frequency components of X^k are forced to zero. Then its inverse DFT is computed. The resulting image may have different values than $x_g[m, n]$ in in the spatial domain at locations $(m, n) \in S$. The projection onto $\mathcal{C}_2$ imposes the known image values such that $x_{k+1}(m, n) = x_g(m, n),\ \forall (m, n) \in S$.

2.2.2 Limitations of Bandlimited Based Interpolation

Despite its appealing concept, this alternating projection method has its limitations. First of all, the assumption of bandlimitedness does not hold in reality, given that bandlimited signals must have infinite support in the space domain. In reality, the spectrum of natural signals decay very quickly at high frequencies, so one should in principle find an effective "bandwidth" for good interpolation. Nevertheless, signals vary in their spectral energy distribution and finding an effective bandwidth parameter for each case is non-trivial. This makes the algorithm impractical for automated tasks. Another concern is that the bandlimited interpolation is optimal in mean-square-sense. This means that interpolating the signal via the aforementioned algorithm is likely to gloss over some fine details that are crucial in some applications, as in locating the gas source in our case. This motivates us to explore the idea of utilizing deep learning in a classical POCS procedure to overcome the aforementioned limitations.

2.3 Deep Learning Methods for Solving Inverse Problems

Deep Learning has shown remarkable success in computer vision and image processing tasks such as object detection, image-to-image translation and denoising, to name a few. Recently, there has been wide interest in applying deep learning to solve inverse problems in imaging [36]. While deep neural networks (DNNs) possess large generalization capabilities, the theoretical understanding of their behavior is still lacking. Furthermore, supervised learning, which is the hallmark of the performance superiority of DNNs, require large amount of data with ground truth, which may not be available. On the other hand, classical approaches are fully unsupervised since the prior assumptions do not depend on any data. Nevertheless, classical methods such as Majorizor Minimization (MM), POCS, gradient-descent-based Landweber iteration, and the more general proximal algorithms are computationally expensive because they are not single-shot forward models. This has motivated researchers to integrate deep learning in such algorithms to improve the quality of the solution and accelerate the convergence process. These methods include Learned Iterative Shrinkage Thresholding (Learned ITSA) algorithm [37], in which the authors learn transforms and thresholding parameters from the data using backpropagation and use them in a forward model that implements the traditional iterative shrinkage and thresholding algorithm for sparse inverse problems. Other methods include deep unrolling, in which one uses a neural network in an iterative inverse problem framework. This type of approach has become popular and has been used for many inverse problem tasks such as image denoising, deblurring, compressive sensing [38, 39].

2.4 Deep Learning Based Projection onto Convex Sets (POCS) Framework

In this section, we describe our deep-learning-based POCS framework in mathematical terms. We first describe our POCS-based approach, that is slightly different from the standard PG algorithm. We then state the deep-learning POCS framework, which we call *Deep DCT POCS*.

2.4.1 Discrete Cosine Transform POCS Procedure for Interpolating Gas Density Field from Sparse Data

Let $x \in \mathbb{R}^{N\times N}$ be the input measurements image of the gas density field [1]. Let $\mathbf{M}_{\mathrm{NZ}} \in \{0,1\}^{N\times N}$ be a logical mask such that $\mathbf{M}_{\mathrm{NZ}_{i,j}} = 1$ if there is a measurement exceeding a preset threshold at location (i,j) and zero otherwise. Let $\mathbf{M}_{\mathrm{Z}} \in \{0,1\}^{N\times N}$ be a logical mask such that $\mathbf{M}_{\mathrm{Z}ij} = 1$ if the reading at location (i,j) is below that same threshold and 0 otherwise. Let $\circ$ denote the Hadamard (element-wise) matrix product. One is presented with a sparse input map $x_g := x \circ \mathbf{M}_{\mathrm{NZ}}$. We wish to recover the missing points $x \circ (1 - \mathbf{M}_{\mathrm{NZ}})$. As stated earlier, one can obtain a "good" approximation of the image x in the subspace of bandlimited functions, which can be practically very useful. Furthermore, we have the additional constraint that the gas density is non-negative. Based on that, we can use alternating projection onto convex sets (POCS) to complete the missing data point with the following constraints: The reconstructed map $\hat{x}$ is in the intersection of the bandlimited subspace and the affine subspace satisfying the observations x_g.

---

[1] We assume that the spatial dimensions are the same and equal to N.

We define the "bandlimitedness" of a signal in the sense of discrete cosine transform (DCT) instead of Fourier transform to avoid dealing with complex-number arithmetic. Let $\mathcal{T}_{\mathrm{DCT}} : \mathbb{R}^{N\times N} \mapsto \mathbb{R}^{N\times N}$ be the two-dimensional type-II DCT. The two-dimensional DCT of an input x is given by [40]

$$X_{k,l} = \sum_{i=0}^{N-1}\sum_{j=0}^{N-1} x_{m,n} \cos\Big[\frac{\pi}{N}(m+\frac{1}{2})k\Big] \cos\Big[\frac{\pi}{N}(n+\frac{1}{2})l\Big] \tag{2.5}$$

We define projection onto the the bandlimited subspace by

$$[\mathbf{P}_{\mathrm{DCT}}X]_{ij} := \begin{cases} [X]_{ij}, & (i,j) \in \Lambda_{\mathrm{BW}} \\ 0, & \text{otherwise} \end{cases} \tag{2.6}$$

We take the set $\Lambda_{\mathrm{BW}} := \{(i,j) \in \{0,1,\dots N-1\}^2 \ | i+j \leq \mathrm{BW}\}$, for some integer bandwidth parameter BW.

We now state our alternating projection procedure; Let $\hat{x}_i$ and $\hat{X}_i$ be the interpolated image at step i and its respective DCT transform, our projection-based method is given by the following

$$\begin{aligned} X &\leftarrow \mathcal{T}_{\mathrm{DCT}}\hat{x}_i \\ Y &\leftarrow \mathbf{P}_{\mathrm{DCT}}X \\ y &\leftarrow \alpha\mathcal{T}_{\mathrm{DCT}}^{-1}Y \quad \text{(inverse DCT)} \\ y &\leftarrow \mathbf{M}_{\mathrm{NZ}} \circ x_g + (1 - \mathbf{M}_{\mathrm{NZ}}) \circ y \\ y &\leftarrow (1 - \mathbf{M}_{\mathrm{Z}}) \circ y \\ \hat{x}_{i+1} &\leftarrow (y)_+ \end{aligned} \tag{2.7}$$

where α is a normalizing scalar that is chosen to accelerate the convergence of the iterative algorithm, and the operator $(.)_+$ zeros-out the negative values. We select our α as follows

$$\alpha = \frac{< \mathbf{M}_{\mathrm{NZ}} \circ (\mathcal{T}^{-1}Y), x_g >}{||\mathbf{M}_{\mathrm{NZ}} \circ (\mathcal{T}^{-1}Y)||_2^2 + \epsilon}, \tag{2.8}$$

where $< ., . >$ denotes the dot product. This variant of the algorithm is known to accelerate the convergence of the POCS procedure. Furthermore, it was shown to converge to a solution provided that one exists [41]. Given an appropriate DCT-bandwidth parameter BW, one can obtain a globally smooth reconstruction of the original image x. Based on that, one can deduce the gas leak location from the reconstructed image.

2.4.2 DCT POCS for Binary Input

In addition to the algorithm prescribed in Equation 2.23, we also consider the case in which one is presented with logical sensor input measurements, indicating whether a sensor is reading a signal or not. In this case, we need to modify our settings to accommodate binary-input constraints. We consider that a sensor gives a reading of 1 if the gas density exceeds a threshold and zero otherwise. In this case, we do not directly enforce the spatial constraints by projecting onto the affine subspace as in Equation 2.23. Instead, we define two new constraint sets, which are

$$\mathcal{C}_2 := \{x_{i,j} \geq \text{Th} \mid \forall (i,j) \in \mathcal{S}_u\} \tag{2.9}$$

and

$$\mathcal{C}_3 := \{x_{i,j} < \text{Th} \mid \forall (i,j) \in \mathcal{S}_l\}, \tag{2.10}$$

where $\mathcal{S}_u$ denotes the set of the locations of the sensors whose measurement exceeds the pre-defined threshold Th, and $\mathcal{S}_l$ denotes the set of sensors whose measurements are below the threshold Th. Without loss of generality, assume our threshold is equal to 1. Apart from projecting onto the bandlimited subspace, we now need to perform projection onto the two convex sets $\mathcal{C}_2$, and $\mathcal{C}_3$. The two sets are polyhedral cones. Given a point $x \in \mathbb{R}^{N \times N}$, its projection onto the set $\mathcal{C}_2$ is given by

$$[P_{\mathcal{C}_2} x]_{ij} = \begin{cases} 1, & x_{ij} \leq 1 \ \& \ (i,j) \in \mathcal{S}_u \\ x_{ij} & \text{otherwise} \end{cases} \tag{2.11}$$

and

$$[P_{\mathcal{C}_3}x]_{ij} = \begin{cases} 1, & x_{ij} \geq 1 \ \& \ (i,j) \in \mathcal{S}_l \\ x_{ij} & \text{otherwise} \end{cases} \tag{2.12}$$

Given that the set $\mathcal{C}_2$ is unbounded, one may need to imposes a further constraint that the signal x be of finite energy. This can be achieved by projecting the signal x onto an ℓ_2 ball with radius E as follows

$$P_E x = \begin{cases} x, & ||x||_2 \leq E \\ E\frac{x}{||x||_2}, & \text{otherwise} \end{cases} \tag{2.13}$$

The DCT-based POCS procedure now becomes

$$\begin{aligned} \hat{X} &= \mathbf{P}_{\text{DCT}}\mathcal{T}x^k \\ \hat{y} &= \mathcal{T}^{-1}\hat{X} \\ \tilde{y} &= \alpha\hat{y} \\ x^{k+1} &= \mathbf{P}_{\mathcal{C}_3} \circ \mathbf{P}_{\mathcal{C}_2}\tilde{y} \end{aligned} \quad , \tag{2.14}$$

where α is an acceleration factor. Nevertheless, we cannot use the same formula for α as in Equation 2.8. This is because in previous case, we want to push the bandlimited signal toward the spatial constraints, while in this case, we want to either upper-bound or lower-bound the

signal at the given constraints locations. In order to do this, we use a very simple, closed-form expression for α that minimizes the following heuristic function

$$L(\alpha) := \sum_{(i,j)\in\mathcal{S}_u} \log(1+\exp(1-\alpha y_{ij})) + \sum_{(i,j)\in\mathcal{S}_l} \log(1+\exp(\alpha y_{ij}-1)) \tag{2.15}$$

The function $\log(1+\exp(y))$ is known as Softplus, and can be seen as a smoothed version of the hinge loss function $\max(x,0)$ [42]. In fact,

$$\text{Softplus}(y) := \log(1+\exp(y)) = \frac{1}{4}\max(x,0) * \text{sech}^2\Big(\frac{x}{2}\Big) = \frac{1}{4}\int_{-\infty}^{\infty} \max(y-t,0)\text{sech}^2\Big(\frac{t}{2}\Big)dt, \tag{2.16}$$

where $*$ denotes convolution, and $\text{sech}(x)$ is the hyperpolic secant function. The function is convex and twice differentiable and is an upper bound on the hinge loss $\max(y,0)$. The function L is convex in α. Therefore, we can perform a one-step update using Newton's method and we obtain the following expression

$$\alpha = 1 - \left\{\frac{\partial^2 L(\alpha)}{\partial \alpha^2}\Big|_{\alpha=1}\right\}^{-1} \frac{\partial L(\alpha)}{\partial \alpha}\Big|_{\alpha=1}, \tag{2.17}$$

where we set our "initial condition" to $\alpha = 1$. After evaluating the derivatives in Equation 2.17, we obtain the following expression

$$\alpha = 1 - \frac{\sum_{(i,j)\in\mathcal{S}_u} -y_{i,j}\sigma_u(y_{i,j}) + \sum_{(i,j)\in\mathcal{S}_l} y_{i,j}\sigma_l(y_{i,j})}{\sum_{(i,j)\in\mathcal{S}_u} {y_{i,j}}^2\sigma_u(y_{i,j})(1-\sigma_u(y_{i,j})) + \sum_{(i,j)\in\mathcal{S}_l} {y_{i,j}}^2\sigma_l(y_{i,j})(1-\sigma_l(y_{i,j}))}, \tag{2.18}$$

where $\sigma_u(y) = \sigma(1-y)$ and $\sigma_l(y) = \sigma(y-1)$, where $\sigma(x)$ is the sigmoid function, which is the derivative of the solftplus function. In this case, we have a simple, one-shot expression for α. Practically speaking, α is important in the early steps of the POCS algorithm, and it is especially helpful in accelerating the process of "filling up" the space initially, given the sparse constraints. Once the algorithm generates an interpolated image with appropriate energy, the importance of α lessens and, if the algorithm is converging to a solution in $\mathcal{C}_1 \cap \mathcal{C}_2 \cap \mathcal{C}_3$, α becomes very close to 1. In practice, we observed that the acceleration factor α converges to $\alpha = 1$ in very few iterations as shown in Figure 1.

2.4.3 Deep DCT POCS

In order to motivate the usage of deep learning, one can consider the following optimization problem

$$\text{minimize } \imath(x \in \mathcal{C}_1) + \imath(x \in \mathcal{C}_2) + \imath(x \geq 0) + f(x), \tag{2.19}$$

where $\imath$ is the $0 - \infty$ indicator function given by

$$\imath(x \in \mathcal{C}) = \begin{cases} 0, & x \in \mathcal{C} \\ \infty, & \text{otherwise} \end{cases} \tag{2.20}$$

and the sets $\mathcal{C}_1$ and $\mathcal{C}_2$ are the sets of bandlimited signal and the signals matching our observation x_g.

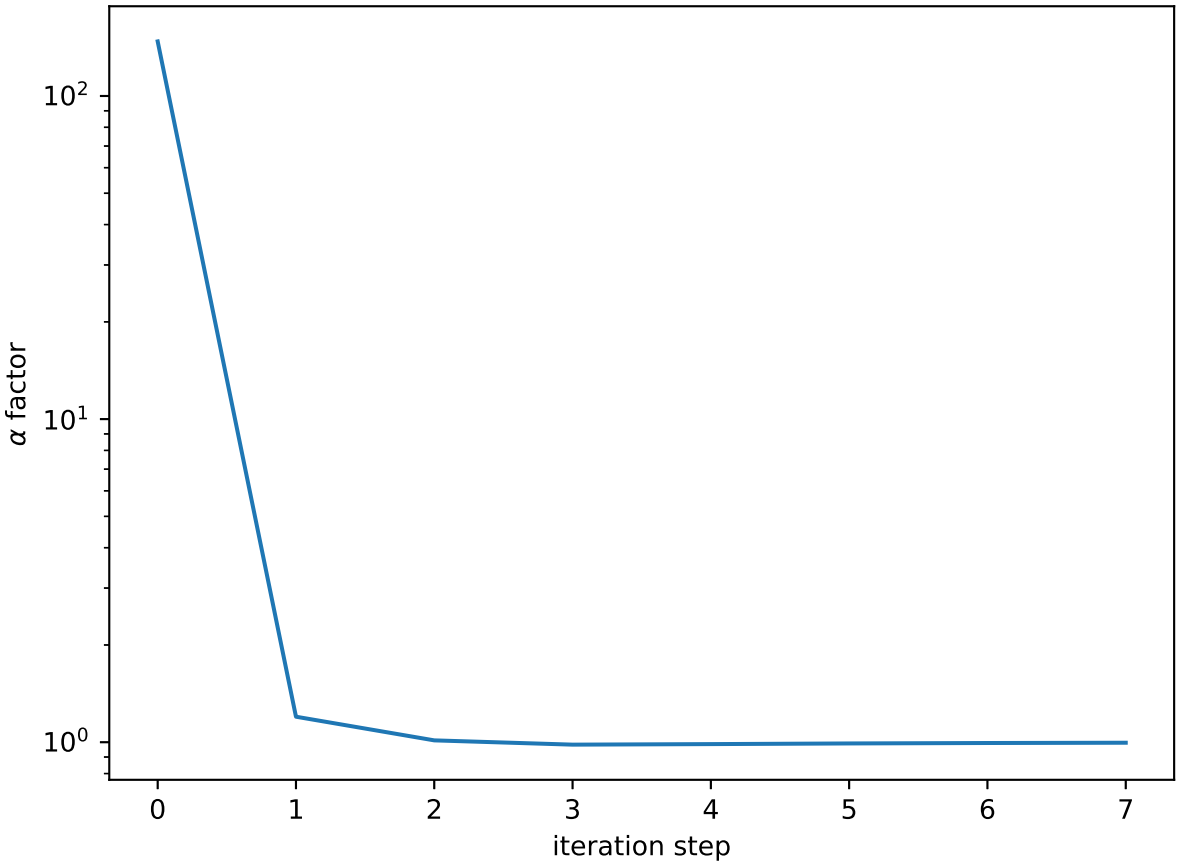


Figure 1. example of the convergence of the acceleration factor α to unity in the POCS procedure with binary constraints

In the case of binary constraints, the problem becomes

$$\text{minimize } \imath(x \in \mathcal{C}_1) + \imath(x \in \mathcal{C}_2) + \imath(x \in \mathcal{C}_3) + \imath(x \geq 0) + f(x), \tag{2.21}$$

where $\mathcal{C}_2$ is the constraint set of measurements exceeding a threshold, and $\mathcal{C}_3$ is the constraint set of measurements below a threshold. The function $f(x)$ is a sub-differentiable regularizer term. When $f(x)$ is absent, the problem reduces to a feasibility problem, and one can use the POCS procedure described in Sec. 2.4.1 to solve the problem. Instead, we want to learn a regularizing function from our data. The problem in Equation 2.19 can be solved using the following procedure

$$\begin{aligned} y &= \text{POCS}(x^i) \\ x^{i+1} &= \text{Prox}_{\lambda f}(y), \end{aligned} \tag{2.22}$$

where POCS is the composite operator of all the steps in Equation 2.7 and $\text{Prox}_{\lambda f}(y) := \underset{z}{\text{argmin}}(\frac{1}{2}||y - z||_2^2 + \lambda f(z))$ is the proximal operator of the regularizing function $f(.)$ with hyper-parameter $\lambda > 0$. Instead of directly learning a regularizing function $f(x)$, we replace the second step in the procedure in Equation 2.22 with a function $\text{DNN}(y)$. This function is implemented by an optimized deep neural network. The procedure now becomes

$$\begin{aligned} y &= \text{POCS}(x^i) \\ x^{i+1} &= \text{DNN}(y) \end{aligned} \tag{2.23}$$

Therefore the neural network can now be interpreted as the proximal operator of a regularizing function $f(.)$ learned implicitly when the deep neural network is optimized.

2.4.3.1 Deep DCT POCS with Unknown Bandwidth

As mentioned earlier, in the POCs procedure one needs to set up the bandwidth constraints in advance. The issue with this is that the bandwidth constraints need to be set for each inverse problem separately. Furthermore, different bandwidths constraints may result in drastically different solutions to the inverse problem. To overcome this limitation, we propose the use of different bandwidth constraints in parallel and let the deep neural network fuse the different solutions and decide on which one fits the observed data. We modify our procedure in Equation 2.23 to become as in Algorithm 1.

where $\text{POCS}(y;\text{BW})$ is the POCS procedure with input y and bandwidth parametrization BW, `concat`(...) is the concatenation of B $N \times N$ gray-scale images into a single 3D tensor $\in \mathbb{R}^{N\times N\times B}$, where B is the number of bandwidth parameters. z_l^i is the output of the iterate i with bandwidth parameters BW_l. The neural network $\text{DNN} : \mathbb{R}^{N\times N\times B} \mapsto \mathbb{R}^{N\times N}$. Note that the intermediate outputs y_l^{i+1} are "damped" version of the output of the neural network. These residual connections help stabilizing the forward procedure and make training the neural network easier via backpropagation as explained in the next subsection. Note that the number of parallel passes is set to 4 in this but it can be done for any number of bandwidth parameters selection.

Algorithm 1: Pseudocode for the deep regularizer DCT-POCS procedure. The algorithm takes the sparse input x_g and the two logical masks $\mathbf{M}_{\mathrm{NZ}}$ and $\mathbf{M}_{\mathrm{NZ}}$, the number of unrolling steps is L, and the number of unrolling steps for the POCS algorithm is K. The algorithm returns the output z^L that represents the completed (interpolated) image data. The subroutine $\texttt{POCS}(y_j^i, x_g, \mathbf{M}_{\mathrm{NZ}}, \mathbf{M}_{\mathrm{Z}}, \mathrm{BW}_j, K)$ implements Equation 2.7 with initial condition y_j^i for K steps. The set $\mathcal{BW}$ has B different bandwidth parameters

Input: $x_g, \mathbf{M}_{\mathrm{NZ}}, \mathbf{M}_{\mathrm{Z}}, L$
$\hat{x}^0 \leftarrow x_g$
$y_j^0 \leftarrow x_g$ (j=1 to B)
for *i=1 to L* **do**
  for *j=1 to B* **do**
    $y_j^i \leftarrow \beta y_j^{i-1} + (1-\beta)x^{i-1}$
    $z_j^i \leftarrow \texttt{POCS}(y_j^i, x_g, \mathbf{M}_{\mathrm{NZ}}, \mathbf{M}_{\mathrm{Z}}, \mathrm{BW}_j, K)$
  end
  $Z^i \leftarrow \texttt{Concat}(z_1^i, z_2^i, \ldots, z_B^i)$
  $x^i \leftarrow \texttt{DNN}(Z^i)$
end
return x^L

Likewise, we apply the same approach to combining the other variant of our algorithm that deals with binary constraints with deep neural network. The pseudocode is provided in Algorithm 2. A visual demonstration of the algorithm is shown in Figure 2.

Algorithm 2: Pseudocode for the deep regularizer DCT-POCS procedure with binary constraints. The algorithm takes the two input masks $\mathcal{S}_u$ and $\mathcal{S}_i$ and the two .The number of unrolling steps is L, and the number of unrolling steps for the POCS algorithm is K. The algorithm returns the output z^L that represents the completed (interpolated) image data. The subroutine $\texttt{POCS}(y_j^i, \mathcal{S}_u, \mathcal{S}_l, K)$ implements Equation 2.7 with initial condition y_j^i for K steps. The initial condition for the entire procedure x^{init} is an $N \times N$ map with 1 if $(i,j) \in \mathcal{S}_u$, and zero otherwise. The set $\mathcal{BW}$ has B bandwidth parameters

Input: $\mathcal{S}_u, \mathcal{S}_l$, x^{init}, L
$\hat{x}^0 \leftarrow x^{\text{init}}$
$y_j^0 \leftarrow x^{\text{init}}$ (j=1 to B)
for *i=1 to L* **do**
  for *j=1 to B* **do**
    $y_j^i \leftarrow \beta y_j^{i-1} + (1-\beta) x^{i-1}$
    $z_j^i \leftarrow \texttt{POCS}(y_j^i, \mathcal{S}_u, \mathcal{S}_l, K)$
  end
  $Z^i \leftarrow \texttt{Concat}(z_1^i, z_2^i, \ldots, z_B^i)$
  $x^i \leftarrow \texttt{DNN}(Z^i)$
end
return x^L

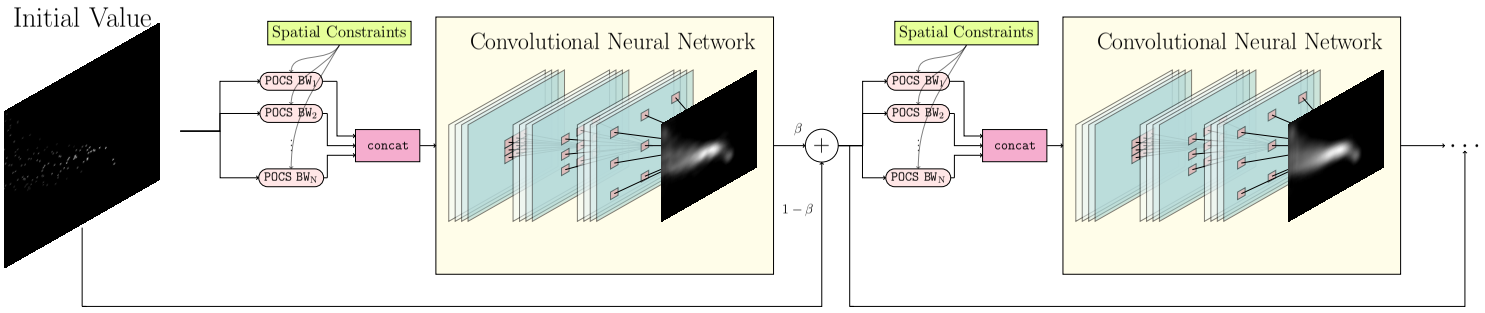


Figure 2. Illustration of the Deep DCT POCS Architecture with two steps unrolled

2.4.3.2 Training Deep DCT POCS

As mentioned earlier, DNN is to be trained to serve as an optimizer of a data-learned regularizing function. While earlier methods such as Plug-and-Play (PnP) suggest placing a pre-trained model into an optimization procedure, this approach has its shortcomings. First, neural networks are highly non-linear functions with difficult-to-control dynamic behavior. While many have attempted to regularize the weights of a neural network such that it serves as a non-expansive operator by design, layer-wise constraints seem very restrictive. Secondly- and perhaps more importantly, is the nature of our task; we are given an input of very sparse observations, and we wish to fill out the missing data. This means that a neural network trained to complete the field image in a single forward pass might generate unexpected results on the second forward pass due to the difference in the energy of the input. These concerns are alleviated by "unrolling" the iterative procedure for a few steps and training the neural network in these settings. In our case, we unroll the procedure expressed in Algorithm 1 for five steps and optimize the final result in a supervised fashion. Our loss function is

$$J(\hat{x}) := ||x - \hat{x}||_{W}^{2} - \lambda \text{SSIM}(x, \hat{x}), \tag{2.24}$$

where x is the ground truth gas density image, and $\hat{x}$ is the interpolated image. W is a weighing mask for the mean-square error (MSE). SSIM is the similarity index measure. The reason for optimizing a weighted MSE, as opposed to uniform MSE, is that we want to put more emphasis

on the region surrounding the source of the gas leak since our ultimate task is to identify the source location. given the source location at (i^*, j^*), the mask W is

$$W(i,j) = 1 + B \exp\left(- \frac{(i-i^*)^2 + (j-j^*)^2}{2\sigma^2} \right) \tag{2.25}$$

where (i^*, j^*) are the coordinates of the source. We select $B = 20$ and $\sigma = 7$. The second term SSIM in Equation 2.24 refers to the structural similarity index metric [43]. The SSIM between two image patches x and y is given by

$$\text{SSIM}(x,y) = \frac{(2\mu_x\mu_y + c_1)(2\sigma_{xy} + c_2)}{(\mu_x^2 + \mu_y^2 + c_1)(\sigma_x^2 + \sigma_y^2 + c_2)}, \tag{2.26}$$

where μ_x and μ_y are the means, σ_x^2 and σ_y^2 are the variances, and σ_{xy} is the cross correlation, and c_1 and c_2 are small constants. The hyperparameter $\lambda = 10^{-3}$. As for the training the binary-constraint algorithm, i.e., Algorithm 2, we use a slightly different loss function as follows

$$J(\hat{x}) := ||(1+W) \circ \frac{x}{\text{Th}} - \hat{x}||_2^2 - \lambda \text{SSIM}((1+W) \circ \frac{x}{\text{Th}}, \hat{x}), \tag{2.27}$$

where Th is the preset threshold used to generate the masks $\mathcal{S}_u$ and $\mathcal{S}_l$, and W is a wight mask given by Equation 2.25. In this case, we artificially make the ground-truth signal larger closer to the source, in order to guarantee the generate image has a peak closer to the source. In this case we set $B = 5$.

2.4.4 Network Architecture

In our framework, we used a convolutional neural network that takes a multi-channel $N \times N$ input and returns a single $N \times N$ output image. The convolutional neural network consists of 7 convolutional layers. Each of the convolutional layers outputs a 64-channel output, except for the last layer, which outputs a one-channel output image. The convolutional layers implement the dilated 2d convolution (correlation) operation [44], which is known to achieve better results for multi-scale recognition tasks. In our case, the use of dilated convolution is to increase the effective size of the receptive fields of the filters. Therefore, enabling feature smoothing at different scales. Furthermore, we noticed that using dilated convolution instead of a down-sampling-up-sampling pipeline avoids creating unwanted artifacts. The architecture of our convolutional neural network is detailed in Algorithm I. It is worth mentioning that, expect for the last convolutional layer, we use leaky Rectified Linear Units (Leaky ReLU) in our neural network. This is to avoid the issue of dying units, i.e., units that generate a zero output.

Layer	Num Channels	Dilation Rate	Activation
Input ($128 \times 128 \times 4$)			
Conv 1	64	1	Leaky Relu (0.2)
Conv 2	64	2	Leaky Relu (0.2)
Conv 3	64	4	Leaky Relu (0.2)
Conv 4	64	8	Leaky Relu (0.2)
Conv 5	64	4	Leaky Relu (0.2)
Conv 6	64	2	Leaky Relu (0.2)
Conv 7	1	1	Relu
Output ($128 \times 128 \times 1$)			

TABLE I

THE ARCHITECTURE OF THE DEEP REGULARIZED NETWORK. THE FILTER SIZE IS 3×3 IN THE LAYERS EXCEPT THE FIRST AND LAST (CONV 1 AND CONV 7), WHERE IT IS SET TO 5×5.

2.5 Methane Leak Detection

As discussed earlier, we are interested in locating a gas source given sparse unreliable chemical sensor data. Due to the difficulty of collecting such data, we used IR-imaged video data of controlled methane leaks collected by [21] in our analysis. The data set corresponds to high-quality IR-imaged videos of controlled methane gas leaks from industrial plants. While our original task is to do gas source localization from sparse chemical sensors, IR based sensing can be used for detecting volatile organic compounds (VOC) such as methane and propane. This is because methane has high absorbance in the long-wave infrared (LWIR) spectrum.

2.5.1 Pre-processing IR Imaged Data

In order to pose the problem as a source identification from sparsely located IR sensor data, we first pre-process the videos by removing the background from the scene. This was done by

estimating the optical flow using Kanade-Lucas algorithm [45]. We segmented the foreground from the background based on whether the magnitude of the optical flow field exceeds a certain threshold. Given that the camera is stationary in these videos, we estimated the background by taking the average over the different background estimates. Let a video have T frames. Let $v[i, j, t]$ be the pixel intensity value at location (i, j, t). Let the background pixels be $\mathcal{BG}$. The estimated background pixel intensity value at location (i,j) is given by

$$b[i, j] = \frac{\sum_{t=1}^{T} v[i, j, t] I((i, j, t) \in \mathcal{BG})}{\sum_{t=1}^{T} I((i, j, t) \in \mathcal{BG})} \quad \text{when} \quad \sum_{t=1}^{T} I((i, j, t) \in \mathcal{BG}) \neq 0 \tag{2.28}$$

As it can be seen from Equation 2.28, we cannot estimate the background at all locations. This is true near the source because it is always detected as foreground. For this reason, we need to complete the background in those regions. We do so by using the PG algorithm. This is because the background is mostly flat (low frequency). Afterward, we manually smooth out any artifacts and obtain a background estimate. The procedure and an example background reconstruction are shown in Figure 3.

2.5.2 Modeling Sparse Unreliable Sensor Data from Pixel Intensity Values

To make the data match low-cost IR sensory data, we model the sensor response from the pixel intensity values to reflect the limitations of their sensing capabilities. To do this, we investigate two scenarios: In the first case, let $v(m, n, t) \in [0, 1]$ for $t \in \{1, \ldots T\}$ be the

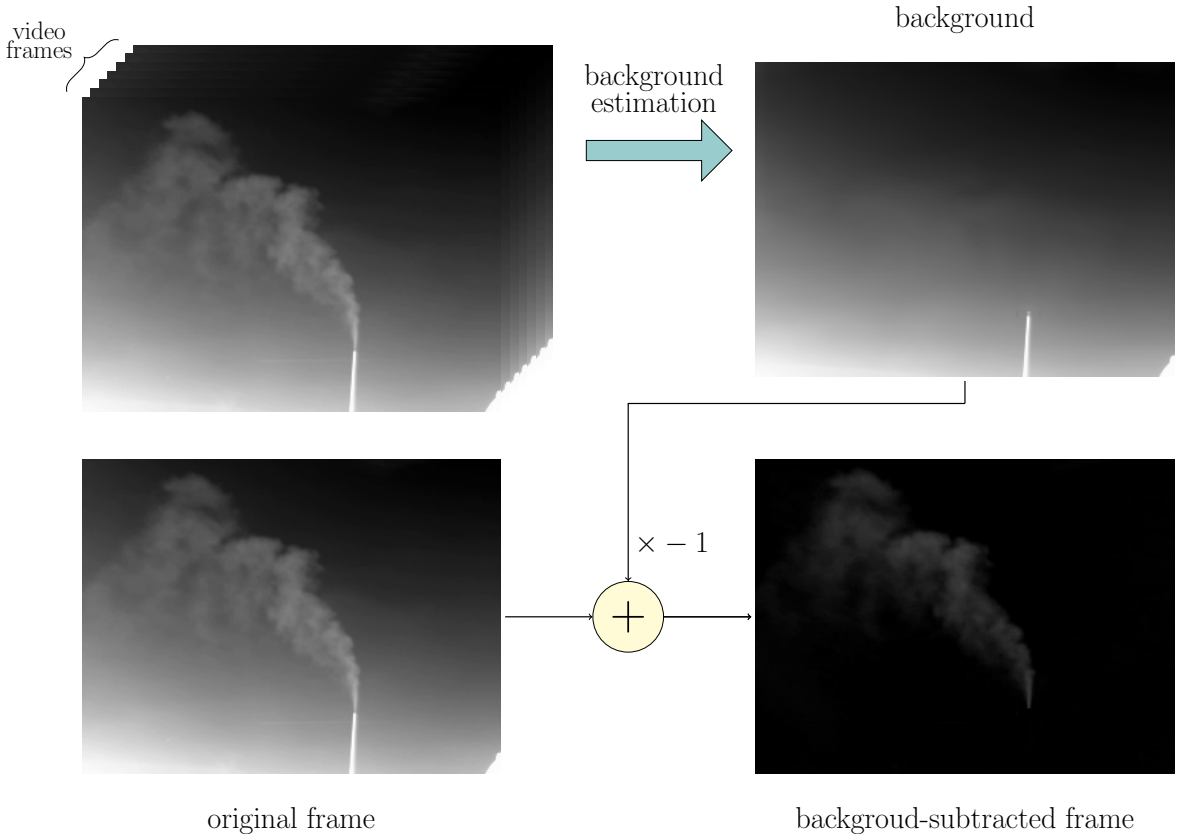


Figure 3. illustration of the background subtraction process

(background-subtracted) pixel intensity value at location m, n at time step t. We use the following relation to model a low-cost sensor response

$$x(m,n) = \frac{1}{T}\sum_{t=1}^{T} I(v(m,n,t) \geq Th), \tag{2.29}$$

where I is the 0-1 indicator function, Th is a certain threshold. Equation 2.29 accounts for the low accuracy and the low temporal resolution of an IR sensor.

We then sample a relatively small number of data points $\{x_g \mid s \in \mathcal{S}\}$ from the image x such that $|\mathcal{S}| << N \times N$, where $N \times N$ is the size of image x_g. We then feed these sparse images, along with constraints masks to our deep framework to try and retrieve the full image x. In our experiments, we set $Th = 16/255$ and $T = 16$.

In the second case, we consider more extreme limitations of the sensor output; we model the sensor response as a binary variable as follows

$$x(m,n) = I\big(\frac{1}{T}\sum_{t=1}^{T} v(m,n,t) \geq Th\big). \tag{2.30}$$

2.6 Experimental Results for Methane IR Methane Data

2.6.1 Results with Real-Valued Input Measurements

We trained our neural network by unrolling a few steps of the forward model in Algorithm 1. We used 4 DCT bandwidth parameters $\mathcal{BW} = \{4, 8, 16, 32\}$. Each inference pass is 4 iterations ($L = 4$ in Algorithm 1) with the POCS procedure applied 4 times ($K = 4$ in Algorithm 1). Our dampening factor β is set to 0.5. We used the same parameters for training and inference. The

final output $\hat{x}$ is then optimized to minimize the loss function in Equation 2.24. We trained the neural network over the frames of one video for 10 epochs. We construct our inputs from the pixel intensity values according to Equation 2.29 our averaging window size is 16 frames, and our threshold Th is set to 10/255. This means the values of $x_g(i,j) \in \{\frac{0}{16}, \frac{1}{16}, \dots \frac{16}{16}\}$.

2.6.1.1 Source Localization Results

We assess the goodness of the output of our Deep-POCS network, i.e. the interpolated image, by considering how close the maxima with the largest amplitude to the source. In order to do this, we first detect all local maxima of the output of the neural network. Let $\hat{x}$ be the output of the framework. We first apply a maximum filter to $\hat{x}$

$$\hat{x}_{i,j}^{\max} = \max\{\hat{x}_{k,l} \mid (k,l) \in \texttt{NBR}(i,j)\}, \tag{2.31}$$

where the neighborhood $\texttt{NBR}(i,j)$ is defined as the 7×7 grid surrounding point (i,j). We then find all the local maxima $\mathcal{S} := \{(i,j) \mid \hat{x}_{i,j} = \hat{x}_{i,j}^{\max}\}$. We then select the maxima with the largest k values and construct our set $\mathcal{S}_k$. In our case, we take the top 5 points. Let the source be located at (p,q), We then define our distance-to-source metric as follows

$$\text{dist}_{\min} := \min(\{\text{dist}\big((p,q),(m,n)\big) \mid (m,n) \in \mathcal{S}_k\}, \tag{2.32}$$

where $\{\text{dist}\big((p,q),(m,n)\big)$ is the euclidean distance between the points (p,q), and (m,n) In addition to that, we perform another pass to the deep NN while removing the constraints

around the maxima points in $\mathcal{S}_k$. Mathematically, we modify our non-zero mask $\mathbf{M}_{\text{NZ}}$ as follows

$$[\mathbf{M}_{\text{NZ}}{}^{\text{new}}]_{i,j} = \begin{cases} 0, & (i,j) \in \bigcup_{s \in \mathcal{S}_k} \texttt{NBR}(s) \\ [\mathbf{M}_{\text{NZ}}]_{i,j}, & \text{otherwise} \end{cases}, \tag{2.33}$$

where $\texttt{NBR}(s)$ is the neighborhood of the extreme point s. In our case, for each of 5 extreme points in $\mathcal{S}_k$ we eliminate the constraints contained in their 7×7 grid neighborhood. Once we obtain the eroded mask $\mathbf{M}_{\text{NZ}}{}^{\text{new}}$, multiply it with the sparse input $x_g^{\text{new}} := x \circ \mathbf{M}_{\text{NZ}}{}^{\text{new}}$ and feed the new input to the deep model again and obtain a new interpolated output. We then detect the maxima points and compare their distances to the source location

Vid ID	Maxima Distance to Source				
	NN (one step)	NN(two step)	GMM	POCS DCT BW=16	POCS DCT BW=32
Exp 1	22.4	15.8	19.7	27.8	29.4
Exp 2	19.1	13.3	20.1	26.6	35.1
Exp 3	11.7	10.1	11.4	17.3	10.1
Exp 4	21.3	18.9	31.9	42.1	48.4
Exp 5	15.7	9.6	12.0	18.5	17.9
Exp 6	28.1	17.9	26.5	33.3	34.7
Exp 7	12.8	8.9	12.0	17.6	11.5
Avg.	**18.7**	**13.5**	19.1	26.2	26.7

TABLE II

RESULTS WITH 480 SENSOR MEASUREMENTS FOR THE DEEP REGULARIZED POCS COMPARED TO GMM FITTING AND TRADITIONAL POCS.

Vid ID	Maxima Distance to Source				
	NN (one step)	NN(two step)	GMM	POCS DCT BW=16	POCS DCT BW=32
Exp 1	25.1	22.5	26.2	27.7	31.4
Exp 2	19.1	14.1	22.0	27.4	29.6
Exp 3	19.9	17.1	15.3	20.4	16.6
Exp 4	27.1	24.2	36.7	43.4	44.5
Exp 5	18.1	15.9	14.3	18.7	19.7
Exp 6	28.5	20.1	25.8	31.4	35.5
Exp 7	18.7	15.2	14.1	18.4	18.6
Avg.	**22.4**	**18.4**	**22.1**	26.8	28.0

TABLE III

RESULTS WITH 100 SENSOR MEASUREMENTS FOR THE DEEP REGULARIZED POCS COMPARED TO GMM FITTING AND TRADITIONAL POCS.

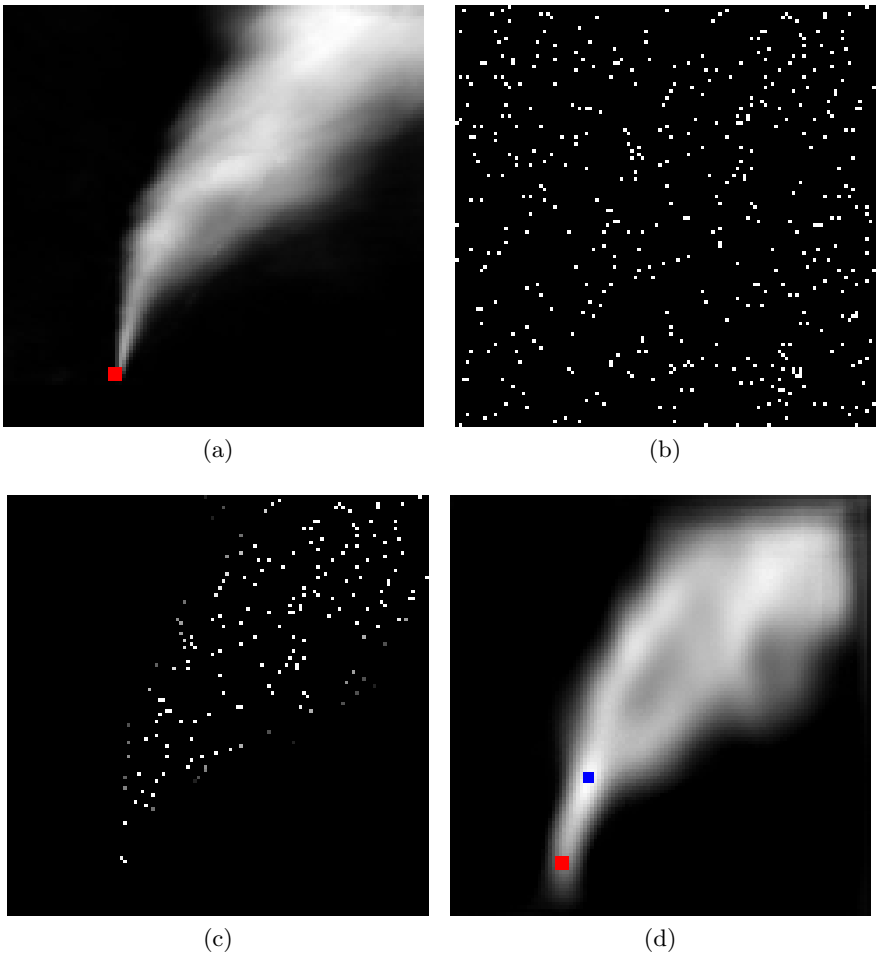
We also compared our Deep-DCT-POCS results with a Gaussian-mixture model (GMM). In this case, we start off by selecting a number of mixtures M. We then repeat the location pair (x, y) of the sparse measurements based on the intensity of measurement at that location. In our case, since the measurements correspond to the count the pixel intensity exceeds the threshold $Th = 10/255$ in a total of 16 consecutive, these counts range from 0 to 16. Therefore, we duplicate these points in our fitting data sets accordingly. For example if the intensity at location (x_0, y_0) is 3/16 and the intensity at location (x_1, y_1) is 10/16, we include the location pair (x_1, y_1) in our fitting data points 10 times, while repeating the location (x_0, y_0) three times. Notice that if the value of $x_g(m, n)$ is 0, we then do not include that point in GMM fitting

process. Once we have constructed our location-tuple dataset according to their intensities, we end up with a $d \times 2$ dataset $\mathcal{D}$, where d is the final size after repetition and now we try to find the two dimensional mean vectors μ_i, the 2×2 co-variance matrices C_i and the weights of each Gaussian in the mixture model π by finding maximum likelihood location

$$\{\pi_i^*, \mu_i^*, C_i^*\}_{i=0}^{M-1} = \text{argmax} \sum_{i=0}^{M-1} \pi_i \mathcal{N}(\mu^i, C_i|\mathcal{D}), \tag{2.34}$$

where $\mathcal{N}$ is a two dimensional Gaussian distribution. We solve Equation 2.34 using the expectation-maximization algorithm. Once the algorithm converges, we then take the locations of the means μ_i as our candidate maxima locations and construct our set $\mathcal{S}_k$ and find the score dist_{min} defined in Equation 2.32. Finally we compare the results with the traditional POCS algorithm as explained in Sec. 2.4.1. Our results are summarized in Algorithm II for inputs with 480 input measurements and in Algorithm III for inputs with 100 sensor measurements.

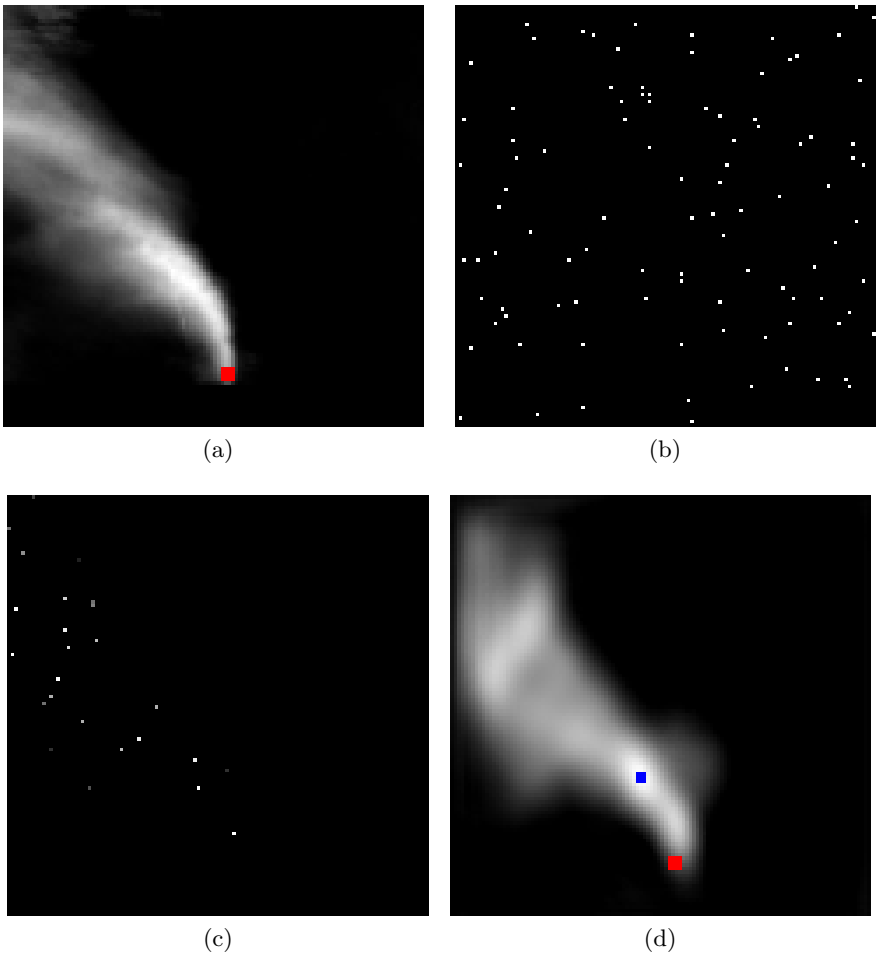
As one can see from Algorithm II and Algorithm III, the Deep DCT-POCS approach provides better score in most cases than in both the GMM fitting and traditional POCS approaches. Furthermore, while the GMM fitting gives distances better that the Deep DCT-POCS in some examples, it is significantly worse that our method in some videos. In addition to that, the GMM will always give a higher mixture weight π_i for the distribution $\mathcal{N}(\mu_i, C_i)$ with μ_i closest to the center of mass, which is far from the peak (source) location. Visual examples are provided in Figure 4.



(a) (b)

(c) (d)

Figure 4. Example of an input signal and the DNN-DCT-POCS with 480 input measurements. (a) the ground truth gas leak image with the source marked in red. (b) Location of the all sensors. (c) the sparse input signal to the DNN. (d) the output of the DNN. The global maximum point is marked in blue.



(a) (b)

(c) (d)

Figure 5. Another example of an input signal and the DNN-DCT-POCS with only 100 input measurements. (a) The ground truth gas leak image with the source marked in red. (b) Location of the all sensors. (c) The sparse input signal to the DNN. (d) The output of the DNN. The global maximum point is marked in blue.

2.6.2 Results with Binary-Valued Input Measurements

We trained our neural network by unrolling a few steps of the forward model in Algorithm 2. Similar to the the case of real-valued constraints, we used 4 DCT bandwidth parameters $\mathcal{BW} = \{4, 8, 16, 32\}$. Each inference pass is 4 iterations ($L = 4$ in Algorithm 2) with the POCS procedure applied 4 times ($K = 4$ in Algorithm 2). Our dampening factor β is set to 0.5. We used the same parameters for training and inference. The final output $\hat{x}$ is then optimized to minimize the loss function in Equation 2.24. We construct our logical inputs from the according to Equation 2.30 our averaging window size is 16 frames. Our threshold is calculated using Otsu's method [46], which seeks to find a threshold based on the histogram of the intensity values. Given random sample locations $\mathcal{S}$, we generate the logical masks $\mathcal{S}_u$ and $\mathcal{S}_l$ based on the threshold parameter Th. We trained the neural network over the frames of one video for 10 epochs. We then normalize the ground truth image x by Th and then train it to minimize our loss function.

2.6.2.1 Peak-Location Aware Loss Criterion

Unlike the real-valued case, optimizing the loss function in Equation 2.24 did not produce high intensity values around the peak. This is because the constraint are binary values. To overcome this issue, used the loss function defined in Equation 2.27. This incentives the model to reconstruct images with higher energy close to the source location, which is our ultimate goal to locate the peak. We also compared with the GMM mixture model approach and with traditional with the traditional POCS approach with binary constraints. Our results with 480 input measurements and for 100 input measurements are summarized in Algorithm IV and

Algorithm V, respectively. Two examples of the results are shown in Figure 6 and Figure 7. Here, it is worth mentioning that we do not do the two-step processes as in Sec. 2.6.1. We compare the results using the metric in Equation 2.32 with k=1 (only the argmax of the predicted image) and with k=5 (the locations of the largest 5 points). As for the GMM model, we fitted a mixture with 5 gaussians, i.e., $|\{\pi\}| = 5$ and compared the results with all the estimated modes $\{\mu\}$ and the location of the source. Furthermore, we compared the distance between the gaussian of the highest mixture density and the source location.

Vid ID	Maxima Distance to Source							
	NN		GMM (5 mix.)		POCS BW=16		POCS BW=32	
	Top 1	Top 5	Top 1	Top 5	Top1	Top5	Top 1	Top 5
Exp 1	14.2	11.0	67.9	23.1	70.4	26.2	74.3	37.8
Exp 2	20.5	14.1	70.8	25.3	80.2	34.1	82.2	41.0
Exp 3	17.1	11.6	44.7	12.9	50.6	15.5	52.1	17.1
Exp 4	12.2	11.7	78.5	35.8	85.7	25.6	82.5	50.9
Exp 5	20.3	13.3	46.3	14.5	51.3	17.7	52.0	21.3
Exp 6	37.5	25.6	66.2	27.5	74.2	32.2	70.7	39.3
Exp 7	21.6	10.4	48.2	12.9	52.0	15.1	48.6	18.8
Avg.	**20.5**	**14.0**	60.3	21.7	66.3	23.8	66.1	32.3

TABLE IV

RESULTS WITH 480 SENSOR MEASUREMENTS FOR THE DEEP REGULARIZED POCS COMPARED TO GMM FITTING AND TRADITIONAL POCS.

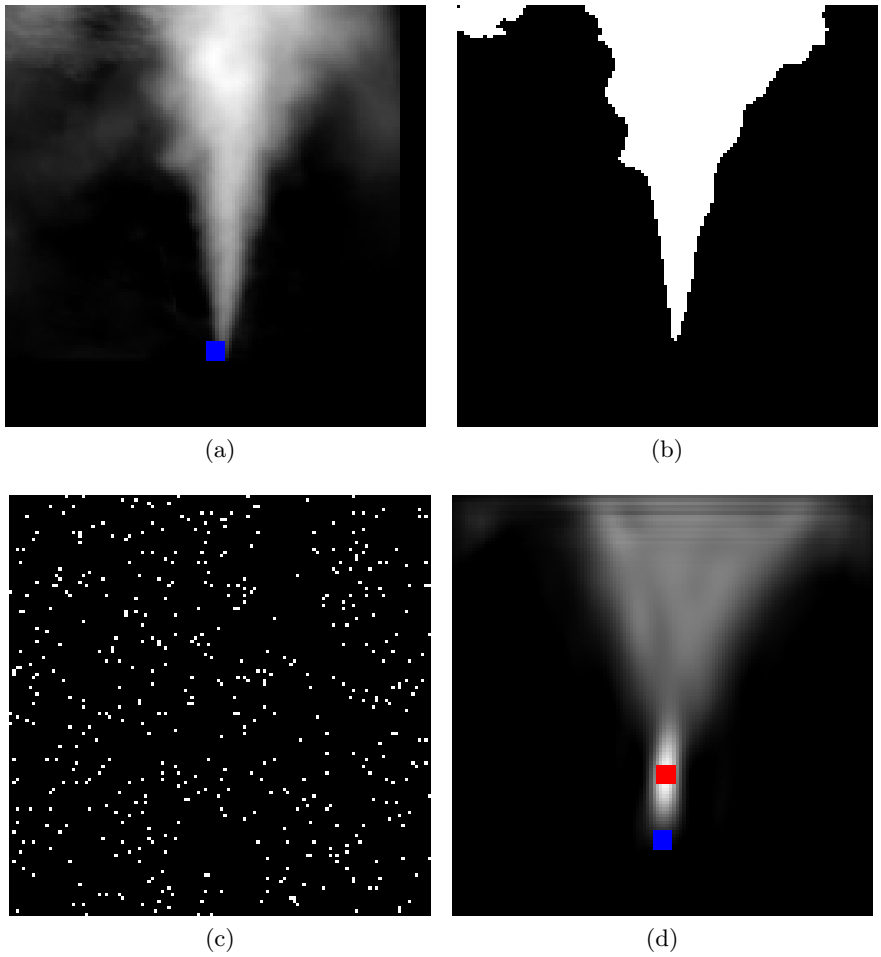
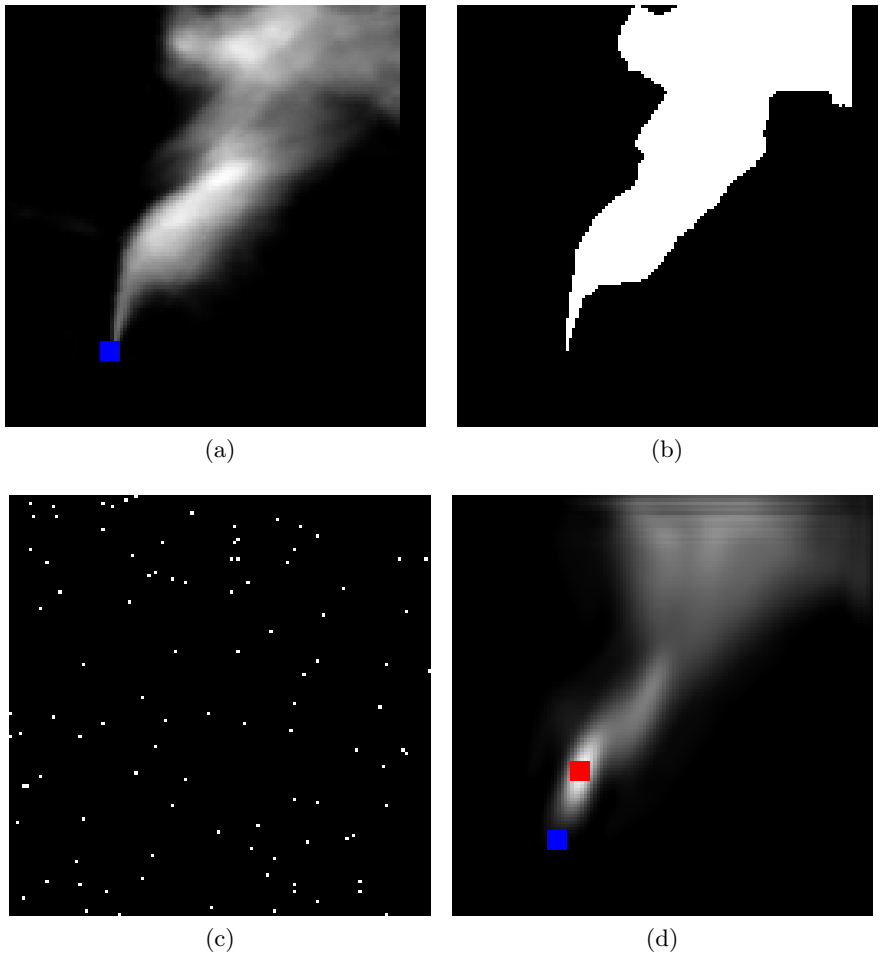


Figure 6. Example of an input signal and the DNN-DCT-POCS with 480 binary input measurements. (a) the ground truth gas leak image with the source marked in red. (b) the fully binarized ground truth. (c) the locations of the sparse measurement. (d) the output of the DNN. The global maximum point is marked in blue.



(a) (b)

(c) (d)

Figure 7. Example of an input signal and the DNN-DCT-POCS with 100 binary input measurements. (a) the ground truth gas leak image with the source marked in red. (b) the fully binarized ground truth. (c) the locations of the sparse measurement. (d) the output of the DNN. The global maximum point is marked in blue.

Vid ID	Maxima Distance to Source							
	NN		GMM (5 mix.)		POCS BW=16		POCS BW=32	
	Top 1	Top 5	Top 1	Top 5	Top 1	Top 5	Top 1	Top 5
Exp 1	23.8	16.6	67.9	23.1	77.5	28.2	76.8	39.6
Exp 2	26.7	23.4	70.8	25.3	80.7	36.1	78.8	37.9
Exp 3	27.5	14.4	44.7	12.9	53.6	19.6	78.8	19.7
Exp 4	29.1	27.1	78.5	35.8	87.9	39.8	85.8	51.2
Exp 5	29.2	18.1	46.3	14.5	54.7	18.6	54.3	23.7
Exp 6	38.3	28.3	66.2	27.5	75.2	32.9	71.6	35.3
Exp 7	33.9	14.5	48.2	12.9	55.7	19.6	53.6	18.6
Avg.	**29.7**	**20.3**	60.4	21.7	69.3	27.8	71.3	32.3

TABLE V

RESULTS WITH 100 SENSOR MEASUREMENTS FOR THE DEEP REGULARIZED POCS COMPARED TO GMM FITTING AND TRADITIONAL POCS.

2.7 Isopropyl Alcohol Leak Detection and Source Localization

We collected sensor measurements using three Tin oxide (SnO2) MQ137 sensors [47]. Data collection was done using an Arduino, as the sensors have an analog output wired connection were used for reliability. For the experiments of source localization a grid of 18×18 inch^2 was created. As the sensors are resistive chemical sensors, they need to be preheated for a certain period of time before testing. This is done so that the sensors form a desired active oxide layer than can detect any gas present in the air. In this case prior to starting the experiments, we heated the sensors for up to 48 hours. These sensors are not calibrated, so to do sensor to sensor calibration firstly a controlled experiment was done. In these controlled experiments, we

placed the three sensors closed to one another so that they all read the same underlying gas concentration signal.

2.7.1 Data Acquisition

The experiment was conducted in the following order:

- Firstly, the source was placed at a an arbitrary location.
- Then the three sensors were placed at different grid points as shown in Figure 8.
- Once the source and the sensors were place at their respective locations, the lid of the source was removed, exposing the air.
- The data was collected from each sensor using the Arduino which transferred the data to a computer.
- After collecting the data for a required amount of time, which was controlled in a way such that the air in the room was not saturated.
- Figure 9 shows how the data from the sensors look like for one set of experiment.
- After one set of experiment was completed, the lid of the source was closed and the room air was vented out. We wait for a while so that the sensors return back to their baseline response, indicating that there is no more gas in the air.
- This was repeated 6 times with different source and sensor positions on the define grid and corresponding sensor data was collected each time. This data was then used in our source localization framework.

2.7.2 Sensors Calibration

Ideally, the relation between the sensor measurement and the actual concentration can be approximated by the following formula

$$s(t) = Ac(t) + b, \tag{2.35}$$

where $s(t)$ is the sensor measurement (in volt) and $c(t)$ is the gas concentration (in ppm). b is the DC offset, and A is a gain factor. Each sensor will have its own parameters A and b. We first estimated the DC offset using the measurement values when no gas signal is present. We then calculated the relative gains of each sensors with respect to a reference sensor. Let sensor s_1 be our reference sensor, The relative gain is given by

$$\frac{A_i}{A_1} = \frac{1}{T}\sum_{t=1}^{T}\frac{s_i(t) - b_i}{s_1(t) - b_1} \tag{2.36}$$

We then calibrated the sensor measurements using the following formula

$$\hat{c}_i(t) = \frac{A_1}{A_i}(s_i(t) - b_i) \tag{2.37}$$

One issue with this calibration approach is that the actual sensor response can deviate significantly from the ideal model expressed in Equation 2.35. In order to get reliable and robust estimates for the relative gains, we manually selected segments for calibration in which the sensor responses are linearly correlated the most.

According to our experiments, we found that the DC offsets b_1, b_2, and b_3 are equal to 0.7 Volt. The relative gains are $\frac{A_2}{A_1} = 1.43$ and $\frac{A_3}{A_1} = 1.61$.

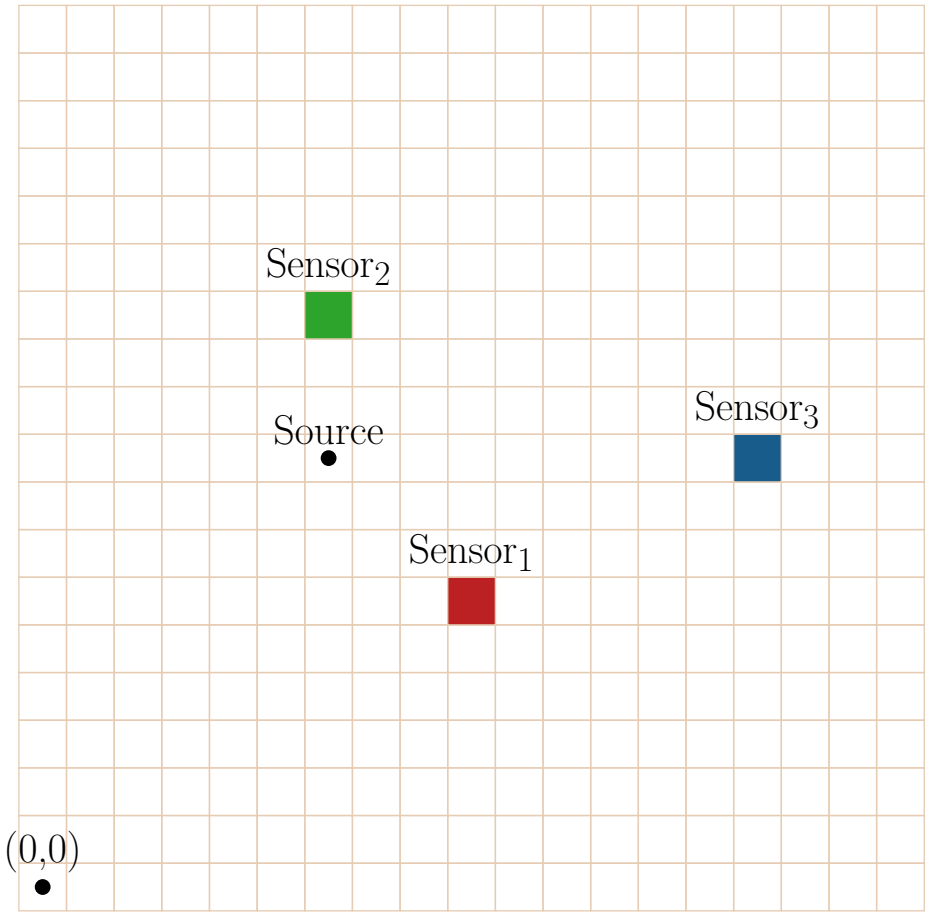


Figure 8. Illustration of the isopropyl alcohol 18×18 inch^2 grid with the source stationed at $(x = 6, y = 9)$ and the three sensors located at $(x = 9, y = 6)$, $(x = 6, y = 12)$, and $(x = 15, y = 9)$. The corresponding time series are shown in Figure 9

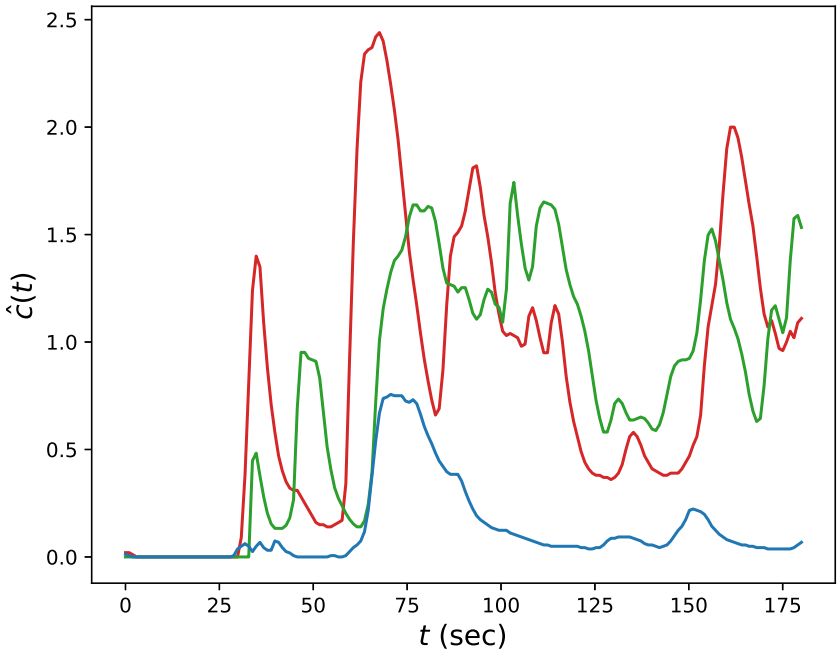


Figure 9. The corresponding calibrated time series measurements of each sensor in the experiment shown in Figure 8

2.8 Experimental Results

After calibrating the chemical sensor data as in accordance with the procedure explained in 2.7.1, we down-sampled the input data by a factor of 2. Therefore, our sampling rate is 1 sample/sec. The original images have a spatial dimension of 18×18. Because we use 32-point DCT transform, our input images and masks must be 32×32 pixels. We pad zeros to the original 18×18 images, and we set our zero mask $\mathbf{M}_z$ as follows:

$$[\mathbf{M}_z]_{ij} = \begin{cases} 1, & 9 \leq i \leq 24 \quad \text{and} \quad 9 \leq j \leq 24 \\ 0, & \text{otherwise} \end{cases} \tag{2.38}$$

This means that we limit the support of to be equal to the original input image size while using a higher resolution DCT in our POCS layers. Given the limited number of data points at each time step, we feed 4 consecutive time frames to our neural network in order to capture some information from the temporal behavior of the the measurements at each location, i.e., our input size is $32 \times 32 \times 4$, with a total number of 12 non-zero constraints.

2.8.0.1 Training the Model with Partially Known Ground Truth

Given the limited amount of measurement we have at each time step, and the fact that we do not have the full gas signal across the entire grid, this makes the problem of estimating the isopropyl alcohol source location very challenging.

In order to supervise the neural network, we create a dense "gas field" signal as follows: given the 12 input constraints, we add 4 additional fake measurements at the source location

for each temporal map. The intensity is selected large enough to be larger that any of the measurements. We then use the POCS procedure to create an artificial dense ground-truth signal from the sparse constraints such that its peaks at each time step coincide with true source location. We then stored these dense fields and supervised the neural network to minimize the following loss

$$\mathcal{L}(\hat{f}) = ||\hat{f} - f_{\text{synth}}||_2^2, \tag{2.39}$$

where $\hat{f}$ is the output of the neural network, and f_{synth} is the synthetic signal with $\underset{N\times N}{\text{argmax}}(f_{\text{synth}}) = (x_{\text{src}}, y_{\text{src}})$. We used two bandwidth parameters BW_1=4, and BW_2=8. The POCS procedure is implemented for 8 iterations ($K = 8$ in Algorithm 1), while the overall Deep DCT-POCS procedure is carried out only for one iteration ($L = 1$ in Algorithm 1).

We used five experiments for training, three for validation, and three more experiments for testing our model. Our results over the test dataset are given in Algorithm VI. Example of an output image is shown in Figure 10. As one can see, our model achieves better distance score than predicting the source location using the center of mass. This suggests that the time series measurements are very noisy and simple averaging is very sensitive to these fluctuations in signal values as can be seen in Sec. 2.7.1. On the other hand, the model learns robust features from the time series that as less sensitive to these fluctuations.

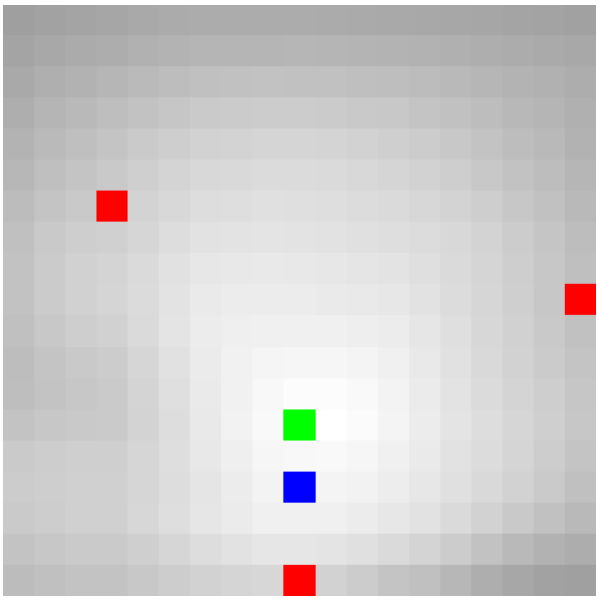


Figure 10. Example of output image given the isopropyl alcohol sensor measurements over three locations. The sensor locations are colored in red. The source is located at the blue point, while the predicted source location (the argmax of the output image) is located at the green point.

Exp	avg dist	
	DNN	Centroid
Exp 1	3.14	4.49
Exp 2	2.77	2.80
Exp 3	3.89	4.82

TABLE VI

AVERAGE DISTANCE BETWEEN THE LOCATION OF THE PEAK OF THE RECONSTRUCTED SIGNAL AND THE TRUE SOURCE LOCATION VERSUS THE DISTANCE BETWEEN THE TRUE SOURCE AND THE CENTROID OF THE THREE SENSOR MEASUREMENTS

2.9 Conclusion

In this chapter, we addressed the issue of locating a leaking gas source given sparse and noisy sensor measurements. For this purpose, we proposed a data interpolation framework that combines deep neural networks with regular projection-onto-convex-sets (POCS)-based one-dimensional signal and two or higher dimensional interpolation algorithms. In particular, we utilized the iterative bandlimited interpolation algorithm, also known as Papoulis-Gerchberg algorithm, with two variants: one deals with real-valued sensor measurements, while the latter deals with binary-valued sensor measurements. Instead of using the Fourier transform, we used the discrete cosine transform to avoid complex numbers. We combined the iterative structure with a convolutional neural network to regularize the iterations to achieve reliable solutions. The overall algorithm is trained with past data.

We experimented with two different data sets at two different scales. The first example is indoor isopropyl alcohol gas leak data, which we collected using three commercially available

chemical sensors. The second data corresponds to methane leak in industrial plants that was extracted from infrared (IR) videos. We tested our approach on the two data sets and we were able to interpolate the gas field spatial signal with high accuracy. We considered the local maxima of the reconstructed two-dimensional data as candidates for the source locations. Our approach achieved better results that Gaussian Mixture Model based interpolation in the case of the methane data, and the center of mass based location estimation in the case of isopropyl alcohol data.

CHAPTER 3

COMPUTATIONALLY EFFICIENT DEEP LEARNING SYSTEM FOR VOC LEAK DETECTION USING INFRARED IMAGING AND CHEMICAL SENSORS

The content of this chapter is based on our work that was published in the IEEE Journal of Selected Topics in Signal Processing, 2020, under the title "Computationally Efficient Spatio-Temporal Dynamic Texture Recognition for Volatile Organic Compound (VOC) Leakage Detection in Industrial Plants" © 2020 IEEE [3].

3.1 Motivation and Background

The US Environmental Protection Agency (EPA) estimates that more than 70,000 tons of Volatile Organic Compounds (VOC) are emitted from leaking equipment such as valves, pumps, and connectors, at petroleum refineries and chemical manufacturing facilities annually [48]. Some types of VOCs such as acetaldehyde, benzene, formaldehyde, methylene chloride, naphthalene, toluene, and xylene are Volatile Hazardous Air Pollutants (VHAPs), which cause cancer, birth defects. Furthermore, they affect reproduction. VOCs are a major contributor to the formation of ozone, which is a toxic gas that causes many respiratory diseases in urban areas and areas close to refineries and chemical plants [49].

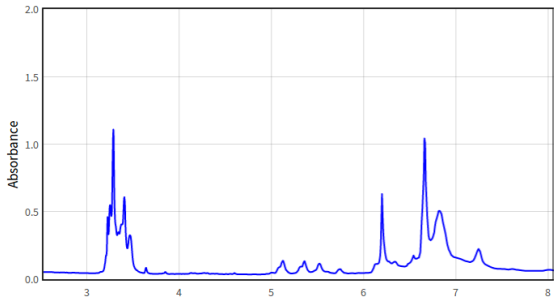


Figure 11. Toluene absorbance as a function of the wavelength in infrared range. The scale of the wavelengths (x-axis) is in micrometers. The plot is downloaded from [1].

Medium Wave Infrared (MWIR) and Long Wave Infrared (LWIR) bands are absorbed by most VOCs. An example is toluene, which absorbs light in both bands. Its absorbance response is shown in Figure 11.

As a result, MWIR or LWIR thermal camera imaging can detect leaking VOC plumes [50–52]. To see the efficacy of using IR thermal cameras, as opposed to ordinary visible light cameras, Figure 12 shows an IR thermal frame and a visible light frame of the same scene. While the VOC leak is obvious from Figure 12(a), which is the IR thermal frame, there is no visible VOC leak in the visible-light frame shown in Figure 12(b). Nevertheless, examining single frames may not be sufficient to detect VOC leak occurrence, as can be seen from Figure 13 (a) and (b), where two frames containing VOC leak are shown. On the other hand, VOC leak regions are not stationary since they move erratically due to wind and other factors. A sequence

of successive frames containing VOC leak is shown in Figure 14. In contrast to the single-frame case, one can easily identify VOC leaks in sequences of frames.

Therefore, a computer vision algorithm should use spatial and temporal information to detect VOC leak regions. The VOC gas leak detection problem in infrared video is similar to the wildfire smoke detection problem [50, 53–59]. Smoke and flames are also dynamic textures in the video [4, 60–64]. While it is possible to train a neural network on cubes of video data for VOC leakage detection, such an approach is not computationally efficient. Furthermore, it may not be possible to implement it on resource-constrained devices for real-time detection.

In this work, we design two types of neural networks for VOC detection. We implement these neural networks in two stages in order to take advantage of the spatial and temporal structure of the dynamic texture created by the leaking VOC plume. We first detect moving pixels that are darker than some of their neighboring pixels. These pixels are likely to be at the boundary of a VOC plume. We then extract one-dimensional (1-D) temporal (history) signals at these locations from the video and feed these 1-D signals to the first neural network. If those pixels are near the edge of a VOC plume, their 1-D temporal signals exhibit high-frequency behavior because VOC clouds move erratically, similar to ordinary smoke. A typical 1-D signal corresponding to a pixel at the edges (or near the edges) of a VOC plume is shown in Figure 15.

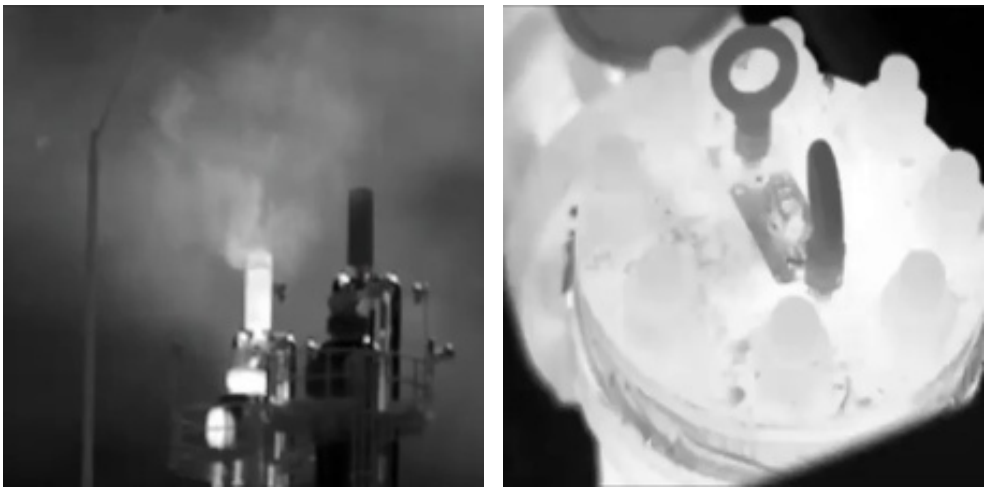
On the other hand, regular background pixels have stationary behavior, as shown in Figure 16. One can notice that the motion patterns of ordinary moving objects are different from the VOC gas leak pixels shown in Figure 15.

(a)

(b)

Figure 12. Example IR thermal image (a) and a corresponding ordinary camera image (b) for the same scene. As we can see, the VOC leak is not visible in the case of visible light image. Images (a) and (b) are taken from [2]

The neural network is trained in such a way that it generates high probability estimates for such pixels. If the 1-D neural network generates high confidence values, we feed the corresponding video frame to a deep convolutional neural network (CNN), which processes image frames. Based on the output of the second CNN, we decide on whether a VOC leak is present in the



(a) (b)

Figure 13. Example IR-thermal frames of VOC leaks. As we can see, it is not easy to figure out the VOC leak from a single image frame.

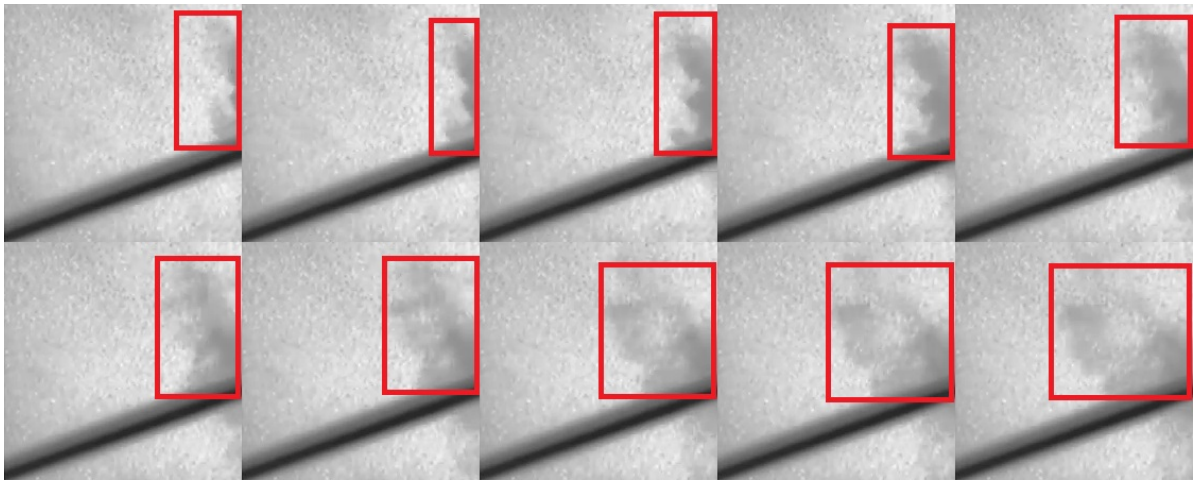


Figure 14. Example IR-thermal frame sequence of VOC leaks.

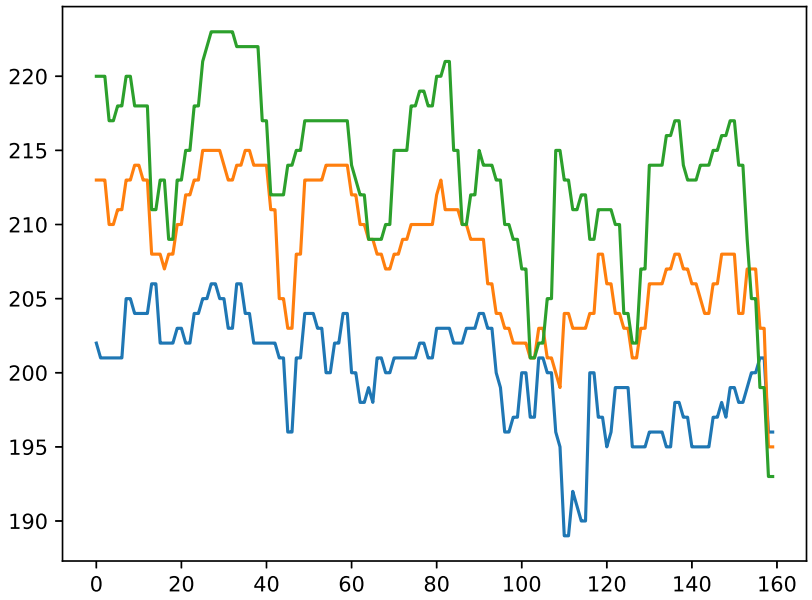


Figure 15. Time-series data of three pixels in VOC leakage regions in IR video.

scene. The overall structure is computationally efficient because the CNN does not process all of the image frames of the captured video. It only processes data after a suspicious activity is identified by the preceding 1-D neural network, which processes the temporal signals of dark pixels.

3.1.1 Organization

The organization of the chapter is as follows. In Sec. 3.2.2 we present the 1-D neural network analyzing the temporal history of a pixel. In Sec. 3.2.3 we present the spatio-temporal 2-D neural network. In Sec. 3.2.4, we present the energy-efficient additive-correlation based

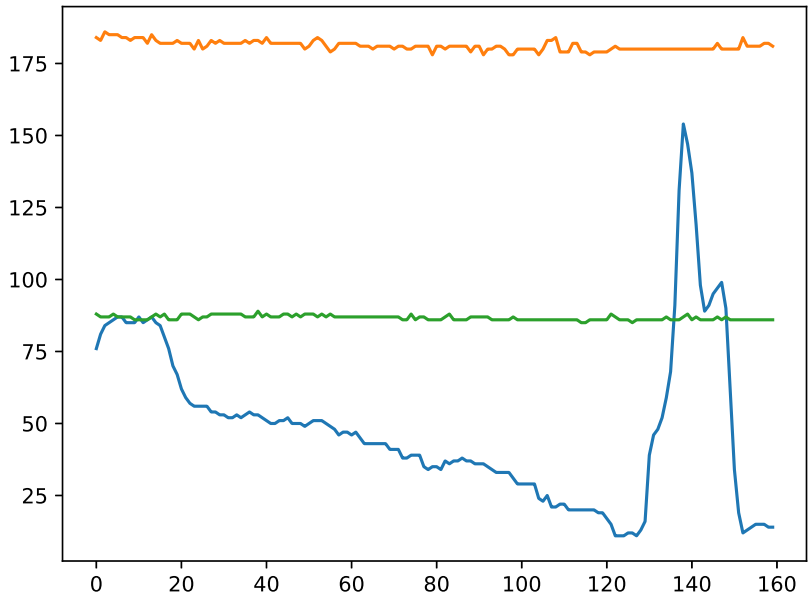


Figure 16. Time-series data of three pixels in thermal IR video. The time-series shown in blue corresponds to a moving object.

spatio-temporal neural network. In Sec. 3.3 we present our experimental results for the different algorithms. Finally, we present our conclusion and discussion in Sec. 3.4.

3.2 Computationally Efficient Spatio-Temporal Video Analysis Framework

As pointed out earlier, plumes of VOC leaks appear as dark regions in a white-hot mode thermal IR video (VOC leaks appear as bright regions in black-hot mode). Such regions do not have a stationary shape over time since they expand and move in an erratic manner that is similar to flames and smoke in regular-imaging videos [56, 58, 65–71]. Therefore VOC leakage in IR video can be determined using spatio-temporal analysis.

Deep neural networks have demonstrated their abilities in complex pattern recognition tasks. However, they are computationally demanding and may not be a practical solution to be used on ordinary low-cost computers for real-time applications. For this purpose, we develop a computationally efficient framework consisting of two components in a cascade. The first component is computationally efficient and can be used for real-time monitoring, while the latter is more computationally demanding and is invoked by the first sub-system only when needed.

In detail, the system first samples temporal trajectories from an IR video and feeds those samples of 1-D time-series signals of pixels of possible VOC regions to a 1-D neural network. Afterward, the system determines a likelihood score of VOC leakage from the results obtained by the 1-D neural network. If the score exceeds a certain threshold, the second component processes the corresponding image frames. The second component is a spatio-temporal convolutional neural network, as shown in Figure 17. In these settings, the task of continuous monitoring is assigned to the relatively efficient 1-D neural network instead of an ordinary 2-D CNN.

3.2.1 Computational Complexity of 1-D CNN vs 2-D CNN

To see the computational saving, one can compare the computational complexity of convolution carried out in 1-D vs. 2-D CNN. For an input of size $N \times N \times D$, and a filter of size $k \times k \times D$, the realization of a single 2-D feature map has a computational complexity $O(N \times N \times k \times k \times D)$. On the other hand, the realization of a single 1-D feature map with an input size $M \times D$ and a filter size of $l \times D$ has a computational complexity of $O(M \times l \times D)$. Therefore, as long as $M < N^2$ and $l < k^2$, the realization of 1-D based convolution feature map requires fewer add-multiply operations. In fact, we have $M << N^2$ in our system, Therefore, the computational saving of the 1-D neural network is enormous.

3.2.2 One-dimensional Temporal Analysis of Dark Moving Pixels in IR Video

The first step of our VOC leak detection method is to identify dark moving regions in IR video. In other words, we extract 1-D temporal records from the original spatio-temporal (video) data. We process these 1-D history signals separately using a neural network to identify whether these history signals are part of a VOC leakage scene or not. If the pixel is at the boundary or near the perimeter of a VOC leak region-As shown in Figure 15, it will exhibit an erratic (high-frequency) behavior over time. On the other hand, if the pixel is from an ordinary object absorbing IR light or a cold place, it will be stationary and exhibit low-frequency behavior most of the time, as shown in

We tried two classifiers of the 1-D temporal data. The first is a regular 1-D convolutional neural network. The second is a long-short term memory (LSTM) recurrent neural network. Our input is a single temporal time-series history signal generated by a moving dark pixel of

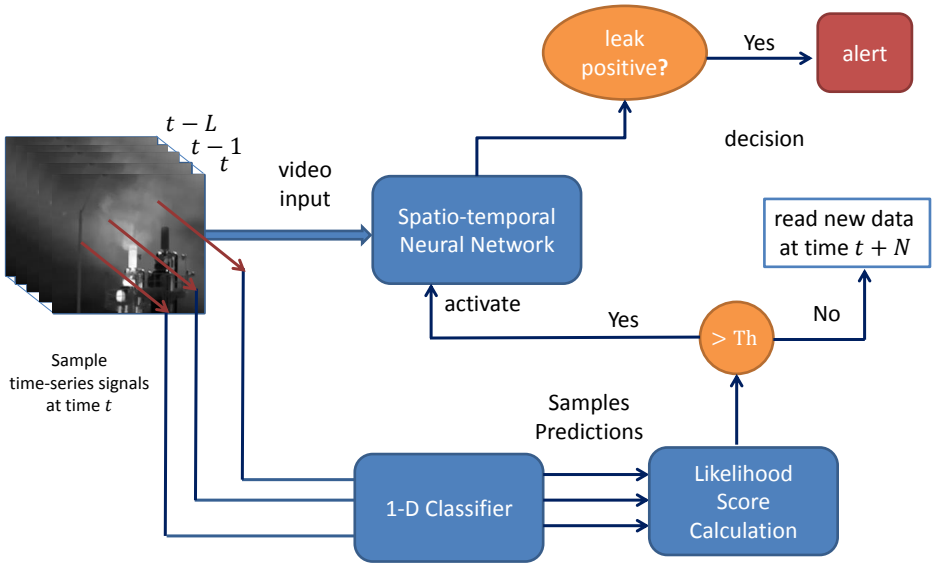


Figure 17. Block diagram of our proposed system. "Th" stands for threshold.

the IR camera, and the output is a binary value indicating whether this specific time-series signal comes from a VOC leakage region or not.

In our settings, we read temporal signals of size 160 at a frame rate of 25 fps, which roughly corresponds to 6.4 seconds from IR videos with spatial dimensions of 224×224. The time span of 6.4 seconds is sufficient for any gas leakage to have a noticeable spread-out across the scene as shown in Figure 15. Therefore, we expect a sufficient number of temporal signals to have time-varying intensity values informative of VOC leakage. Given a 1-D temporal classifier, we

expect to be able to recognize any potential leakage from other moving objects in the IR video. The architecture of the CNN used is given in Algorithm VII.

Since our decision is based on the collective prediction results of the 1-D time-series signals, we devise a confidence score that quantifies our confidence as to whether the scene contains VOC leaks or not. In this regard, let $\mathbf{x} \in \mathbb{R}^{160}$ represents the input signal, which is a vector of length 160. Let $D(x) \in \{0, 1\}$ be the hard decision made by the 1-D convolutional classifier, where "0" corresponds to predicting ordinary temporal signals, and "1" corresponds to a VOC leak. Let N be the number of sample trajectories extracted from the entire spatio-temporal video data. The VOC-leak confidence score is defined as follows:

$$L := \frac{\sum_{i=1}^{N} I\big(D(x_i) = 1\big)}{N}, \tag{3.1}$$

where $I(.)$ is the indicator function and x_i is the i-th signal. In other words, if there are enough time-series signals identified as class Positive (VOC leak), the VOC-leak confidence score L will be high. On the other hand, if the sampled time-series signals do not contain any leakage, the score will be low. When the confidence score L exceeds a certain threshold, the system recognizes a suspicious event and, subsequently, invokes the deep neural network that analyzes spatial data to verify the entire scene. The architecture of the 1-D convolutional neural network is summarized in Algorithm VII.

Layer	Specification
Input Layer Conv Layer Batch-norm Layer	input size: 160×1 32 5×1 filters, strides=2 applied
Conv Layer Max-pooling Layer Batch-norm Layer	64 3×32 filters, no strides down-sampling by 2 applied
Conv Layer Max-pooling Layer Batch-norm Layer	128 3×64 filters, no strides down-sampling by 2 applied
Global Average-pooling	output size: 128
Dense Layer Batch-normalization Layer	output size: 128 applied
Output Layer	output size: 1 (soft prediction)

TABLE VII

ARCHITECTURE OF THE 1-D CONVOLUTIONAL NEURAL NETWORK USED IN TEMPORAL SIGNAL CLASSIFICATION

3.2.3 Two-Dimensional (2-D) Spatio-Temporal Analysis Network

As mentioned earlier, we also utilize the image-based deep convolutional network. Because of the first stage in the system, only a relatively few consecutive frames need to be processed by the image-based deep CNN, in contrast to processing the entire scene. The input to the 2D CNN is 3-D spatio-temporal images, where the first two dimensions correspond to the height and width, and the last dimension corresponds to the number of successive temporal frames. The reason for using multiple consecutive frames rather than single frames is two-fold: First, in some VOC leak scenes, the leakage is very weak and barely discernible by the human eye from

Layer	Specification
Input Layer	input size: $112 \times 112 \times N$
Conv Layer	64 5×5 filters, strides=2
Batch-norm Layer	applied
Conv Layer	128 3×3 filters, no strides
Max-pooling Layer	pooling size: 2
Batch-norm Layer	applied
Conv Layer	256 3×3 filters, no strides
Max-pooling Layer	pooling size: 2
Global Average-pooling	output size: 256
Batch-norm Layer	applied
Dense Layer	output size: 256
Batch-norm Layer	applied
Output Layer	output size: 1 (soft prediction)

TABLE VIII

ARCHITECTURE OF THE 2-DIMENSIONAL SPATIO-TEMPORAL NEURAL NETWORK AND THE ADDNET. "N" REFERS TO THE NUMBER OF SUCCESSIVE FRAMES FED TO THE CNN (TEMPORAL DEPTH DOMAIN). THROUGHOUT OUR EXPERIMENTS, WE SET N TO 3,4 AND 5.

a single frame. Secondly, incorporating temporal information ensures that the network does not erroneously associate spatial features, like pipelines or chimneys, always with the presence of VOC leaks. The architecture of our network is shown in Sec. 3.2.3.

3.2.4 Additive-Correlation Based Spatio-temporal Neural Network

This subsection reviews an energy-efficient neural network that can be used in mobile systems or cameras. We implemented a neural network, which we call the Additive Neural Network (AddNet). AddNet was first introduced in [4, 6, 72] and it performs what we call Additive Cor-

relations (AC) in its neurons. The additive correlation is based on the following arithmetic operation:

$$x \oplus w := \text{sgn}(xw)(|x| + |w|) \tag{3.2}$$

where x and w are two real-valued scalars, sgn(.) is the Signum function. One can easily generalize the definition to the multi-dimensional (vector) case. Let $\mathbf{x}$ and $\mathbf{w} \in \mathbb{R}^d$. We define the additive-correlation of two vectors as follows:

$$\mathbf{x} \oplus \mathbf{w} := \sum_{i=1}^{d} \text{sgn}(x_i w_i)(|x_i| + |w_i|) \ , \tag{3.3}$$

where each entry of the above equation has the same sign of regular multiplication. As a result, whenever x_i and w_i have the same sign, they positively contribute to the AC [4, 6, 72]. On the other hand, if they have opposing signs, they negatively contribute to the AC, just as in the regular correlation operation between the two input vectors. The above operation avoids the use of multiplication operation which consumes significant amount of energy in many mobile systems. It is straightforward to show that Equation 3.3 can be also expressed as follows:

$$\mathbf{x} \oplus \mathbf{w} = \sum_{i=1}^{d} \text{sgn}(x_i) w_i + x_i \text{sgn}(w_i) \tag{3.4}$$

While the ordinary dot product induces the ℓ_2 norm, the new operation induces the ℓ_1 norm, multiplied by 2, as follows:

$$\mathbf{x} \oplus \mathbf{x} = \sum_{i=1}^{d} \text{sgn}(x_i x_i)(|x_i| + |x_i|) = 2||\mathbf{x}||_1 \tag{3.5}$$

The operation can be written in the same fashion as ordinary matrix-vector multiplication: let the vector $\mathbf{x} \in \mathbb{R}^N$ and the weight matrix $\mathbf{W} \in \mathbb{R}^{N \times M}$. We operation is now defined as follows [4, 72]:

$$\mathbf{y} := \mathbf{W} \oplus \mathbf{x} = [\mathbf{x} \oplus \mathbf{w_1} \ \mathbf{x} \oplus \mathbf{w_2} \ \ldots \ \mathbf{x} \oplus \mathbf{w_M}]^T, \tag{3.6}$$

which carries out the AC operations between the columns $\mathbf{w_i}$ of the matrix $\mathbf{W}$ for $i = 1, 2, \ldots, M$ and the input vector $\mathbf{x}$.

In regular fully connected layers, the forward function is expression mathematically as:

$$\mathbf{y} = \sigma(\mathbf{W}^T \mathbf{x} + \mathbf{b}) \tag{3.7}$$

where $\mathbf{W} \in \mathbb{R}^{N \times M}$ is the weights matrix, $\mathbf{x} \in \mathbb{R}^N$ is the output of the previous layer, $\mathbf{b} \in \mathbb{R}^M$ is the additive bias, and $\sigma(.)$ is the nonlinear activation function. In AddNet, one replaces the matrix-vector multiplication in feed-forwarding by the operation defined in Equation 3.6. Furthermore, we introduce a normalization vector $\gamma \in \mathbb{R}^M$ so as to control the range of the

responses of the term $\mathbf{W} \oplus x$. We express the forward operation by an AC-based fully connected layer as follows:

$$\mathbf{y} = \sigma(\gamma \odot (\mathbf{W} \oplus \mathbf{x} + \mathbf{b})) \tag{3.8}$$

where $\odot$ represents the element-wise vector-vector product. Scaling $\mathbf{W} \oplus \mathbf{x} + \mathbf{b}$ by the vector γ is computationally inexpensive because it is an element-wise vector-vector product ($O(M)$). Convolutional AC layers can be realized by replacing the multiplication in the convolution operations with the AC operator. AddNet is more energy-efficient than regular neural networks because it performs only one multiplication per "convolution" operation.

In our experiments, we used the architecture described in Sec. 3.2.3, i.e., the same as in the case of our regular convolutional network used in analyzing IR video frames.

3.3 Dataset and Results

We compiled a data set of hundreds of frames and a data set of thousands of time-series pixel history (trajectory) signals from 29 publicly available IR thermal videos and 12 videos that we recorded using a low resolution bolometer type IR camera. We used these data sets to train the two components of the systems and evaluate their recognition capabilities separately and jointly. It is worth mentioning that some videos contain more than one scene of interest. For information about the data sources, the reader may refer to Appendix.

3.3.1 One-Dimensional (1-D) Data Set

We gathered a time-series data set of 15,000 pixel history signals from 6 normal IR videos, and 7 IR videos that contain VOC leaks in order to create a binary-class data set for training

the 1-D classifier. In order to obtain an accurate data set, we carefully extracted the temporal data from VOC leakage regions from the 7 video clips containing VOC leaks. On the other hand, we randomly extracted 1-D signals from different locations in 6 normal video clips. This is to ensure our normal data cover different motion and intensity patterns. One can also select one pixel out of each 8 by 8 block or 16 by 16 block of the relatively dark regions of the video clip. IR cameras produce Discrete Cosine Transform (DCT) compressed data. Therefore it is also possible to use the DC value of each 8 by 8 or 16 by 16 image blocks. Example time-series signals corresponding to pixels in VOC leaks and ordinary pixels are shown in Figure 15 and Figure 16, respectively.

In a bid to enforce classification invariance to background intensity levels, we augmented our training data set by adding a a small random DC value to the recorded signals while training. Since the 1-D temporal signals are pixel intensities, their values are bounded between $0 - 255$.

We also implemented an LSTM based classifier. Similar to the 1-D CNN case, we used time-series signals of length 160. The architecture is an LSTM layer, which has 20 cells that read the input signals. The output size of the LSTM layer is 20. We then feed the output vector to a dense linear layer, which serves as our output layer. The input to each LSTM cell is a feature vector extracted from a segment of the original temporal signal of length 16. The segments corresponding to adjacent LSTM cells overlap by 50%. The features fed to each cell are the magnitudes of the discrete Fourier Transform (DFT) of 16-sample long time-series segments. The reason for using DFT instead of time-series data segments is to achieve translation invariance.

We reserved 20% of the aforementioned 1-D data set to use it for early-stopping validation purposes. We trained our 1-D neural network for 5 epochs using Adam Optimizer. We were able to achieve 98% accuracy over the validation data set.

In order to establish a VOC-leak confidence score threshold as defined in (Equation 3.1), we carried out stochastic inference over another validation data set of video scenes. This data set is obtained from 19 video scenes (Scene 1-20 as in Algorithm IX and Algorithm XI)[1] , each of which contains successive frames of a spatial size of 224. The videos have significantly different resolutions and scales because they were obtained using different IR cameras. We have 7 videos that do not contain VOC leaks. The remaining 12 videos contain scenes in which there are VOC leaks. The VOC-leak videos have different scenarios and vary greatly in the VOC gas eruption location and the scale. These videos are different from those used initially to train the 1-D classifier. In each trial, we sampled 80 1-D temporal signals randomly from each video and calculated the VOC confidence score defined in Equation 3.1. We repeated the process 10 times for each video. We report the average score and the standard deviation of the confidence score for the different scenes in Algorithm IX. Example IR image frames of the two classes are shown in Figure 18 and Figure 19, respectively.

As we can see from Algorithm IX the VOC confidence scores of temporal signals obtained from ordinary scenes are significantly lower than those of VOC-leak scenes, with the highest confidence score being 0.16 for the 1-D CNN as shown in the 4-th column of Table III. On the

---

[1]For information about the data sources, see Algorithm XX in Appendix.

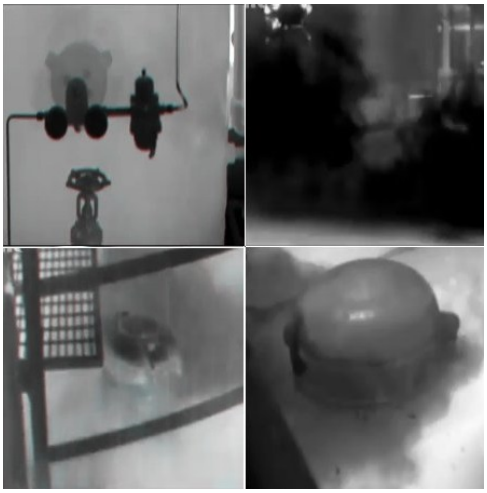


Figure 18. Example IR image frames containing VOC leaks.

other hand, the lowest confidence score for VOC leaks is 0.26. When the input 1-D signals come from VOC leaks, the lowest score is 0.26. We can set the threshold for the confidence score defined in Equation 3.1 as 0.16. All the other scores of video scenes 1-7 are much lower than 0.16. Example 1-D signals and some intermediate layers are shown in Figure 20.

On the other hand, the LSTM-based classifier does not produce as good results as does the CNN-based classifier. The confidence scores of positive videos are consistently higher compared to negative (no-leakage) videos. We can still set a threshold separating the two classes in our data set. However, the score margin between the two classes is small when compared to that of the CNN-based classification (0.35-0.31 vs 0.26-0.16). This suggests that the LSTM-based classifier has a higher tendency of producing more false alarms in comparison with the CNN based classifier.

TABLE IX

RESULTS OF THE CONFIDENCE SCORE OVER DIFFERENT SCENES. THE MEAN SCORES AND THE STANDARD DEVIATIONS ARE ESTIMATED FROM 10 DIFFERENT TRIALS.

Scene ID	Scene Description	Contains leak? (Y/N)	CNN Confidence Score		LSTM Confidence Score	
			mean	std.	mean	std.
Scene 1	wildlife	No	0.01	0.01	0.22	0.04
Scene 2	wildlife	No	0.02	0.02	0.23	0.04
Scene 3	wildlife	No	0.11	0.05	0.21	0.06
Scene 4	wildlife	No	0.16	0.08	0.20	0.06
Scene 5	road	No	0.05	0.04	0.23	0.05
Scene 6	road	No	0.03	0.01	0.31	0.08
Scene 7	road	No	0.02	0.01	0.18	0.05
Worst-Case Score (for VOC negative videos)			0.16		0.31	
Scene 8	pipe leak	Yes	0.42	0.1	0.71	0.09
Scene 9	pipe leak	Yes	0.41	0.08	0.40	0.08
Scene 10	jet engine	Yes	0.54	0.09	0.57	0.08
Scene 11	chimneys	Yes	0.36	0.13	0.50	0.07
Scene 12	pipe gas leak	Yes	0.37	0.1	0.35	0.08
Scene 13	natural gas leak	Yes	0.85	0.06	0.47	0.09
Scene 14	natural gas leak	Yes	0.26	0.11	0.47	0.07
Scene 15	chimneys	Yes	0.49	0.11	0.38	0.07
Scene 16	VOC leak	Yes	0.34	0.08	0.72	0.08
Scene 17	VOC leak	Yes	0.71	0.07	0.76	0.09
Scene 18	chimneys	Yes	0.37	0.07	0.68	0.07
Scene 19	natural gas leak	Yes	0.62	0.12	0.64	0.08
Worst-Case Score (for VOC positive videos)			0.26		0.35	

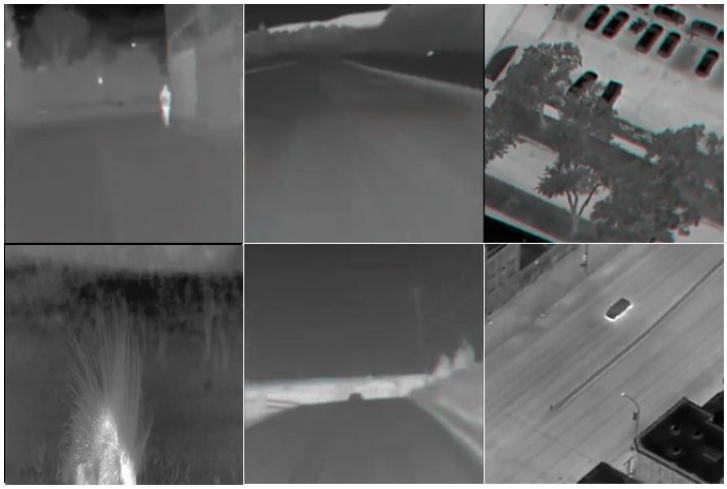


Figure 19. Example ordinary image frames from IR videos used in training the neural networks

3.3.2 IR Video Dataset for 2-D Spatio-Temporal Processing

We extracted infrared image frames from 17 publicly available videos, which account for 48 different scenes, and constructed a training data set for the spatial classifier stage [1]. For validation, we utilized the data set we used earlier for establishing the confidence score for the 1-D classifier as mentioned in Sec. 3.3.1 (Scene 1-20).

We used entire frames and we tried different temporal depths. In particular, we set our temporal depth to 3, 4 and 5 image frames, respectively. In our data set we normalized the image input size to 112×112. We gathered a total of $8,400$ frames of VOC scenes and $10,246$

[1]See Algorithm XXI in Appendix for details about the data sources

frames of ordinary scenes for our training data set. As for the validation data set, we used the data set that we test upon the 1-D neural network as detailed in Sec. 3.3.1.

In order to enhance the capabilities of the network to detect VOCs, we randomly rotated the frames in the spatial domain during training so that the network is exposed to the textures in all different locations. This mitigates the risk of having the classifier over-fit the background scenes. Furthermore, We employed the early stopping criterion based on the recognition rates of the validation data set. Our validation dataset consists of 7 normal scenes and 12 VOC-leakage scenes. This is the same dataset we used for validating the 1-D temporal signals. It should be noted that both training and validation sets are disjoint. We implemented an ordinary 2-D convolutional neural network and an AddNet, both of which have the architecture shown in 3.2.3. We investigated 3, 4 and 5 consecutive image frames for the temporal depth of the input. We used Adam optimizer with a learning rate of 10^{-4}. We used Tensorflow-Keras in our implementation. We compared our method with a regular smoke detection algorithm that we developed with Mobilenet-V2 [73]. We utilized transfer learning using Mobilenet-V2 due to the relatively small data set size. We trained (fine-tune) only the last dense layers while keeping the weights of the convolutional layers intact. Our VOC image frame recognition results over the validation data set were $91 - 95\%$ for regular networks with different number of input frames (3-5) and 93% for AddNet. We were able to identify the events of VOC leak in all of the videos. Smoke detection algorithm developed using Mobilenet-V2 did not produce as good results as our algorithms.

We also tested the neural networks over a set of videos we gathered using a bolometer type FLIR IR camera. These 12 videos contain butane leakage [1]. We report the results per video/scene in Algorithm X and Algorithm XI. Images generated by low cost bolometer IR cameras are corrupted by noise as shown in Figure 21. As it can be seen from Algorithm X and Algorithm XI, the true positive rates are high in the case of CNN and AddNet, in contrast to Mobilenet-V2, which misses out more VOC-positive videos. This can be attributed to the fact that the earlier layers in Mobilenet-V2, which are used for feature extraction and are not trained during fine-tuning, were originally optimized using a dataset, namely ImageNet, which is radically different from IR-thermal images. Furthermore, despite the fact that AddNet and CNN do suffer from false positive rates in some videos as demonstrated in Algorithm XI, what matters the most is not missing out any VOC-positive leak. It should be pointed out that we use image analysis to verify the results of 1-D network which detects all the VOC leaks in our validation data set. Two feature maps corresponding to the same filter from the first convolutional layers are shown in Figure 23 for a VOC leakage example and a normal example. As it can be seen from Figure 23, feature maps have high response in the regions of darker areas. This is expected since the gas leak will generate dark spots in white-hot mode thermal IR video. Nevertheless, we can see that the response is zero for the animal appearing in the negative example as in Figure 23 (a). We can interpret this feature extraction process to be sensitive to darker areas, while not responsive to other patterns.

---

[1]These videos are available on YouTube given the following link: `https://www.youtube.com/playlist?list=PL9_9ATqpfzwPcHBnq2UdxHJ96aSVgVTF-`

TABLE X

TRUE POSITIVE RATES OVER BUTANE-POSITIVE IR-THERMAL VIDEOS WE GATHERED USING A LOW RESOLUTION BOLOMETER-TYPE IR CAMERA. RESULTS ARE SHOWN FOR DIFFERENT NUMBER OF INPUT CHANNELS (FRAMES) OF THE 2D CNN.

Video ID	# of Frames	True Positive Rate (%)				
		CNN			Add-Net	Mob-NetV2
		3 frames	4 frames.	5 frams		
104414	101	100.0	60.5	72.2	98.6	56.3
104703	56	84.0	82.0	65.6	70.7	27.6
103421	104	22.7	20.3	100.0	0.0	0.0
104741	95	100.0	80.2	100.0	100.0	0.0
103934	103	100.0	100.0	100.0	98.2	3.0
104903	100	3.9	70.6	100.0	100.0	93.0
104955	97	100.0	98.9	100.0	100.0	100.0
104255	99	100.0	100.0	100.0	100.0	30.2
103732	78	100.0	80.0	100.0	100.0	10.0
104534	88	100.0	100.0	100.0	100.0	0.0
103236	113	74.8	20.2	10.1	2.6	67.8
103126	97	100.0	94.5	100.0	100.0	100.0
Total	1131	81.1	74.8	86.6	79.3	42.1

We also note that bolometer type low-cost IR cameras are not as reliable as regular LWIR or MWIR cameras due to the noisy nature of bolometer images.

3.3.3 Joint Performance Evaluation and Discussion

We tested the overall VOC detection system over a data set consisting of 7 normal and 5 VOC-positive scenes extracted from 9 videos[1]. The recognition results are reported in Algo-

[1]For information about the data sources, see Table IX in the following link: `https://github.com/Diaa0/Volatile-Organic-Compound-VOC-Leakage-Detection/blob/master/VOC_leak_appendix.pdf`

TABLE XI

RECOGNITION RATES OVER THE POSITIVE AND THE NEGATIVE SCENES WE USED IN OUR VALIDATION DATA SET.

Scene ID	Scene Description	Contains leak?	# Frames	CNN			AddNet	MobileNet-V2
				3 frames	4 frames	5 frames		
				False Positive Rate (%)				
Scene 1	wildlife	No	48	0.0	0.0	0.0	0.0	41.7
Scene 2	wildlife	No	48	0.0	0.0	0.0	0.0	14.6
Scene 3	wildlife	No	48	0.0	0.0	0.0	100.0	0.0
Scene 4	wildlife	No	82	7.2	0.0	0.0	1.2	7.3
Scene 5	road	No	143	26.6	55.2	11.8	0.0	11.9
Scene 6	road	No	485	4.1	0.0	0.0	0.0	2.4
Scene 7	road	No	67	28.4	0.0	0.0	0.0	9.0
Total	-	-	921	8.3	8.6	1.8	5.3	7.4
				True Positive Rate (%)				
Scene 8	pipe leak	Yes	157	95.6	80.2	62.3	80.3	87.3
Scene 9	pipe leak	Yes	289	99.7	100.0	91.9	91.0	82.0
Scene 10	jet engine	Yes	51	60.8	100.0	38.2	100.0	31.4
Scene 11	chimneys	Yes	72	100.0	100.0	100.0	100.0	100.0
Scene 12	pipe gas leak	Yes	156	100.0	100.0	100.0	54.5	94.9
Scene 13	natural gas leak	Yes	48	100.0	100.0	100.0	100.0	100.0
Scene 14	natural gas leak	Yes	58	100.0	49.1	100.0	12.7	100.0
Scene 15	chimneys	Yes	50	98.0	0.0	100.0	100.0	100.0
Scene 16	VOC leak	Yes	41	100.0	100.0	100.0	0.0	100.0
Scene 17	VOC leak	Yes	629	75.8	100.0	99.8	92.9	4.3
Scene 18	chimneys	Yes	406	100.0	99.7	100.0	69.7	55.4
Scene 19	natural gas leak	Yes	3544	99.4	100.0	93.2	98.9	97.7
Scene 20	natural gas leak	Yes	528	41.5	96.8	100.0	100.0	0.0
Total	-	-	6029	91.6	97.7	94.1	92.9	75.0

rithm XII. As one can see from Algorithm XII, the 1-D CNN classifier with zero-mean input (CNN 0) recognizes all the VOC leaks in positive video clips. However, it produces false alarms in Scene 26 (aerial scene) and the wildlife scene 27. The 2-D spatio-temporal network with 3 or 5 image inputs can correct the false alarm in Scene 26, as shown in the 6th row of Table VI.

The 1-D network without mean subtraction (CNN 1) recognizes all the VOC leaks except the gas pump (scene 28) and produces a false alarm in Scene 27. The gas pump leak is relatively faint compared to other leak-positive video clips. An example frame from Scene 27 is shown in Figure 24. The 2-D spatio-temporal network recognizes the leak in more than 80% of the image frames of the IR video consisting of 177 frames.

The 2D spatio-temporal networks with 3, 4, or 5 inputs recognize the VOC leaks in all the video clips. They have a low recognition rate in Scene 29. However, it is enough to identify the VOC leak to sound an alarm. The 2D network with 3 or 5 images produces a false alarm only in Scene 27.

3.3.4 Computational Efficiency of AddNet

We carried out time analysis over inference passes for a regular CNN and AddNet on a PC equipped with an Intel Core I7-7700HQ CPU. We measured the inference time for mini-batches of different sizes. The averaged results are presented in Algorithm XIII. As it can be seen from Algorithm XIII, the computational efficiency of AddNet is not significant in the case of a single-example batch. However, AddNet could process batches of 3 images faster than CNN by 5% in a regular PC. Furthermore, It can achieve 15% efficiency when the batch size increases to 20 images. Cameras output compressed video, and decoders generate batches of

TABLE XII

THE PERFORMANCE RESULTS OF THE SPATIO-TEMPORAL VOC DETECTION SYSTEM OVER A TEST DATA SET.

Scene ID	Scene Description	Contains leak?	# of frames	confidence Score (1D)		VOC Recognition rates (2D CNN)		
				CNN 1	CNN 0	3 frames	4 frames	5 frames
Scene 21	building	No	417	0.08	0.0	0.0	0.0	0.0
Scene 22	pedestrians	No	237	0.14	0.04	0.0	0.0	0.0
Scene 23	pedestrians	No	507	0.08	0.05	0.0	16.9	0.0
Scene 24	building	No	717	0.08	0.07	0.0	0.0	0.0
Scene 25	aerial scene	No	230	0.06	0.04	0.0	10.0.0	0.0
Scene 26	aerial scene	No	117	0.11	0.41	0.0	89.0	0.0
Scene 27	wildlife	No	148	0.25	0.23	89.0	87.8	87.5
Scene 28	gas pump	Yes	177	0.10	0.57	88.0	90.0	85.1
Scene 29	oil well	Yes	177	0.27	0.44	12.4	13.6	18.2
Scene 30	pipe gas leak	Yes	177	0.18	0.47	100.0	100.0	100.0
Scene 31	pipe gas leak	Yes	217	0.19	0.26	32.5	100.0	100.0
Scene 32	pipe gas leak	Yes	207	0.19	0.40	99.5	97.2	98.0

image frames in practice. The time saving that we achieved using AddNet allows for processing more mini-batches of spatio-temporal frames. This, in turn, increases the recognition capacity of the system.

It is worth mentioning that the energy efficiency of AddNet depends on the type of processor that is being used in video analysis, but it is related to the computational time savings.

TABLE XIII

EXECUTION TIME RESULTS OF CNN AND ADDNET MINI-BATCH INFERENCE FOR DIFFERENT MINI-BATCH SIZES.

Mini-Batch Size	CNN Average (ms)	AddNet Average (ms)	AddNet Saving Rate
1	2.532	2.489	1.70%
3	1.175	1.106	5.88%
5	1.147	1.051	8.37%
10	1.074	0.968	9.87%
20	0.994	0.839	15.59%

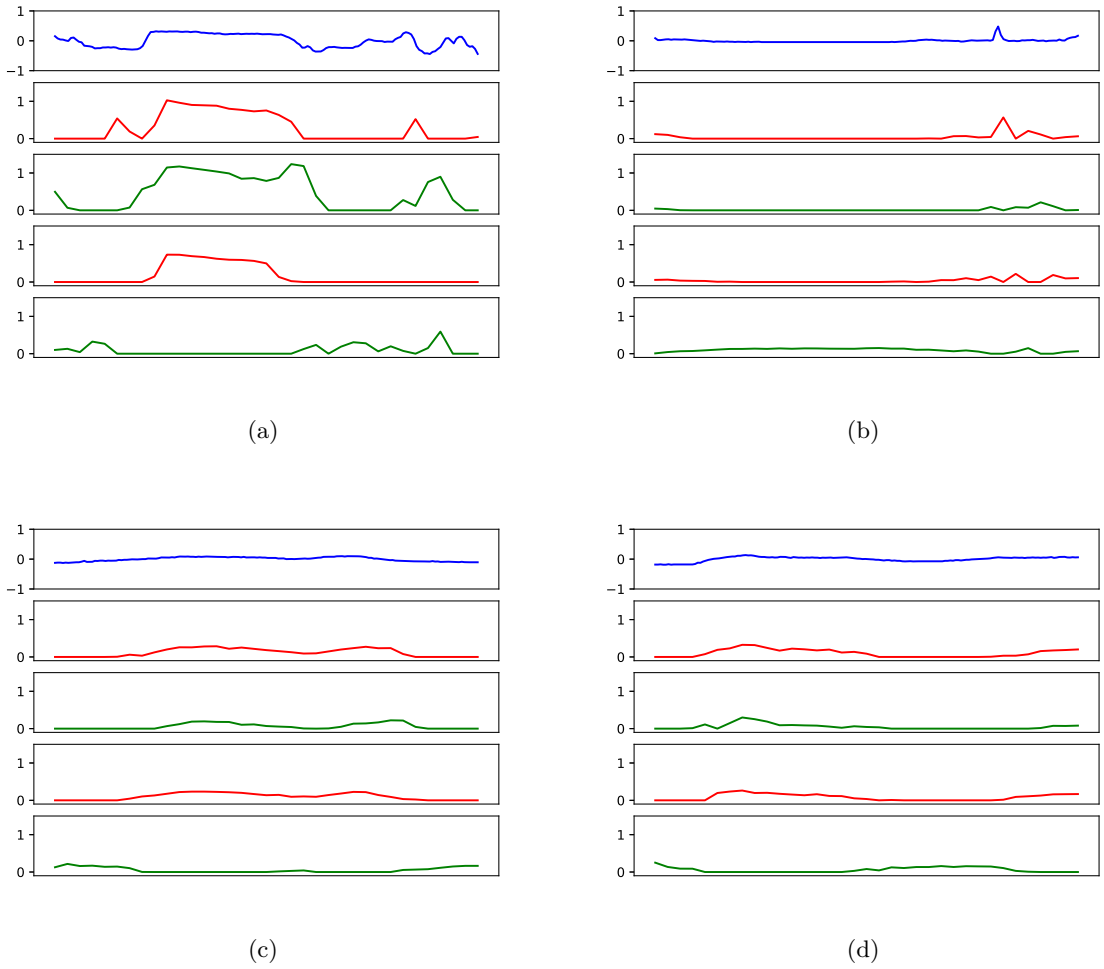
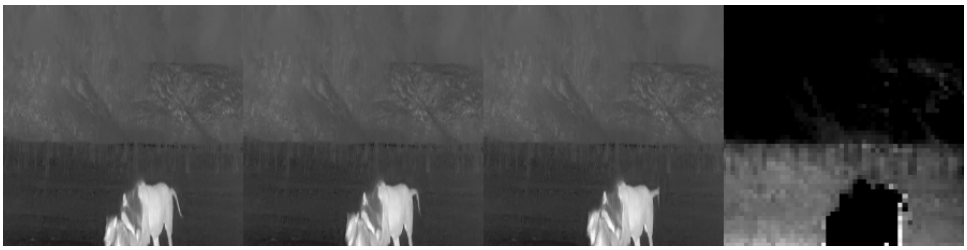


Figure 20. Intermediate feature maps for 4 different example time-series signals. The signals in blue are the input signals and the signals in green and red are four different features maps from the second convolutional layer. Example (a) is taken from a VOC-positive video and classified as VOC-positive. Example (b) is taken from a VOC-positive video and classified as VOC-negative. Example (c) is taken from a normal (VOC negative video) and classified as negative, whereas Example (d) is taken from a VOC negative video and classified erroneously as VOC-positive. Each feature signal has a length of 35 samples and is scaled to match the length of the original signal (160 samples) for demonstration purposes.

Figure 21. An example of thermal image obtained by a low-cost bolometer-type IR camera. The darker region corresponds to butane leakage.

Figure 22. Example image frames from various videos that we used in testing our deep neural networks. All the frames except the bottom-left frame are correctly recognized by the CNN and AddNet.



(a)

(b)

Figure 23. Example feature maps of the first convolutional layer for two examples: (a) A wildlife scene with no VOC and (b) a scene with VOC gas leak. The values are re-scaled for demonstration purposes.

Figure 24. Example image from Scene 27 (pump gas leak).

3.4 Conclusion

We presented a computationally efficient VOC gas leak detection method, which is based on two neural networks connected in series. The first neural network analyzes the time-series data generated by some of the moving dark pixels of the thermal IR camera. If such pixels exhibit erratic behavior, it is possible that the scene may contain a dark cloud-like region due to a VOC gas leak. In such cases, three or more consecutive frames of the video are fed to the 2-D spatio-temporal neural network. The overall system achieved high recognition rates in our dataset. The VOC-leak detection structure that we propose is scalable in the sense that one can use only 1D temporal history signals if the processor is of limited capacity. If more processing power is available, the 2D spatio-temporal network can also be used for more reliable VOC-leak detection results. The spatio-temporal network will verify the results of the 1D temporal neural network. We also used a novel neural network, AddNet, which is based on what we call the "additive-correlation" operation. The AddNet is more energy-efficient than regular neural networks. For this reason, they can be used in mobile applications, including drones. The recognition results of the AddNet are slightly inferior to the regular deep 2-D convolutional neural network. However, the gain in energy-efficient and time saving of AddNet allows for processing more frames to compensate for the drop in performance.

CHAPTER 4

REAL-TIME LOW-COST DRIFT COMPENSATION FOR CHEMICAL SENSORS USING A DEEP NEURAL NETWORK WITH HADAMARD TRANSFORM AND ADDITIVE LAYERS

The content of this chapter is based on our work that was published in IEEE Sensors Journal, 2021, under the title "Real-Time Low-Cost Drift Compensation for Chemical Sensors Using a Deep Neural Network With Hadamard Transform and Additive Layers" © 2021 IEEE [8]. A preliminary version of this work was published in IEEE International Conference on Acoustics (ICASSP) 2021, under the title " Discrete Cosine Transform Based Causal Convolutional Neural Network for Drift Compensation in Chemical Sensors" © 2021 IEEE [7]

4.1 Introduction

Drift correction is a crucial pre-processing step for reliable and accurate gas analyte detection and identification in chemical sensors and Electronic nose (E-nose) systems [19, 74–77]. Sensor drift causes the characteristics of a chemical sensor's response to change over time. This happens due to multiple factors, such as variations in temperature and humidity, aging, and the so-called sensor poisoning [18]. An electronic nose system can neither be reliable nor accurate without addressing the sensor drift problem [78].

Recently, there has been significant interest in drift correction in the sensors community by utilizing machine learning algorithms. Zhang *et al.* [79] propose an unsupervised subspace

projection method, named Domain Regularized Component Analysis (DRCA), which aims at adapting the distribution of the drifted data to that of drift-free data for analyte classification. Liu *et al.* [80] propose an online active-learning algorithm that calibrates sample drift for class identification. In [81], Tao *et al.* propose learning drift-invariant features in an adversarial manner by minimizing the Wasserstein distance to perform domain adaptation between the drifted data and the drift-free data domains. In [6], we proposed a generative adversarial framework to train a discriminator-classifier network to learn drift-invariant feature parameters using the chemical sensor dataset collected by Vergara *et al.* [18].

While the previous work addresses long-term drift in chemical sensors, the machine learning algorithms are implemented over several extracted features, such as the maximum and minimum values of the original time-series data. Unfortunately, the original time-series measurement data is not available in [6, 18, 80, 81]. Huang *et al.* propose a Papoulis-Gerchberg (PG) algorithm-based method for drift correction. The PG algorithm-based algorithm first extrapolates the drift signal from the observed data by assuming that the drift signal is a band-limited low-pass signal [82]. This PG-based algorithm is not applicable to online, real-time applications since the PG algorithm is an iterative method that requires computing successive Fourier Transforms (FT) between time and frequency domains until a satisfactory convergence level is achieved. Furthermore, some baseline drift samples should be known for convergence. Other chemical sensor drift estimation approaches include Kalman filtering and shallow neural network-based methods [78, 83–87].

In this chapter, we propose using deep learning to estimate the sensor drift signals from raw time-series data in real time. Deep convolutional neural networks (CNNs) have excelled in various time-series related tasks, such as prediction, interpolation, and classification [88]. Researchers have recently used CNNs in the analyte classification problem [3,6,89]. CNNs have been increasingly preferred to recurrent neural networks in time-series recognition problems thanks to their highly flexible architectures and relative ease to train [90]. Since our goal is to carry out drift correction in real-time using a causal regression framework, we propose to use novel temporal convolutional neural networks (TCNNs) for sensor drift estimation. TCNNs have been able to outperform recurrent neural networks over several benchmark data sets [90,91]. In TCNNs, convolutional layers implement causal convolution (or correlation), meaning that the current output depends only on the current and previous input values. Convolutions are carried out at different dilation rates, thus enabling the network to learn long- and short-term features for the task.

To take advantage of the slowly varying nature of the sensor drift signal, we propose incorporating orthogonal transforms and thresholding layers in the TCNNs architecture to produce smooth intermediate features, which in turn will generate a smooth drift estimate. This approach also removes the noise in the observed data. In particular, we compute the Hadamard Transform and Discrete Cosine Transform (DCT) over sliding windows of the past and current intermediate feature maps and apply soft thresholds to the high-frequency components in the transform domain. The transform layer will essentially suppress the high-frequency components and regularize the features. The thresholding parameters are also learned during training us-

ing artificially created data. Both transforms are fast and can be implemented efficiently using $O(n \log n)$ operations. Hadamard transform is even faster than the DCT as it is a multiplication-free transform, which can be constructed from the Haar wavelet transform [92]. Furthermore, we replace the convolutional layers of TCNN with multiplication-free additive layers that can be implemented efficiently on a low-performance processor to design an energy-efficient and low-cost system.

Our results show that the proposed framework can accurately provide smooth and slowly varying drift estimates from the sensor measurements in real-time, even for severely degraded sensors.

4.1.1 Organization

The organization of this chapter is as follows: In Sec. 4.2, we review the sensor drift problem. In Sec. 4.3 we describe the TCNN framework, the transform domain layers, and the additive convolutional layers. In Sec. 4.4 we present a novel multiplication-free operator that defines a mercer-type kernel. In Sec. 4.5 we present an algorithm for unsupervised drift estimation using kernel PCA and DCT. In Sec. 4.6, we present and discuss our experimental results. Finally, we provide our conclusion in Sec. 4.7.

4.2 The Sensor Drift Problem

Sensor drift is a common problem in chemical sensors, such as the E-nose technology, leading to inaccurate measurements. Sensor drift stems from many sources, such as the binding of molecules to the sensor surface, electronic aging of components, environmental contamination, and temperature variations. Examples of sensors drift are shown in Figure 25 and Figure 26.

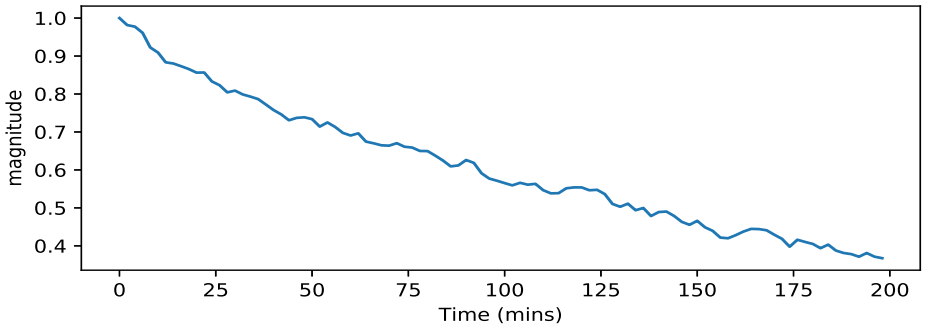


Figure 25. Example of drift signal showing the low-frequency nature of the drift signal

In Figure 25, the signal decays over time, although there is no methane gas excitation. Comparatively, in Figure 26 the baseline drift signal first decreases until around t=110 minutes and slowly increases afterward. Ideally, both sensors should not have produced any output. Therefore, the measured output is the offset, which drifts in time. Furthermore, although the sensors are identical in design, they exhibit different drift signals. These two examples show that it is difficult to rely on analytical models to characterize drift signals due to the individual nature of each sensor.

In [82], it is further assumed that the drift signal is a band-limited lowpass signal, and the drift estimation is formulated as an interpolation problem. The iterative Papoulis-Gerchberg (PG) algorithm [30, 32, 93] is used to construct missing parts of the drift signal. In particular, this drift estimation method assumes that the sensor is not exposed to a chemical gas initially and at the end of the measurement cycle. PG algorithm imposes time and frequency domain information iteratively to perform interpolation until both the time and frequency constraints

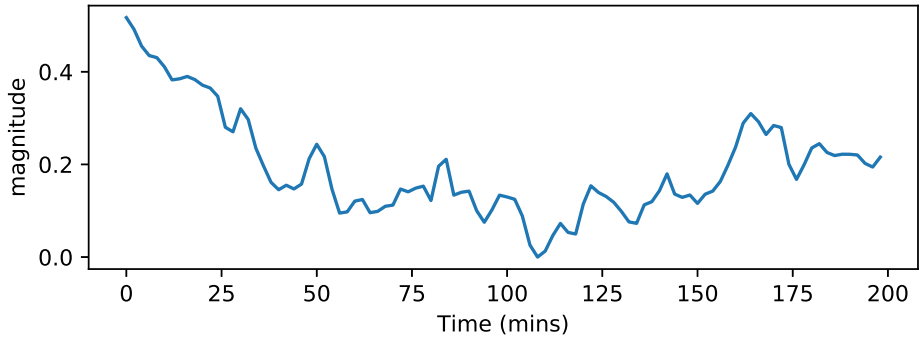


Figure 26. Example of drift signal showing that the drift signal can increase and decrease

are satisfied, or a satisfactory convergence level is achieved. The shortcoming of the PG method is the need for prior information about the gas exposure to be able to interpolate a section of the drift signal corresponding to gas exposure [82]. As pointed above, it is also necessary to assume a lowpass bandwidth for the drift signal in [82]. On the contrary, we do not assume any prior bandwidth information. The neural network automatically imposes sparsity constraints on the drift signal in the transform domain during training by learning the transform domain soft thresholds. Furthermore, we do not use future samples during the DCT and Hadamard transform computations to estimate the drift signal. We use only the past and current samples of sensor measurement signal $y(t)$ to estimate the drift signal $d(t)$. As a result, we compute a real-time estimate of $d(t)$.

In this paper, we study the real-valued DCT and Hadamard transforms instead of the Discrete Fourier Transform, which is complex. The Hadamard transform is based on the Haar

wavelet transform and can be computed without performing any actual multiplication operations.

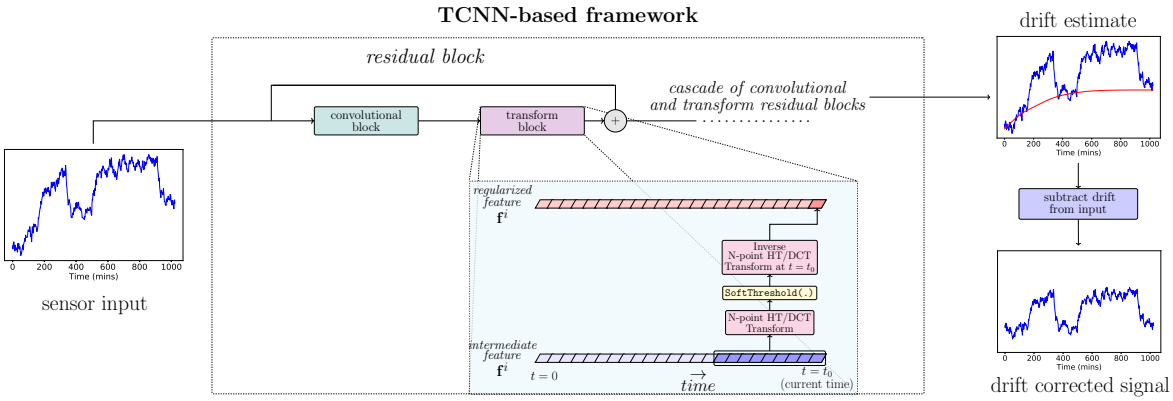


Figure 27. Block diagram of the proposed system: The system receives sensor measurements and estimates the drift using a TCNN network with transform domain layers. The TCNN network is made up of a cascade of dilated convolutional and orthogonal transform residual blocks. The system then subtracts the drift estimate from the input signal and generates the drift-corrected signal in real-time.

4.3 TCNN with Spectral Transform Domain Layers

TCNNs have been widely used in time-series related tasks [90]. Typically, TCNNs are made up of successive blocks of convolutional layers and residual connections. The convolutional

layers carry out one-dimensional causal convolution at different dilation rates. The so-called dilated convolution is expressed mathematically as follows:

$$y_r^a[n] := \sum_{c=0}^{C-1} \sum_{k=0}^{K-1} h^a[k, c] \times x[n - rk, c], \tag{4.1}$$

where n is the time index, c is the input channel index, a is the output feature map index, and r is the dilation rate. One block is typically made of a dilated convolutional layer, followed by a 1×1 convolutional layer, another dilated convolutional layer, and finally a residual connection between the output and the input of the block.

In this chapter, we are interested in finding a drift estimate $\hat{d}[n]$ at time $t = nT_s$ given the sensor time series $y[t]$ for $t \in \{0, T_s, 2T_s, \ldots, nT_s\}$, where T_s is the sampling period. Notice that we have a causal framework that estimates baseline drift signals. After we estimate $\hat{d}[n]$, we can obtain an estimate of the desired sensor response signal $\hat{p}[n]$ by subtracting the drift signal from $y[n]$. The TCNN structure is suitable for this causal time-series estimation task because the dilated convolution operation uses not only recent samples (short-term features), but also past samples (long-term features) to estimate the current output.

4.3.1 Transform Domain Thresholding Blocks

The sensor drift signal is a slowly varying signal without any high-frequency noise. Therefore, one needs to obtain a smooth estimate using the deep learning structure. In our case, we perform denoising and smoothing using orthogonal transforms [94, 95]. We propose novel orthonormal transform-based blocks to perform smoothing and denoising as a part of the deep

neural network. The orthonormal blocks with soft-thresholding features serve as smoothing units. We perform orthonormal transform operations on sliding causal windows to make them compatible with the online estimation task because feature parameters will be shifted one step at a time.

Let $\{f_n^i\}_{n\in\mathbb{N}}$ be the i-th feature map of a specific layer resulting from the earlier convolutional layers at time step n. The corresponding feature vector $\mathbf{f}_n^i$ is defined using a sliding window as follows:

$$\mathbf{f}_n^i := \Big[f^i[n] \;\; f^i[n-1] \;\; \ldots \;\; f^i[n-N+1] \Big]^T \tag{4.2}$$

where N is the size of the causal sliding window. We select N to be a power of 2 to take advantage of the fast $O(N\log N)$ efficiency of the Hadamard transform and the DCT. The transformed feature vector, denoted by $\mathbf{F}_n^i$, is defined as:

$$\mathbf{F}_n^i = \mathbf{W}_N \mathbf{f}_n^i \tag{4.3}$$

where $\mathbf{W}_N \in \mathbb{R}^{N\times N}$ is the transform matrix.

The Hadamard Transform (HT) is based on an orthogonal matrix defined using the recursive relation:

$$\mathbf{H}_N = \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} \otimes \mathbf{H}_{N/2} \tag{4.4}$$

with $\mathbf{H}_0 = [1]$ and $\otimes$ is the Kronecker product. The matrix $\mathbf{W}_N = \frac{1}{\sqrt{N}}\mathbf{H}_N$ is the unitary version of the orthogonal HT. It is essentially constructed from butterflies, and it does not require any

actual multiplication operation to compute the transform domain coefficients. It can also be constructed from the Haar wavelet transform [92]. It is worth mentioning that there are different ordering conventions for the Hadamard matrix. For example, the four-by-four sequency-ordered Walsh-Hadarmard transform matrix is defined as follows:

$$\mathbf{H}_4 = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 \\ 1 & -1 & 1 & -1 \end{bmatrix} \tag{4.5}$$

The first row of the matrix approximates the action of two successive half-band lowpass filtering and down-sampling operations; It generates the low-low output with an approximate bandwidth of $[0, \pi/4]$. The second row of the matrix generates the so-called low-high output with the approximate bandwidth of $[\pi/4, \pi/2]$, the third row is the high-low and the last row is the high-high with approximate bandwidths $[\pi/2, 3\pi/4], [3\pi/4, \pi]$, respectively.

In the case of DCT, $\mathbf{W}_N$ is given by the N-point type-II unitary DCT matrix, whose entries are defined as follows:

$$[\mathbf{W}_N]_{ij} = \begin{cases} \sqrt{\frac{2}{N}} \cos\left(\frac{\pi j}{2N}(2i+1)\right) & i \neq 0 \\ \sqrt{\frac{1}{N}} & i = 0 \end{cases} \tag{4.6}$$

It is well-known that the DCT approximates the Karhunen-Loeve transform when the correlation between the entries of the input vector is high [40].

Once we have values in the transform domain, we apply a soft-thresholding function to each coefficient except the DC value to obtain a sparse transform domain representation as follows:

$$\text{SoftTh}(x) = \begin{cases} x - b, & x \geq b \\ 0, & |x| \leq b \\ x + b, & x \leq -b \end{cases}, \tag{4.7}$$

where b is the soft-thresholding parameter that we learn during the training process. In practice, we do not apply the soft-thresholding function to the first $N_o = 3$ values of the DCT transformed feature parameters and the first value (DC) of the Hadamard-transformed features. This is to preserve the DC level of the drift estimate over the sliding windows, given that our ultimate task is regression. Leaving the DC value unchanged is also a common practice in denoising [94, 95]. After we obtain the thresholded coefficients in the transform domain, we compute the inverse transform and feed the resulting smoothed feature vector to the next layer of the deep neural network. We repeat the same process whenever we get a new reading from the sensor.

Unlike the work of [82], we do not assume any specific bandwidth for the low-frequency drift signal; the soft-thresholding parameters of each transform domain coefficient are automatically learned during training.

It is worth mentioning that, in this chapter, we refer the transform domain thresholding blocks as spectral-domain thresholding blocks. This is because both the Hadamard and the DCT transforms approximate the DFT.

4.3.2 Additive TCNN with Transform Domain Layers

While CNNs are very powerful, they are computationally expensive because of the large number of add-multiply operations needed for the inference phase. This becomes more of a pressing issue when deploying CNNs on embedded systems for real-time monitoring, in which case the power and computational capabilities are often limited. In our bid to reduce the computational cost of CNNs, we utilize the so-called additive (multiplication-free) layers defined in Equation 3.2. AddNets also improve the energy efficiency of the network because they perform only one multiplication per convolution operation [88]. Additive nets have been applied to many recognition tasks in fire detection, gas leak detection, and time-series prediction [3, 4, 88]. For further details, see Sec. 3.2.4. We define the multiplication-free ($\oplus$) dilated "convolution" as follows

$$y_r^a[n] := \beta \sum_{c=0}^{C-1} \sum_{k=0}^{K-1} h^a[k, c] \oplus x[n - rk, c], \tag{4.8}$$

where β is a normalization factor used to scale down the resulting $\oplus$ product. Although normalization by β requires a multiplication, it is performed once for each $y_r^a[n]$ value, as opposed to $C \times K$ multiplication operations required in regular dilated convolution defined in Equation 4.1. We set $\beta = \frac{1}{\sqrt{C \times K}}$. By selecting a number equal to the power of 2 as β, it is possible to eliminate all the multiplications in the dilated convolution defined in Equation (Equation 4.1).

4.3.3 TCNN Architecture

We combine the convolutional units, the transform domain thresholding blocks, and the residual connections to construct our TCNN. At time step n, the input to the TCNN is a vector of size M containing the sensor measurements $y[n], y[n-1], ..., y[n-M+1]$ and the output is a single value $\hat{d}[n]$, which is the estimate of the drift signal at time instant $t = nT_s$.

Our TCNN design is summarized in Figure 27 and Algorithm 3, where `Conv1d(k,r,D)` is the one-dimensional causal convolutional layer with filter size k, dilation rate r, and D output features. `Transform`(32) represents the orthogonal transformation of size 32. L is the number of convolutional blocks and we set it to 7. Each convolutional block carries out temporal convolution, followed by 1×1 convolution, and finally another temporal convolution. We apply orthogonal transforms after the 4-th block. We use the LeakyReLU with leakage factor = 0.2 as our nonlinearity throughout the TCNN.

The network has residual connections between successive blocks similar to the well-known ResNet [96] architecture, which introduced the so-called identity shortcut connection. These connections skip one or more layers. In our case, a skip connection linearly combines the input of a block, after scaling it by 0.5, to the output of that block. The next block of the network processes $F_{n-1}(x) + 0.5x$, where x represents the input to the previous block, and F_{n-1} represents the output of the previous block. The scaling by 0.5 is important so that the magnitude of the final output is stable. This is necessary since we carry out a regression task. We only apply bias in the 1×1 convolutional layers. We do not have the bias term in the transform domain.

As it can be seen from Algorithm 3, we have a bottleneck layer that maps from 64 feature channels to 32. We then map back to 64 feature channels in each block. Using 64 feature channels makes the network powerful enough while not being very computationally expensive, while the bottleneck layer (mapping to 32 channels) regularizes these features.

The number of blocks is tied to the dilation rates and the input size. Because we double the dilation rate, the number of blocks is in order of the logarithm of the input size. In our case, we have seven blocks, and the dilation rate of the last block is $2^7 = 128$, which means the filters of the last block can look back in time by this amount (128) multiplied by the filter size. The actual "effective" length will still be larger, given the contribution from earlier blocks. Increasing the dilation rate after some point will have the effective filter size larger than the input size. This means that early filter coefficients will lie outside the input support.

The orthogonal transform can be either the Hadamard transform or the DCT, depending on the network. It is applied after the 4th round of convolutions. This is because the size of the features exceeds the transform block size of 32 after the 4th round of successive convolutions. "Soft -thresholding" operations have threshold values that are learned during training. We do not apply thresholds to the DC value of the transforms to maintain continuity between the blocks, just like in image and audio processing.

Algorithm 3: Pseudocode of the design of TCNN. `Conv1D`(k,r,D) is a convolutional layer of filters of sizes k, dilation rates r, and outputs D feature maps.

```
Out = Conv1D(3,1,64)(Input)
OutRes = LeakyReLU(Out)
for idx ∈ {1, 2, ..., L} do
    r = 2^idx Out = Conv1D(3,r,64)(OutRes)
    Out = LeakyReLU(Out)
    Out = Conv1D(1,1,32)(Out)
    Out = LeakyReLU(Out)
    Out = Conv1D(3,r,64)(Out)
    Out = LeakyReLU(Out) if idx>= 4 then
        Out = Transform(32)(Out)
        Out = SoftThresholding(Out)
        Out = InverseTransform(32)(Out)
    end
    if idx> 1 then
        OutRes = 1/2 Out + 1/2 OutRes
        else
        end
        OutRes = Out
    end
end
Out = Conv1D (1,1,1)(OutRes)
return Out
```

As mentioned earlier, we train the TCNNs to find a drift estimate $\hat{d}[n]$. In order to do so, we minimize the following regularized cost function:

$$\begin{aligned}\mathcal{J} := \sum_{n=0}^{N-1} \big(d[n] - \hat{d}[n]\big)^2 + \lambda \sum_{n=1}^{N-1} |\hat{d}[n] - \hat{d}[n-1]| \\ - \gamma \sum_{i=0}^{I-1} \sum_{j=0}^{J-1} \log b_{ij}\end{aligned} \tag{4.9}$$

where the first term in Equation 4.9 is the reconstruction square errors, and the second term is the Total Variation (TV) regularization term [97, 98], which also imposes smoothness on the drift estimates because it minimizes the difference between the neighboring samples. The last term is a log penalty for the soft-thresholding parameters b_{ij} for layer i and channel j. This term is required to make sure that the threshold parameters are pushed away from zero. We use synthetically generated data using Equation 4.21 and Equation 4.22 to train the networks, as detailed in Sec. 4.6.

4.4 Min Operator: Multiplication-Free Kernel ℓ_1 Based-Operator

The good performance of addNet motivates us to study other energy-efficient dot-product-like operators. In this section, we discuss another ℓ_1-norm-inducing operator and some of its properties. Let $\mathbf{x} \in \mathbb{R}^N$ and $\mathbf{w} \in \mathbb{R}^N$, we define the min-operator as follows

$$\mathbf{x} \text{ⓜ} \mathbf{w} := \sum_{i=1}^{N} \operatorname{sgn}(x_i w_i) \min(|x_i|, |w_i|). \tag{4.10}$$

This operator induces the ℓ_1 norm because

$$\mathbf{x} \textcircled{m} \mathbf{x} = ||\mathbf{x}||_1 \tag{4.11}$$

Furthermore, just like the AddNet operator, it does not require any multiplication.

Another important property of the operator defined in Equation 4.10 is that it induces a kernel, i.e., for any arbitrary vector $\mathbf{x} \in \mathbb{R}^N$ we have

$$\sum_i \sum_j c_i c_j K(x_i, x_j) \geq 0 \quad \forall c_i \in \mathbb{R}, \tag{4.12}$$

which, according to Mercer's theorem, means that we can write the Kernel as an inner product in a Reproducing-Kernel Hilbert Space (RKHS) [99]

$$K(x, w) =< \Phi(x), \Phi(w) >= \sum_{i=1}^{\infty} \phi_i(x)\phi_i(w), \tag{4.13}$$

where the basis $\{\phi_i\}_{i \in \mathbb{N}}$ are orthogonal.

We show that the operator defined in Equation 4.10 induces a kernel. To do so we need the following theorems

Theorem 7 (Schur product theorem)**.** *[100] Let* $\mathbf{X}$ *and* $\mathbf{Y} \in \mathbb{R}^{N \times N}$ *be two positive semi-definite matrices, then their Hadamard product* $(\mathbf{A} \times \mathbf{B})_{ij} := \mathbf{A}_{ij}\mathbf{B}_{ij}$ *is also positive semi-definite.*

Theorem 8. *[101] if* $K_1(x, y)$ *and* $K_2(x, y)$ *are both kernels, then the product* $K(x, y) = K_1(x, y)K_2(x, y)$ *is also a kernel.*

Lemma 9 (Schur Compliment Criterion for Positive Semi-Definite Matrices)**.** *[102] Let* $\mathbf{M}$ *be a symmetric matrix of the form* $\mathbf{M} = \begin{bmatrix} \mathbf{A} & \mathbf{B} \\ \mathbf{B}^T & \mathbf{C} \end{bmatrix}$*, then* $\mathbf{M}$ *is positive-definite if and only if both* $\mathbf{C}$ *and* $\mathbf{A} - \mathbf{B}^T\mathbf{C}^{-1}\mathbf{B}$ *are both positive definite.*

Our main result is the following

Corollary 9.1. $\mathbf{x}\textcircled{m}\mathbf{w} := \sum_{i=1}^{N} sgn(x_i w_i)\min(|x_i|, |w_i|)$ *is a mercer kernel.*

Proof. Let $\mathbf{x}$ be a vector in $\mathbb{R}^N$ such that $x_1 \geq x_2 \geq x_3 \ldots x_N \geq 0$. We will show by induction that $\mathbf{x}\textcircled{m}\mathbf{x}^T$ is positive definite.

The case when $N = 1$ is trivial to prove since $x\textcircled{m}x^T = x \geq 0$.

Let $\mathbf{x}\textcircled{m}\mathbf{x}^T$ for $\mathbf{x} \in \mathbb{R}^N$ as defined above be positive semi-definite. Define $\mathbf{A} := \begin{bmatrix} \mathbf{x} \\ x_{N+1} \end{bmatrix} \textcircled{m} \begin{bmatrix} \mathbf{x} & x_{N+1} \end{bmatrix} \in \mathbb{R}^{(N+1)\times(N+1)}$. We will now show that the matrix $\mathbf{A}$ is positive semi-definite.

$$
\begin{aligned}
\mathbf{A} &= \begin{bmatrix} \mathbf{x} \\ x_{N+1} \end{bmatrix} \text{ⓜ} \begin{bmatrix} \mathbf{x} & x_{N+1} \end{bmatrix} \\
&= \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_{N+1} \end{bmatrix} \text{ⓜ} \begin{bmatrix} x_1 & x_2 & \dots & x_{N+1} \end{bmatrix} \\
&= \begin{bmatrix} x_1 & x_2 & x_3 & \dots & x_{N+1} \\ x_2 & x_2 & x_3 & \dots & x_{N+1} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ x_{N+1} & x_{N+1} & x_{N+1} & \dots & x_{N+1} \end{bmatrix}, \\
&= \begin{bmatrix} \mathbf{x}\text{ⓜ}\mathbf{x}^T & x_{N+1}\mathbf{1} \\ x_{N+1}\mathbf{1}^T & x_{N+1} \end{bmatrix}
\end{aligned}
\tag{4.14}
$$

where $\mathbf{1} \in \mathbb{R}^N$ is an all-one vector. if $x_{N+1} = 0$, then then matrix $\mathbf{A}$ becomes

$$
\mathbf{A} = \begin{bmatrix} \mathbf{x}\text{ⓜ}\mathbf{x}^T & \mathbf{0} \\ \mathbf{0} & 0 \end{bmatrix}, \tag{4.15}
$$

which is obviously positive semi-definite, since the extra zeros do not contribute to the quadratic form $\mathbf{z}^T\mathbf{A}\mathbf{z}, \quad \forall z \in \mathbb{R}^{N+1}$. When $x_{N+1} > 0$, then we need to show that the Schur compliment $\mathbf{x}\textcircled{m}\mathbf{x}^T - x_{N+1}\mathbf{1}[x_{N+1}]^{-1}x_{N+1}\mathbf{1}^T = \mathbf{x}\textcircled{m}\mathbf{x}^T - x_{N+1}\mathbf{1}\mathbf{1}^T$ is positive semi-definite.

$$\begin{aligned}
\mathbf{x}\textcircled{m}\mathbf{x}^T - x_{N+1}\mathbf{1}\mathbf{1}^T &= \begin{bmatrix} x_1 - x_{N+1} & x_2 - x_{N+1} & x_3 - x_{N+1} & \dots & x_N - x_{N+1} \\ x_2 - x_{N+1} & x_2 - x_{N+1} & x_3 - x_{N+1} & \dots & x_N - x_{N+1} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ x_N - x_{N+1} & x_N - x_{N+1} & x_N - x_{N+1} & \dots & x_N - x_{N+1} \end{bmatrix} \\
&= \begin{bmatrix} x_1 - x_{N+1} \\ x_2 - x_{N+1} \\ \vdots \\ x_N - x_{N+1} \end{bmatrix} \textcircled{m} \begin{bmatrix} x_1 - x_{N+1} & x_2 - x_{N+1} & \dots & x_N - x_{N+1} \end{bmatrix} \\
&= \mathbf{x}'\textcircled{m}\mathbf{x}'^T
\end{aligned} \tag{4.16}$$

The vector $\mathbf{x}' \in \mathbb{R}^N$. We have $x'_1 \geq x'_2 \geq \cdots \geq x'_N$. Furthermore, it is non-negative because $x_{N+1} \leq x_N$. By the induction assumption, $\mathbf{x}'\textcircled{m}\mathbf{x}$ is positive semi-definite. Therefore, the matrix $\mathbf{A}$ is positive semi-definite. For any vector $\mathbf{y} \in \mathbb{R}^N$ with positive entries, one can permute it into a vector $\mathbf{x}$ with descending entries. Therefore

$$\begin{aligned}
\mathbf{y}\textcircled{m}\mathbf{y}^T &= [\mathbf{P}x]\textcircled{m}[\mathbf{P}x]^T \\
&= \mathbf{P}(\mathbf{x}\textcircled{m}\mathbf{x}^T)\mathbf{P}^T
\end{aligned} \tag{4.17}$$

Given that $\mathbf{x}\textcircled{m}\mathbf{x}^T$ is positive semi-definite. It admits the eigen decomposition $\mathbf{x}\textcircled{m}\mathbf{x}^T = \mathbf{U}\mathbf{\Sigma}\mathbf{U}^T$, where $\mathbf{U}$ is an orthonomral matrix, and $\mathbf{\Sigma}$ is a diagonal matrix with non-negative entries. Therefore $\mathbf{y}\textcircled{m}\mathbf{y}^T = \mathbf{P}\mathbf{U}\mathbf{\Sigma}\mathbf{U}^T\mathbf{P}^T$. The matrix $\mathbf{PU}$ is orthonormal since the permutation matrix is orthonomral. This proves that for any non-negative vector $\mathbf{y} \in \mathbb{R}^N$. $\mathbf{y}\textcircled{m}\mathbf{y}^T$ is positive semi-definite.

To proof 9.1 for any real vector $\mathbf{x}$, observe that

$$\mathbf{x}\textcircled{m}\mathbf{x}^T = \begin{bmatrix} |x_1| & \operatorname{sgn}(x_1x_2)\min(|x_1|,|x_2|) & \dots & \operatorname{sgn}(x_1x_N)\min(|x_1|,|x_N|) \\ \operatorname{sgn}(x_1x_2)\min(|x_1|,|x_2|) & |x_2| & \dots & \operatorname{sgn}(x_2x_N)\min(|x_2|,|x_N|) \\ \vdots & \vdots & \ddots & \vdots \\ \operatorname{sgn}(x_1x_N)\min(|x_1|,|x_N|) & \operatorname{sgn}(x_2x_N)\min(|x_2|,|x_N|) & \dots & |x_N| \end{bmatrix}$$
$$= (\operatorname{sgn}(\mathbf{x})\operatorname{sgn}(\mathbf{x})^T) \circ (|\mathbf{x}|\textcircled{m}|\mathbf{x}|^T) \tag{4.18}$$

The first term $\operatorname{sgn}(\mathbf{x})\operatorname{sgn}(\mathbf{x})^T$ is a rank-one positive semi-definite matrix. The second term $|\mathbf{x}|\textcircled{m}|\mathbf{x}|^T$ is a positive semi-definite matrix, as shown above. Therefore, according to 7, the element wise product $(\operatorname{sgn}(\mathbf{x})\operatorname{sgn}(\mathbf{x})^T) \circ (|\mathbf{x}|\textcircled{m}|\mathbf{x}|^T) = \mathbf{x}\textcircled{m}\mathbf{x}^T$ is also positive definite. This establishes that the operator $\textcircled{m}$ defines a kernel. □

4.5 Unsupervised Sensor Drift Estimation Using Min-Op PCA and Discrete Cosine Transform

In this section, we study the problem of unsupervised drift estimation using multiple sensor measurements. Here we use the dataset from the sensor array simultaneously and use the fact

that the desired (drift-corrected) signal is strongly correlated across the different sensor measurements, while the drift signals are slowly varying. We investigate two variants of Principal Component Analysis (PCA): Linear PCA, and min-operator-based kernel PCA. Mathematically speaking, given the sensor array measurements $\mathbf{Y} \in \mathbb{R}^{T\times N}$, where N is the number of sensors, and T is the temporal size. We are interested in estimating the drift signals $\mathbf{D} \in \mathbb{R}^{T\times N}$ and the drift-corrected signals $\mathbf{X} \in \mathbb{R}^{T\times N}$. Let T be the type-2 dct transform. As in Chapter 2, we define bandlimitedness in the dct domain. Let $\mathcal{C} = \{x \in \mathbb{R}^N |\ X[i] = 0 \quad i \geq \text{BW}\}$ be the subspace of bandlimited one-dimensional signal parametrized by bandwidth parameter BW, where X is the dct transform of signal x. We pose the problem as follows

$$\begin{aligned} \underset{\mathbf{D},\mathbf{X}}{\text{minimize}} \quad & ||\mathbf{Y} - (\mathbf{X} + \mathbf{D})||_F^2 \\ \text{subject to} \quad & \mathbf{d}_i \in \mathcal{C} \quad i \in \{1, 2, \ldots, N\} \\ & \text{rank}(\mathbf{X}) = R \end{aligned} \tag{4.19}$$

The parameter $R < N$ is equal to the number of principal components we wish to use (project onto). We propose the following iterative algorithm in Algorithm 4 to solve the problem posed in Equation 4.19. The algorithm first starts with initial guesses for both $\mathbf{X}$ and $\mathbf{D}$. The algorithm then minimizes the square-error $||\mathbf{Y} - (\mathbf{X} + \mathbf{D})||_F^2$ by performing one-step gradient-descent update (Lamdweber iteration) (line 3-4). Afterwards, the algorithm projects the drift estimates, i.e., the column vectors of $\mathbf{D}$ onto the subspace of $\mathcal{C}$ (line 5-7). Afterwards, the algorithm calculates the leading R principal components for the matrix $\mathbf{X}$, and then projects

the columns of $\mathbf{X}$ onto the subspace spanned the these principal components. The algorithm repeats until it converges.

Algorithm 4: Proposed algorithm for separating the drift and the desired signals from sensor measurements from sensor array. α is the update rate. $PCA(., R)$ calculates and returns the leading R principal components

initialize $\mathbf{X}, \mathbf{D}$
while *not convergent* **do**
  $\mathbf{X} = \mathbf{X} + \alpha(\mathbf{Y} - [\mathbf{X} + \mathbf{D}])$
  $\mathbf{D} = \mathbf{D} + \alpha(\mathbf{Y} - [\mathbf{X} + \mathbf{D}])$
  $\hat{\mathbf{D}} = \mathbf{T}\mathbf{D}$
  $[\hat{\mathbf{D}}]_{ij} = \begin{cases} [\hat{\mathbf{D}}]_{ij} & i < \text{BW} \\ 0, & \text{otherwise} \end{cases}$
  $\mathbf{D} = \mathbf{T}^{-1}\hat{\mathbf{D}}$
  $\mathbf{U} = \text{PCA}(\mathbf{X}, R)$
  $\mathbf{X} = \mathbf{U}\mathbf{U}^T\mathbf{X}$
end

We also inspected another variant of the algorithm in which we perform Kernel PCA, with the min operator defined in Equation 4.10. in other words, the covariance matrix used is :

$$K(\mathbf{X}, \mathbf{X}) = \mathbf{X} \textcircled{m} \mathbf{X}^T, \tag{4.20}$$

where $\textcircled{m}$ is defined in Equation 4.10.

4.6 Experimental ans Simulation Results

4.6.1 Datasets

In our experiments, we used data from Electronic-nose (E-nose) sensors used for air quality monitoring. This dataset is collected by the Jet Propulsion Lab (JPL) [103] using 32 different carbon-polymer sensors. We use data from 16 sensors for the experimental evaluation of the proposed technique.

We also collected our own data using the commercially available MQ-137 ammonia sensor, which has a detection range of 5-500 ppm [104].

Some sensor recording examples are shown in Figure 28 and Figure 29. In what follows, we refer to this data set as JPL sensor data. We considered recordings corresponding to single-gas (methanol) exposure experiments as in [82]. In E-nose sensors, the sensor reacts with the vapor upon contact with its surface. Once the sensor is no longer exposed to the gas vapor, it starts exuding the vapor it had absorbed earlier. Absorbing and exuding the vapor depends on the sensor material, the analyte type, and the environment.

The ideal sensor response in the absence of drift can be approximated analytically as follows [105]

$$p(t) = \begin{cases} 0 & t \leq T_s \\ \beta\tau \tan^{-1}\left(\frac{t-T_s}{\tau}\right) & T_s \leq t \leq T_s + \Delta T \\ \beta\tau\left[\tan^{-1}\left(\frac{t-T_s}{\tau}\right) - \tan^{-1}\left(\frac{t-T_s-\Delta T}{\tau}\right)\right] & t \geq T_s + \Delta T \end{cases}, \tag{4.21}$$

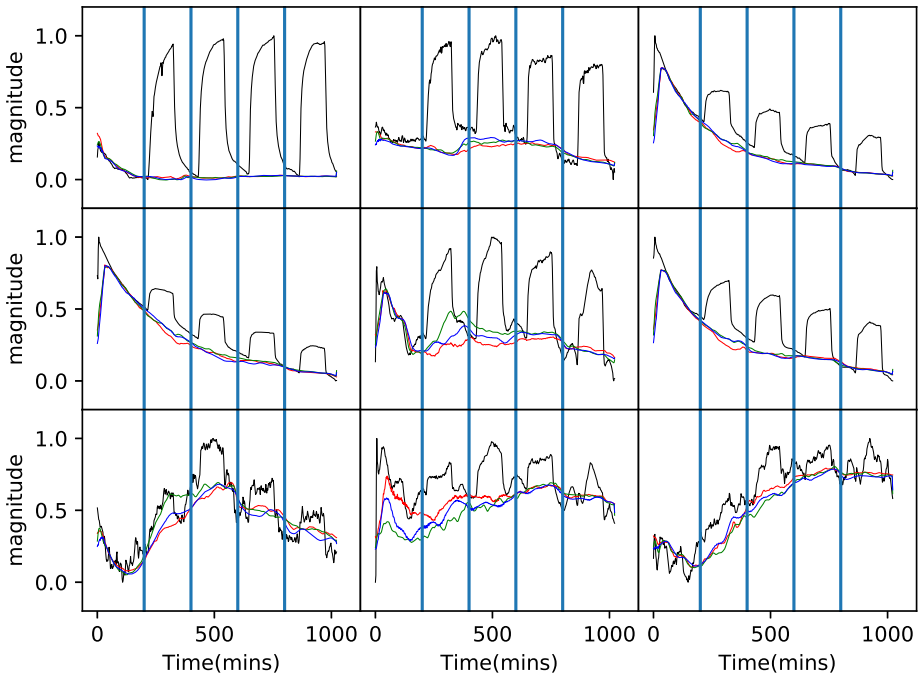


Figure 28. Test examples from the JPL data set (in black) and the estimated drift from JPL data set using regular (multiplicative) TCNNs with no spectral thresholding (in red), with DCT based layers (in green), and with HT based layers (in blue). Vertical lines mark the beginnings of gas excitations.

where T_s is the starting time of the exposure, and ΔT is the exposure duration.

The sensors are exposed to the gas vapor after 200, 400, 600, and 800 minutes in Figure 28 and the exposure duration is about 150 minutes.

The sensor responses more or less obey the model described in Equation (Equation 4.21), in the first two rows of Figure 28. Even in some of these sensors, it may not be possible to set a threshold without estimating the drift waveform to detect the VOC gas because the drift

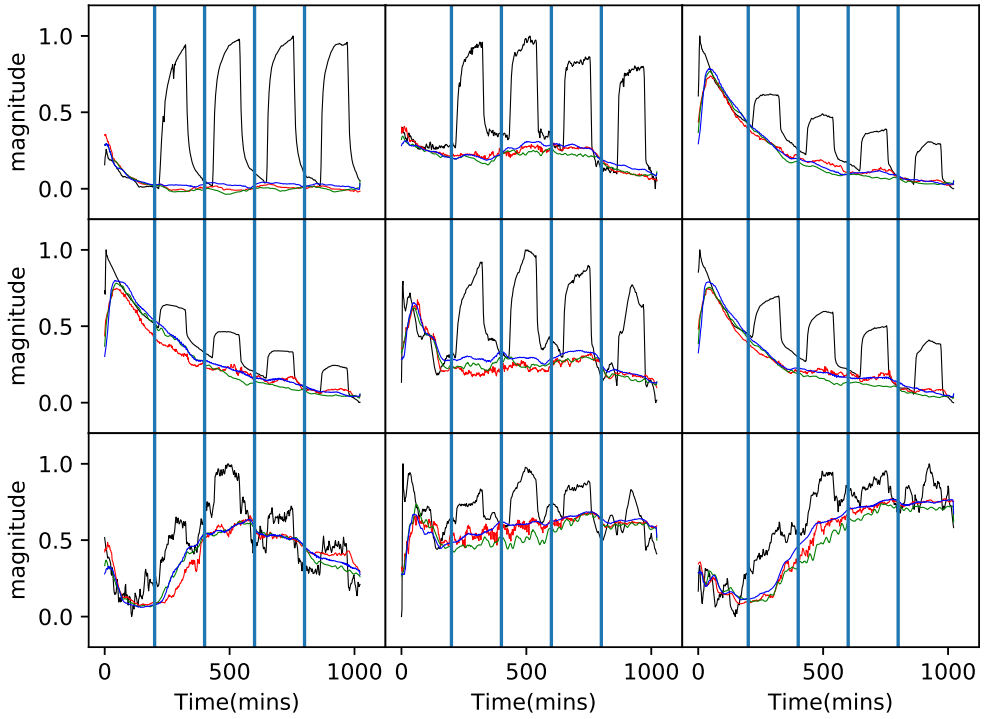


Figure 29. Test examples from the JPL data set (in black) and the estimated drift using additive (multiplication-free) TCNNs with no spectral thresholding (in red), with DCT based layers (in green), and with HT based layers (in blue).

waveform is decaying. In the last row of Figure 28, there are "noninformative" sensors, and the pulses due to gas are irregular and noisy.

Three typical sensor recordings obtained from the ammonia sensors are shown in Figure 30 and Figure 31.

We refer to this data set as the Ammonia data set. The real E-nose data was used as our test set. We trained and validated our TCNN models on synthetic data that we created. The

drift-correct signal modelling is based on Equation 4.21. We created a total of 10,000 time series with the different levels of gas exposure duration. The length of the time series is resampled to 512 samples.

In our synthetic data, we randomized the starting time and the end time of each gas exposure session and the parameters β and τ in synthetic training data. Furthermore, to generate slowly varying drift signal, we sampled data from a Gaussian process of mean μ and the covariance given by:

$$\mathrm{Cov}\big(x(t_1), x(t_2)\big) = \exp\Big(\frac{(t_1 - t_2)^2}{\sigma^2}\Big), \tag{4.22}$$

where σ^2 is a hyperparameter controlling how strongly correlated the samples of the realizations are. The process is locally smooth, and by selecting a large σ^2 value, we can generate slowly varying realizations of the random process. In particular we select σ^2 to be equal to either 2048, 4096 or 8192. The mean value μ is also chosen randomly. Afterward, the synthetic sensor measurements are created by adding the synthetic drift signal and signals created using Equation 4.21. Finally, we add zero-mean white noise with various standard deviation levels to the training waveforms.

We trained our networks using the synthetic training data for 80 epochs. We employed early stopping over our validation data set, starting from the tenth epoch. We used mini-batches of size 32. We used Adam optimizer with a learning rate equal to 10^{-3}, $\beta_1 = 0.9$, and $\beta_2 = 0.99$. We set the TV parameter λ in Equation (Equation 4.9) to 0.1 and the log penalty parameter γ to 10^{-3}.

Our numerical results over the JPL data set and the Ammonia data set are shown in Algorithm XIV and Algorithm XV, respectively. We also show the drift estimates for some examples in Figure 28 for baseline TCNN, TCNN with Hadamard Transform (HT) thresholding layers, and TCNN with DCT thresholding layers. The drift estimates for the additive neural networks are shown in Figure 29.

As it can be seen from results in Figure 28 and Figure 29, the TCNN structures with Hadamard Transform (HT) and DCT layers can accurately estimate the drift. The TCNN with the HT layer achieves the lowest average MSE in both regular and additive systems in the JPL data set.

The sensors in the 3rd row of Figure 28 and Figure 29 generate very noisy data. In all cases, the methane gas was applied to the sensors after roughly 250 min four times. The DCT-based network overfits the data compared to the regular TCNN and TCNN with HT layer in one case.

In JPL sensor 12 (shown in the first column, third row in Figure 28), the TCNN with the DCT layer misses the first pulse (the green curve). The network follows the sensor measurements very closely. As a result, this methane gas exposure cannot be detected by thresholding the difference between the sensor measurement and the estimated drift signal because they follow the baseline drift level of the sensor. On the other hand, the TCNN with HT layer (blue) and regular TCNN (red) can detect the first pulse due to methane exposure because they generate drift curves well below the sensor measurement data. Therefore, our results in Figure 28 and Figure 29 indicate that degraded sensors can be used for methane detection when the sensor data is processed using the TCNN with HT layers in real-time.

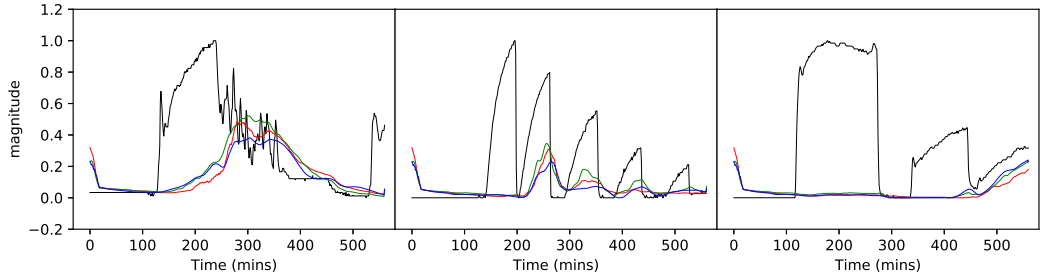


Figure 30. Results over the Ammonia data set obtained by the three TCNN models. The original signal is in black. The red-colored signals are obtained by the baseline TCNN, while the green- and blue-colored are obtained by the TCNNs with DCT and HT layers, respectively.

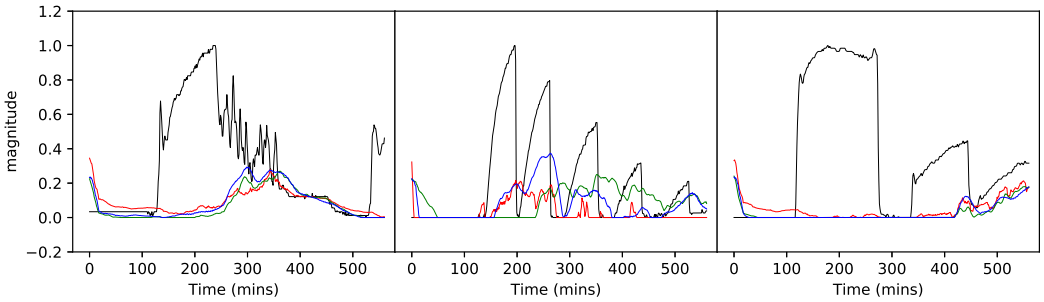


Figure 31. Results over the Ammonia data set obtained by the three additive TCNN models. The original signal is in black. The red-colored signals are obtained by the baseline TCNN, while the green- and blue-colored are obtained by the TCNNs with DCT and HT layers, respectively.

In the second set of experiments, we used the Ammonia data set. Our sensors are new, and we do not observe any baseline drift in the Ammonia data set. Since they are not degraded sensors the baseline level is zero as shown in Figure 30 and Figure 31. The proposed TCNN-

based algorithms should not generate false curves in this data set. All of the network models more or less follow the zero baseline level.

The results for the Ammonia data set are shown in Figure 30 and Figure 31 corresponding to regular and additive TCNNs, respectively. It is possible to detect all the ammonia gas pulses because the drift signals are well below the pulse levels except for the additive network with the DCT transform block in the middle signal. Although the overall MSE values of the regular networks are lower than the additive networks- as shown in Algorithm XV, the estimated drift levels of the regular networks are very close to zero whenever the gas excitation occurs in all the three sensors, as shown in Figure 30. The additive TCNN with HT and DCT layers yields inferior results compared to the regular networks in the middle plot. They may miss the 5th gas exposure in the middle plot of Figure 31. Only the regular additive network without any transforms (red curve) can detect the 5th pulse of the middle experiment in Figure 31.

The TCNN with the HT layer generates the lowest MSE among the three regular network models, as shown in Algorithm XV.

The additive networks do not produce as good results as the regular networks. Nevertheless, they can be used in low-cost processor embedded systems, such as IoT devices.

The sensor at the bottom right of Figure 28 and Figure 29 is a "noninformative" sensor. After subtracting the estimated drift waveform, one can clearly detect the gas exposure, as shown in the demonstrative figure (Figure 27). We can set up thresholds to detect gas leaks because the sensor signal has an approximate zero baseline level.

It is worth mentioning that the additive network replaces the regular dot-product operations used in convolutions with a correlation operation related to the ℓ_1 norm. As a result, intermediate features obtained in additive network learning turn out to be different from those in traditional networks. This is reflected in our finding that the additive network can achieve better performance on some of the examples in terms of the MSE. We also compared the waveforms using the ℓ_1 norm-based Mean Absolute Error (MAE). It turns out that the average MAE results of the regular (additive) network with DCT and Hadamard layers are 0.07 (0.06) and 0.06 (0.06), respectively[1].

4.6.2 Comparison with Papoulis-Gerchberg (PG) Algorithm-Based Method

Huang *et al.* developed a PG algorithm-based method for sensor drift estimation method as described in Sec. 4.2 [82]. We compared our TCNN-based networks with the method introduced in [82] using the same JPL data set. In our case, we assume that we only know that there is no gas exposure in the initial portions of the sensor recording and an upper bound on the duration of the gas exposure. This is an advantage of our method compared to the PG algorithm-based method because we do not need to know when the gas exposure starts and ends. In [82], it is assumed they know the exact time that the gas exposure ends, which is not a realistic assumption for a typical gas leak detection application in an open field or inside a building [6, 106].

---

[1]We did not include the MAE results per example in order not to overcrowd the tables.

Next, we present a summary of the PG method; Let $x[n]$ be the desired discrete-time signal for $n \in \{0, \ldots, N-1\}$. let $I[n \in \mathcal{S}]$ be the indicator function which is equal to 1 if the predicate is true and zero otherwise. Let $\mathcal{S}$ be the set of "available" samples, which is the first 100 samples out of 512 in JPL sensor recordings. The PG algorithm aims at reconstructing all $x[n]$ values from $x[n]I[n \in \mathcal{S}]$ through the following iteration:

$$\hat{x}^i[n] = x[n] \times I[n \in \mathcal{S}] + \hat{y}^i[n] \times (1 - I[n \in \mathcal{S}]) \tag{4.23}$$

where

$$\hat{y}^i[n] = \mathcal{F}^{-1}[P(e^{j\omega})\hat{X}^i(e^{j\omega})] \tag{4.24}$$

where $\hat{X}^i(e^{j\omega})$ is the Discrete-Time Fourier transform (DTFT) of $\hat{x}^i[n]$, $P(e^{j\omega})$ is an ideal low-pass filter performing band-limiting operation, $\mathcal{F}^{-1}$ is the inverse-DTFT operator and i is the iteration index. The algorithm is globally convergent regardless of the initial estimate [30].

One weakness of the PG algorithm is that one has to know the bandwidth of the desired signal *a priori*, which is not possible in general. In addition, the algorithm is very sensitive to the noise present in the available portion of the data [32, 95]. It generates high mean-square-error results, as shown in the last three columns of Algorithm XIV, especially for noisy sensor signals shown in the third row of Figure 28.

To estimate the drift signals, we implemented the algorithm using the Discrete Fourier Transform (DFT) of size 4096. We tried three different bandwidth choices with the following cut-off frequencies: 8/4096, 16/4096, and 24/4096. The MSE results are presented in the last

three columns of Algorithm XIV. As it can be seen from the table, the PG-based algorithm does not produce as good results as the deep learning-based TCNN structures. Furthermore, the PG algorithm is very sensitive to the quality of the samples. We conclude that the PG algorithm is not suitable for gas leak detection. However, it can be useful when used as an interpolator as in [82].

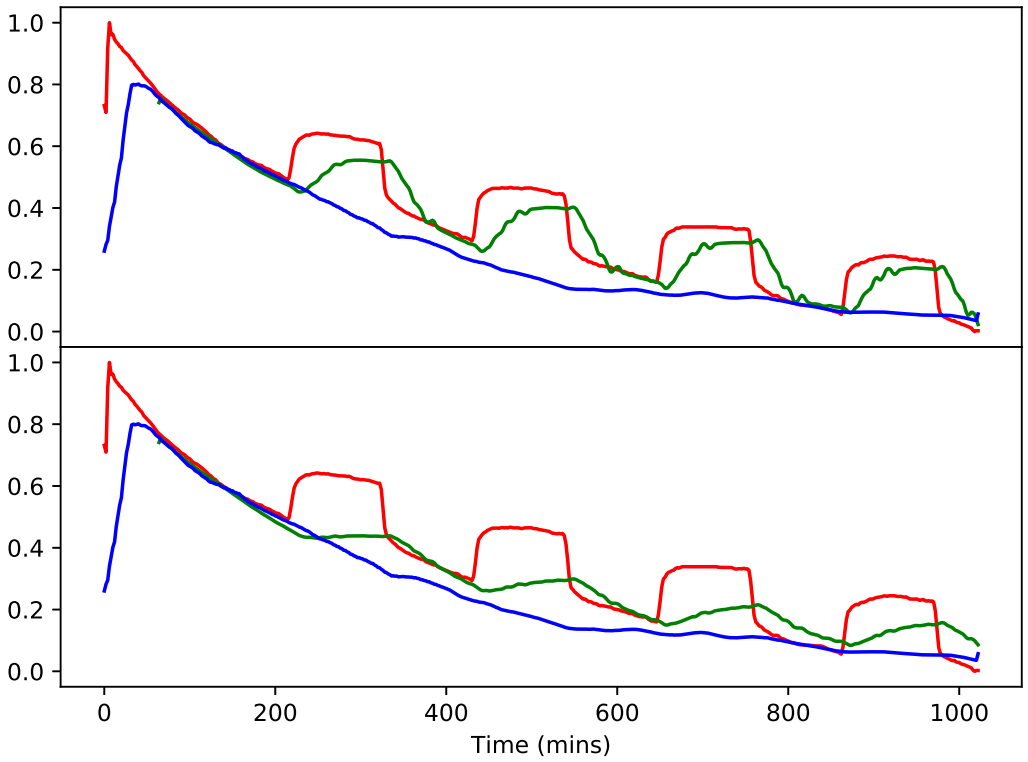


Figure 32. JPL Sensor 5: Original Signal (in red) and the predicted drift using a MLP predictor (in green), with $\epsilon = 0.1$(top), and 0.7(bottom), compared to the drift estimate of the HT based TCNN (in blue).

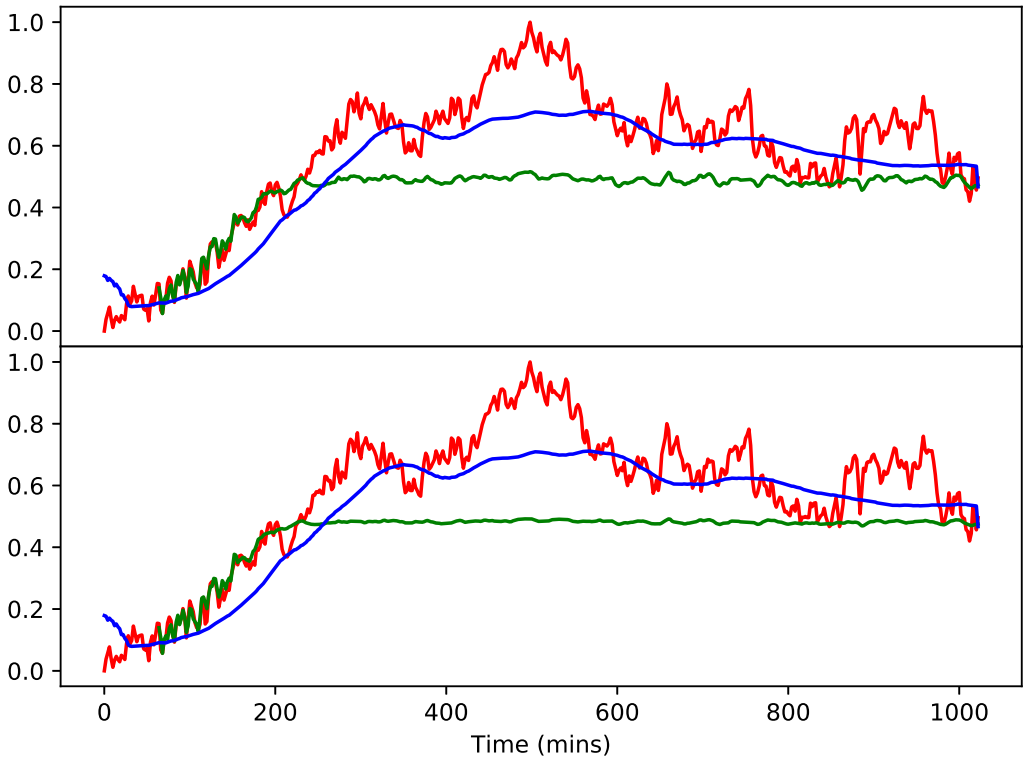


Figure 33. JPL Sensor 13: Original Signal (in red) and the predicted drift using a MLP predictor (in green), with $\epsilon = 0.1$(top), and 0.7(bottom), compared to the drift estimate of the HT based TCNN (in blue).

4.6.3 Comparison with Shallow Multi-Layer Perceptron-Based Predictor

We also compared the deep learning-based TCNN methods with the shallow neural network. In [107], a Radial-Basis Function (RBF) neural network is trained to predict the sensor drift using the current and past samples.

We also implemented a shallow multi-layer-perceptron (MLP) neural network. In [107], the authors gathered long time series of recordings of three different chemical sensors. The recordings correspond to baseline drift signals, and the first portion of each time-series are used to train a shallow one-step RBF neural network predictor.

In the JPL dataset, the initial portions of the time series sensor data correspond to the baseline sensor drift.

$$\hat{d}[n] = f(y[n-1], y[n-2], \ldots, y[n-N+1]), \tag{4.25}$$

where $f(.)$ represents the neural network function, N is the number of past samples used in the nonlinear neural network-based predictor, $y[n] = d[n] + p[n]$ is the augmented input signal at time n containing the underlying drift $d[n]$ and the added pulse signal. We then trained the neural network to minimize the MSE between $d[n]$ and $\hat{d}[n]$ over the training data. Once the MSE converges, we infer the drift over the remaining portion of the signal using the following expression:

$$\hat{d}[n] = \epsilon f(y[n-1], \ldots y[n-N+1]) + (1-\epsilon)\hat{d}[n-1], \tag{4.26}$$

where $0 < \epsilon \leq 1$ is a memory factor . We use ϵ to account for the fact that the pulse signal might last longer than the predictor order N, in which case the rule in Equation 4.25 has no means of distinguishing the drift from the additive signal $p[n]$.

We trained a one-hidden neural network with input size $N = 16$ with the hyperbolic tangent activation function and a hidden layer of size 32. The results of two examples from the JPL

data set are shown in Figure 32 and Figure 33. In Figure 32, one sees that it is possible to get a reasonable drift prediction when using an appropriate smoothing factor ϵ. In contrast, in Figure 33, the initial portion of the drift does not look similar to the remaining portion, resulting in a total failure of the shallow predictor in generating any meaningful drift estimate. This is in contrast to our proposed approach, in which the network can learn all sorts of slowly varying signals during the training because the TCNN is deep with many layers, which can "store" and "learn" the large training set of possible slowly varying signals.

4.6.4 Results of Usupervised Sensor Drift Estimation via Min Operator KPCA on the JPL Dataset

We used Algorithm 4 to estimate the drift from the 16 JPL sensor measurements. We excluded one example (JPL 10) because the sensor measurement is noninformative. We have $T = 512$, and $N = 15$. For estimating the drift signals $\mathbf{D}$, we set our DCT bandwidth parameter BW is set to 6. We used $\alpha = 0.01$. We investigated different numbers of principal components R for estimating the drift-corrected signals $\mathbf{X}$. We compared the estimated drift signal with the manually estimated drift signals (ground-truth). Our numerical results are summarized in Algorithm XVI.

As it can be seen from Algorithm XVI, the Min-operator KPCA with one principal component yield the best results for the drift estimation. It achieves a MSE of 1.69, compared to the case of the linear PCA with a single component. On the other hand, using two principal components for estimating $\mathbf{X}$ results in poor estimations of the drift, in case of using linear PCA. Nevertheless, using the min-operator kernel PCA with two principal components to estimate $\mathbf{X}$

results in good drift signal estimation, albeit not as good as in using one principal component.

Figure 34 shows the drift estimates of 12 sensors.

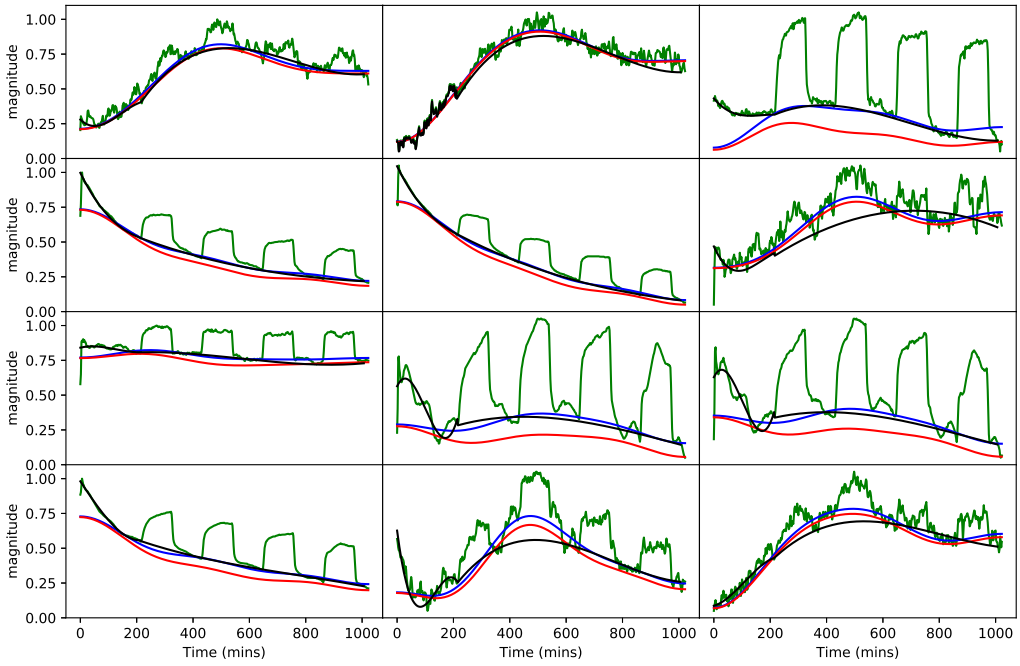


Figure 34. Results of the drift estimates using the unsupervised PCA-DCT approach. The black signals correspond to the manually estimated drift. The blue signals correspond to the drift estimates using Min-op kernel PCA (R=1), and the red signals correspond to the drift estimates using linear PCA (R=1).

Example	Regular Network			Additive Network			PG Algorithm ($\frac{f_{\text{cutoff}}}{4096}$)		
	None	DCT	HT	None	DCT	HT	8	16	24
JPL 0	3.41	4.59	**2.52**	4.17	2.98	4.59	12.42	11.27	5.96
JPL 1	**0.75**	0.98	0.97	1.12	1.09	1.20	2.82	4.48	7.54
JPL 2	1.02	0.97	**0.78**	1.10	1.02	1.23	2.74	6.20	8.85
JPL 3	2.27	**1.52**	1.74	1.55	1.81	1.91	2.21	5.87	7.45
JPL 4	0.75	**0.40**	0.52	0.44	0.88	0.54	13.55	7.15	5.78
JPL 5	1.02	**0.50**	0.73	1.35	0.70	0.34	16.56	9.32	6.04
JPL 6	**4.41**	5.21	5.93	3.02	2.40	2.34	11.43	13.52	20.13
JPL 7	2.39	1.87	**1.56**	2.86	1.65	3.91	4.43	1.86	7.28
JPL 8	1.52	2.45	**1.25**	2.03	1.47	1.15	5.56	10.13	12.47
JPL 9	1.96	3.08	**1.55**	2.10	1.02	1.21	2.51	7.89	11.43
JPL 11	1.11	**0.47**	0.60	1.11	1.49	0.74	19.51	12.33	10.92
JPL 12	5.23	8.27	**4.37**	7.17	3.89	4.90	15.71	12.60	13.86
JPL 13	5.62	8.65	**5.01**	5.88	5.86	6.34	21.23	12.44	11.29
JPL 14	**1.39**	4.14	2.53	1.35	2.99	1.45	7.47	9.81	11.23
JPL 15	3.11	3.92	**2.25**	6.58	8.24	4.49	12.82	8.15	10.77
Average (MSE)	2.63	3.22	**2.35**	3.04	3.20	**2.60**	8.46	8.84	11.62
Average (MAE)	0.07	0.07	0.06	0.07	0.06	0.06	0.23	0.25	0.30

TABLE XIV

MEAN SQUARE ERROR (MSE) OVER THE JPL DATA SET FOR SIX TCNN MODELS AND THE PAPOULIS-GERCHBERG (PG) ALGORITHM: THREE REGULAR AND THREE ADDITIVE (MULTIPLICATION-FREE) MODEL. WE IMPLEMENTED THREE MODELS USING THE REGULAR TCNN AND ADDITIVE TCNN, RESPECTIVELY: "NONE" MEANS REGULAR TCNN, THE SECOND AND THE 3RD COLUMNS REFER TO DCT AND HT BASED TCNNS, RESPECTIVELY. THE LAST THREE COLUMNS REPORT THE MSE RESULTS OF THE PG ALGORITHM FOR DIFFERENT BANDWIDTH SELECTION. THE NUMBERS 8,16, AND 24 CORRESPOND TO THE CUTOFF FREQUENCY INDEX FOR A DFT OF SIZE 4096 USED IN PG ALGORITHM. WE ALSO SHOW THE AVERAGE MEAN-ABSOLUTE ERROR FOR THE DIFFERENT ALGORITHMS USED.

Example	Regular Network			Additive Network		
	None	DCT	HT	None	DCT	HT
Ammonia 1	22.30	28.32	17.95	5.10	10.22	12.85
Ammonia 2	2.99	4.34	2.06	2.12	3.56	4.23
Ammonia 3	1.00	1.86	2.18	2.07	2.70	3.06
Average (MSE)	8.67	11.51	**7.39**	**3.09**	5.49	6.71
Average (MAE)	**0.08**	0.10	0.09	**0.08**	0.10	0.12

TABLE XV

MEAN SQUARE ERROR (MSE) OVER THE AMMONIA DATA SET FOR SIX TCNN MODELS: THREE REGULAR AND THREE ADDITIVE (MULTIPLICATION-FREE) MODEL. "NONE" MEANS A REGULAR TCNN STRUCTURE (WITHOUT TRANSFORM DOMAIN LAYERS). WE ALSO REPORT THE AVERAGE MEAN-ABSOLUTE ERROR FOR THE DIFFERENT MODELS USED.

Example	Linear PCA		Min-op KPCA	
	R=1	R=2	R=1	R=2
JPL 0	**0.56**	18.8	0.79	1.0
JPL 1	0.75	4.23	**0.74**	0.79
JPL 2	**0.66**	4.05	0.70	0.75
JPL 3	7.11	13.8	**5.7**	7.7
JPL 4	2.0	10.6	**1.54**	2.3
JPL 5	1.2	9.06	**0.8**	1.0
JPL 6	1.3	4.29	2.08	**0.86**
JPL 7	**0.8**	1.9	0.93	1.55
JPL 8	3.1	6.1	**1.6**	4.0
JPL 9	3.31	6.5	**2.8**	4.4
JPL 11	3.32	11.3	**1.8**	2.6
JPL 12	1.5	**1.2**	2.7	3.4
JPL 13	**0.52**	8.3	0.96	1.2
JPL 14	0.79	1.3	0.59	**0.52**
JPL 15	**0.54**	9.1	0.77	1.0
Average (MSE)	1.83	7.36	**1.69**	2.29

TABLE XVI

MEAN SQUARE ERROR (MSE) OF THE DRIFT ESTIMATION USING UNSUPERVISED PCA AND DCT. WITH TWO TYPES OF PCA: LINEAR AND MIN-OPERATOR KERNEL PCA.

4.7 Conclusion

In this chapter, we introduced a novel deep learning-based framework for sensor drift estimation. The proposed TCNN structures have built-in Hadamard and additive transform-based layers, which smooth and denoise the input signal because the sensor drift signal is a slowly varying signal. The Hadamard transform is related to the Haar wavelet transform, and it can be implemented without performing any real multiplication operations. The Hadamard transform can be implemented using binary operations. Similarly, the proposed additive layers can be implemented using only binary operations.

We experimentally observed that the Hadamard transform performs better than the DCT, which approximates the Karhunen-Loeve Transform, in our dataset. The DCT is used in signal, image, and data compression, and it fits the input very efficiently. However, it overfits the data, which are feature parameters of the previous layer in our case. Since our goal is smoothing and denoising, the Haar wavelet transform-based Hadamard transform is more suitable than DCT. Other wavelet transforms can also be used in this problem. Nevertheless, they are not as computationally efficient as the Hadamard transform.

We also introduced a novel multiplication-free ℓ_1-inducing operator. We show that this operator can be used to define a mercer-type kernel. We used this kernel to perform unsupervised drift estimation using DCT and PCA. Our results indicate that the kernel PCA achieves better drift estimation that linear PCA.

CHAPTER 5

DETECTING ANOMALY IN CHEMICAL SENSORS VIA REGULARIZED CONTRASTIVE LEARNING

The content of this chapter is based on our work that was published in IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), 2022, under the title "Detecting Anomaly in Chemical Sensors via Regularized Contrastive Learning" © 2022 IEEE [10].

5.1 Introduction

Chemical sensory technology provides cheap and mobile solutions to detecting and identifying different gas analytes. Chemical sensors are widely used in ammonia, methane, and Volatile Organic Compound (VOC) detection, which are known to be carcinogenic and main contributors to the greenhouse effect [1, 3, 6, 8, 18, 48–50].

One of the main challenges facing chemical sensors technology is the fact that sensor responses vary significantly due to process variations and in-field degradation, which is dubbed sensor drift. Sensor drift can arise due to internal factors which result in a low-frequency change in a sensor response and due to external reasons such as changes in humidity and temperature. This makes detecting anomalous behavior in the sensory system crucial for reliable gas identification, and concentration estimation [19, 78, 108–111]. In this work, we focus on detecting anomalies in sensor behavior due to external factors in a system consisting of multiple sensors. In [109], the authors propose an online method to detect anomalous changes in wireless sensor

measurements by fitting piecewise linear models for the time-series data and comparing them with reference signals. If the absolute differences are larger than a threshold, an anomaly is declared. However, with large sensor-to-sensor variation, such consensus methods will likely fail to identify the failing sensors and introduce unnecessary uncertainty into the decision-making process.

Deep learning architectures such as recurrent neural networks (RNN) and temporal convolution neural networks (TCNN) have been quite popular in learning tasks involving time-series data and anomaly detection [111]. In [112], the authors propose using deep autoencoders to detect sudden changes in a sensor time series in wireless sensor networks. One type of learning, namely contrastive learning, has proven effective in learning pretext tasks that are useful when there not enough labeled data [113–115] is available. In standard contrastive learning, the objective is to learn representations with maximum agreement among data samples of the same concept while having a minimal agreement with data samples from other concepts. Many frameworks have been recently developed such as unsupervised non-parametric Instance discrimination (InstDisc) [113], Momentum Contrast (MoCo) [114], and Simple Contrastive Learning (SimCLR) [115]. In InstDisc, the authors try to maximize the contrast between individual instances across the entire data set.

In SimCLR, the authors sample a subset of data points and duplicate each data point by applying an augmentation transform. The objective therein is to maximize the similarity between the data points of each pair against the remaining data points. Unlike previous methods, Sim-

CLR is simple to implement as it avoids the usage of memory banks used in other paradigms. Furthermore, SimCLR achieves state-of-the-art results on ImageNet with a linear classifier.

Motivated by these recent advances, we aim to leverage a deep contrastive framework for detecting an outlier sensor in a sensor array system based on the temporal responses of the sensors. Recent work that applies contrastive learning to detect outliers includes Novelty Detection via Contrastive Learning [116] on Distributionally Shifted Instances (CSI) [116]. In this framework, the authors apply so-called shifted transforms to create OOD samples. Like in InstDisc, the authors try to minimize similarity between the original instance and its OOD counterparts.

In this work, we have actual sensor measurements rather than relying on data augmentation transforms to learn similarity/dissimilarity. We modify the standard multi-view contrastive loss criterion to encourage learning representations that are similar among in-distribution samples, i.e., good sensor time series. Simultaneously, we want to learn contrasted representations for outlier time series, i.e., degraded sensor time series. In contrastive learning, the cost function uses the cosine similarity measure. We also study an ℓ_1 norm-based similarity measure as a part of contrastive learning for anomalous sensors.

5.2 Organization

The organization of this chapter is as follows: In Sec. 5.3 we present our anomaly detection framework. In Sec. 5.4 we discuss our dataset and present our experimental results. In Sec. 5.5 we provide our conclusion.

5.3 Anomaly Detection Network

In standard multi-view contrastive learning, a random mini-batch of size N is sampled, and each sample is augmented by a set of data augmentation transforms. This yields a total of $2N$-sample bag. Let $g(z, w)$ be a similarity score (e.g., the cosine similarity measure) between the representation codes z and w. Let (z_i, z_j) be a pair of samples, where z_i is an image from the mini-batch, and z_j represents its augmented version. In [115], the following loss function is minimized:

$$\mathcal{L}_{i,j} := -\log \frac{\exp\big(g(z_i, z_j)/\tau\big)}{\sum_{k=1}^{2N} \mathbb{I}_{i \neq k} \exp\big(g(z_i, z_k)/\tau\big)}, \tag{5.1}$$

where $\mathbb{I}$ is an indicator function, and $\tau > 0$ is the temperature hyper-parameter controlling how strong the contrast should be.

5.3.1 Outlier-Modified Contrastive Loss

The loss function in Equation 5.1 is not suitable for anomalous sensor detection problem. This is because the loss in Equation 5.1 tries to maximize the similarity between one data instance and its corresponding augmented version one at a time. This is in contrast to our objective to maximize similarity among all in-distribution samples, while maximizing the dissimilarity between the in-distribution and the outlier samples.

Given a set of in-lier sensor measurements $\mathcal{X} = \{x_1, x_2, \dots x_n\}$ and an outlier $\{x^{n+1}\}$. Let $z = f(x)$ be the corresponding features extracted by a neural network. Define p_{ij} for $i \in \{1, 2, ..., n\}$ and $j \in \{1, 2, ..., n+1\}$ as follows

$$p_{ij} = \frac{\exp\big(g(z_i, z_j)/\tau\big)}{\sum_{k=1}^{N} \exp\big(g(z_i, z_k)/\tau\big) + \exp(g(z_i, z^{n+1})/\tau)} \tag{5.2}$$

Our goal is to have $p_{i,n+1} = 0$ while maximizing the entropy of the in-liers, i.e., minimizing the following objective

$$\mathcal{L}_i = \sum_{l=1,\ \ l\neq i}^{N} p_{il} \log p_{il} \tag{5.3}$$

where $p_{i,n+1}$ represents the softmax score between an in-lier x_i and the outlier x^{n+1}. In our framework, we minimize the unconstrained modification loss criterion defined as follow

$$\mathcal{L}_i = \sum_{l=1, l\neq i}^{N} p_{il} \log p_{il} - \alpha \log(1 - p_{i,n+1}), \tag{5.4}$$

where $\alpha > 0$. The first term measures the disparity between sample x_i and all other in-lier instances in the mini-batch. The second term measures the repulsion between sample x_i and the outlier sample x^{n+1}. If $p_{i,n+1}$ approaches to 1, the second term grows to infinity. On the other, hand, when $p_{i,n+1}$ approaches zero, as desired, the second term vanishes in Equation 5.4. By controlling the Lagrange multiplier α, we can emphasize learning invariance among the in-lier samples, in the case of small alpha. For a large value of α, we put more emphasis on

differentiating the samples x_i from the outlier point x^{n+1}. The overall minibatch loss is given by

$$\mathcal{L} := \frac{1}{N} \sum_{i=1}^{n} \mathcal{L}_i \tag{5.5}$$

The loss function defined in Equation 5.4 can be easily modified to accommodate multiple outliers. Let $\{x^{n+1}, x^{n+2}, ..., x^{n+k}\}$ be a set containing k outliers, then the corresponding loss function becomes:

$$\mathcal{L}_i = \sum_{l=1,\ \ l\neq i}^{n} p_{il} \log p_{il} - \alpha \sum_{j=n+1}^{n+k} \log(1 - p_{i,j}), \tag{5.6}$$

5.3.2 Kernel-Based Cosine Similarity Metric

The cosine similarity metric is widely used to measure the similarity between representations in contrastive learning. The correlation coefficient or the cosine similarity is bounded between -1 and +1. While cosine similarity is the most common similarity metric, one can devise a kernel-based similarity metric as follows

$$\text{sim}(\mathbf{x}, \mathbf{y}) = \frac{K(\mathbf{x}, \mathbf{y})}{\sqrt{K(\mathbf{x}, \mathbf{x}) K(\mathbf{y}, \mathbf{y})}} \tag{5.7}$$

The metric defined in Equation 5.7 ranges between -1 and +1. This follows immediately from the fact that a Mercer-type (positive semi-definite kernel) can be written as $k(\mathbf{x}, \mathbf{y}) =< \phi(\mathbf{x}, \phi(\mathbf{y}) >$ for some function ϕ, using Cauchy-Swartz inequality, we obtain

$$K(\mathbf{x}, \mathbf{y}) =< \phi(\mathbf{x}), \phi(\mathbf{y}) >\leq ||\phi(\mathbf{x})||_2||\phi(\mathbf{y}))||_2 = \sqrt{K(\mathbf{x}, \mathbf{x})K(\mathbf{y}, \mathbf{y})} \tag{5.8}$$

Motivated by the fact that the min operator $\mathbf{x}\textcircled{m}\mathbf{y}$ defined in Equation 4.10 induces a Mercer-type kernel [117], as shown in Sec. 4.4, we define a kernel cosine similarity metric as follows

$$g_{\textcircled{m}}(\mathbf{z}, \mathbf{w}) = \frac{\mathbf{z}\textcircled{m}\mathbf{w}}{\sqrt{||\mathbf{z}||_1||\mathbf{w}||_1}}, \tag{5.9}$$

where $||.||_1$ is the ℓ_1 norm. Furthermore, we also try another quasi-cosine metric based on the $\textcircled{m}$ operator defined as follows

$$g_{\textcircled{m}}(\mathbf{z}, \mathbf{w}) = \frac{\mathbf{z}\textcircled{m}\mathbf{w}}{\min(||\mathbf{z}||_1||\mathbf{w}||_1)} \tag{5.10}$$

The second variant has the advantage of giving a similarity metric equals to +1(-1) when $\mathbf{z} = \alpha\mathbf{w}$ where α is a positive(negative) scalar. We try these two $\textcircled{m}$-based quasi cosine metrics in our framework, i.e., replacing the usual cosine similarity metric $g(.,.)$ in our calculation of the probability scores in Equation 5.2 in both training the contrastive model and carrying out inference.

5.3.3 Inference Phase

During inference, we are presented with a patch of N time series collected by the N sensors in our multi-sensor system. We extract features $z_i = f(x_i)$, where x_i is the i-th time series and $f(.)$ is our neural network. Once we have extracted a feature z_i, we find its "distance" scores from the remaining features z_j for $j \neq i$. The notion of distance score is closely related to the similarity metric. We define it to be

$$d(z_i, z_j) := \frac{1 - g(z_i, z_j)}{2} \tag{5.11}$$

where $g(.,.)$ is the cosine similarity metric or the metrics defined in Equation 5.9 or Equation 5.10. As one can see, if the similarity score is close to 1, $d(.,.)$ will be close to zero, and if the similarity score is close to -1, $d(.,.)$ will be close to one. Given all distance scores between feature vector z_i and the remaining features, we now define our anomaly score $S(z_i)$ as follows

$$S(z_i) := \text{Median}\{d(z_i, z_j) : j \neq i\} \tag{5.12}$$

Note that for a patch of N time series, one needs to calculate $\frac{N(N-1)}{2}$ distance scores. If the feature-extracting model has been trained properly, one should be able to find a threshold $T \in (0, 1)$ such that if $s(z) = s\big(f(x)\big) \geq T$, the corresponding sample x should be declared out-of-distribution.

Input	Operator	Filter size	Channels	dilation rate
384×1	Conv1d	3	16	1
384×16	Conv1d	1	32	1
384×32	Conv1d	3	16	1
384×16	Conv1d	3	16	2
384×16	Conv1d	1	32	1
384×32	Conv1d	3	16	2
384×16	Conv1d	3	16	4
384×16	Conv1d	1	32	1
384×32	Conv1d	3	16	4
384×16	Conv1d	3	16	8
384×16	Conv1d	1	32	1
384×32	Conv1d	3	16	8
384×16	Conv2d	1	1	1

TABLE XVII

TEMPORAL CONVOLUTIONAL NEURAL NETWORK ARCHITECTURE. EACH BLOCK (EXCEPT THE FINAL LAYER), CONSIST OF 3 LAYERS. THERE ARE RESIDUAL (SKIP) CONNECTIONS BETWEEN CONSECUTIVE BLOCKS.

5.3.4 Feature Extraction Deep Network

We train a temporal convolution neural network with four convolutional blocks, each comprising three convolutional layers. The first and the last layers implement causal dilated convolution, while the middle layer implements 1×1 convolution. There is skip (residual) connections between these blocks. The dilation rates are $\{1, 2, 4, 8\}$. Finally, we have a 1×1 convolutional layer, and the output is a 1-dimensional representation z that has the same length as the input x. The details of our model are summarized in Algorithm XVII.

5.4 Experiments and Results

The commercial sensors used for data collection were MQ137 which are Tin oxide(SnO2) based sensors [47]. When heated and exposed to air, the sensors react with the oxygen present in the air and form a layer of negative ions on the surface and reduce the surface conductivity [47]. However, when gases such as Ammonia come in contact with the surface, they combine with the oxide ion layer on the top and release electrons for conduction, increasing the surface conductivity. This change in surface resistance can be measured in the form of voltage.

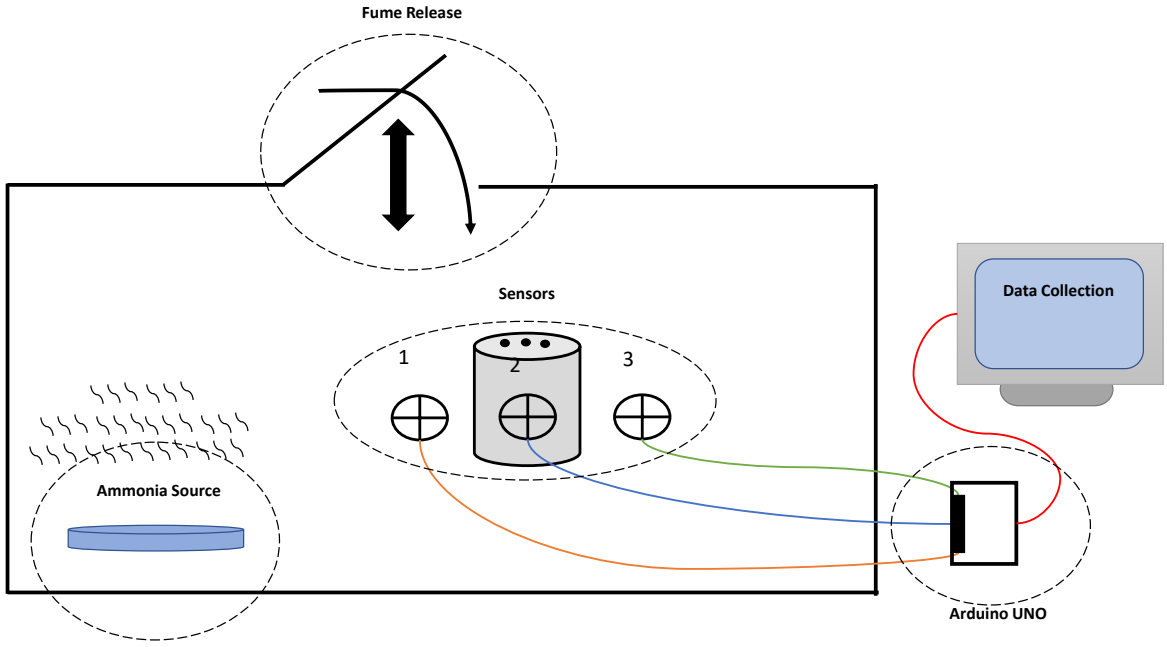


Figure 35. Illustration of our experimental setup.

5.4.1 Experiment Setup and Data acquisition

For the experiment, three MQ137 sensors were used. The data was recorded using an Arduino UNO board. The experimental set-up is illustrated in Figure 35. The three sensors are placed in an airtight chamber. The pre-heating duration of the sensors is 48 hours, during which time no ammonia is introduced into the chamber. A cylindrical ammonia source (commercial low concentrations ammonia cleaner liquid) is placed in the chamber. The lid of the ammonia cylinder is removed to release ammonia gas into the chamber slowly. The ammonia then starts to leak up into the closed environment, gradually building up in the air. Since the chamber is small ($<$ 10 Liter in volume), the concentration of ammonia homogenizes quickly. The sensor responses change accordingly. Based on the proximity of the sensors to one another, they should read the same ammonia concentration. One of the sensors is obstructed by a cylindrical cover with multiple holes. The covering of one of the sensors causes it to react more slowly to the ammonia build-up and release. Therefore, by inducing discrepancies in the sensor response with respect to the other unblocked sensors, we created outlier sensor measurements. The obstruction level of the outlier sensor is modified from experiment to experiment to avoid overfitting to one condition. The chamber lid is opened at random intervals and at random duration to generate a more realistic environment with varying ammonia concentration levels. We repeatedly opened and closed the lid to create different rise and fall responses.

5.4.2 Anomaly Detection Example

In the training phase, we segment our time series into segments of 384 units. We then pre-process each segment by subtracting its mean and dividing by its ℓ_2 norm. This is to eliminate

Model	AUC score
Deep Contrastive Model (Cosine Similarity)	93,6%
Deep Contrastive Model (Min-operator Similarity [Equation 5.9])	91.9%
Contrastive Model (Min-operator Similarity [Equation 5.10])	88,1%
Cosine Similarity (No Feature Extraction)	90,1%
Min-operator Similarity (No Feature Extraction [Equation 5.9])	90,0%
Min-operator Similarity (No Feature Extraction [Equation 5.10])	89,4%
Absolute Difference [109] (No Feature Extraction)	77,6%

TABLE XVIII

AREA UNDER CURVATURE (AUC) FOR VARIOUS METHODS. THE DEEP CONTRASTIVE LEARNING FRAMEWORK PROVIDES THE BEST AUC.

global biases among different sensors. We train a temporal convolutional neural network with the architecture described in Sec. 5.3.4. We set α in Equation 5.4 and τ in Equation 5.2 to 0.5 and 1, respectively.

We used measurements corresponding to five experiment sessions for training. Each experiment has three time series belonging to the three sensors, one of which is an outlier (or anomaly). We extracted a total of 6000 384-unit-long segments from these time series. We reserved 20% for the data for validation. We used a dataset of four experiments as our test data. During the test phase, we apply three sliding windows of size 384 on each time series coming

from the three sensors. We pre-process each segment as in training, and feed them into our neural network to extract features. We then calculate the anomaly score as described in Sec. 5.3.3. For comparison, we applied the cosine similarity and the min-operator similarity metric defined in Equation 5.10 directly on these segments without applying the temporal neural network. Furthermore, we also applied the absolute difference metric $d(z_i, z_j) := \frac{1}{T}\sum_{t=1}^{T} |z_i[t] - z_j[t]|$ used in [109] for anomaly detection. We summarize our accuracy results on the test data set in Algorithm XVIII. The best result is obtained by the contrastively learned TCNN model as shown in Algorithm XVIII. We plot the Receiver Operating Characteristic (ROC) curve in Figure 37. By examining the ROC curves, the contrastive-learning-based model can achieve a true positive rate (TPR) of 85% at a false positive rate (FPR) of 2.5%. On the other hand, the other models (the cosine-similarity and min-operator-similarity modes) cannot achieve a TPR higher than 78% at the same FPR.

In this case, the standard cosine similarity based contrastively learned TCNN produces superior results, as opposed to the ℓ_1-norm based methods defined in Equation 5.9 and Equation 5.10. We believe that in different scenarios, in which the data is contaminated by impulsive noise, the min op-related approaches could provide more robust results as demonstrated in [117].

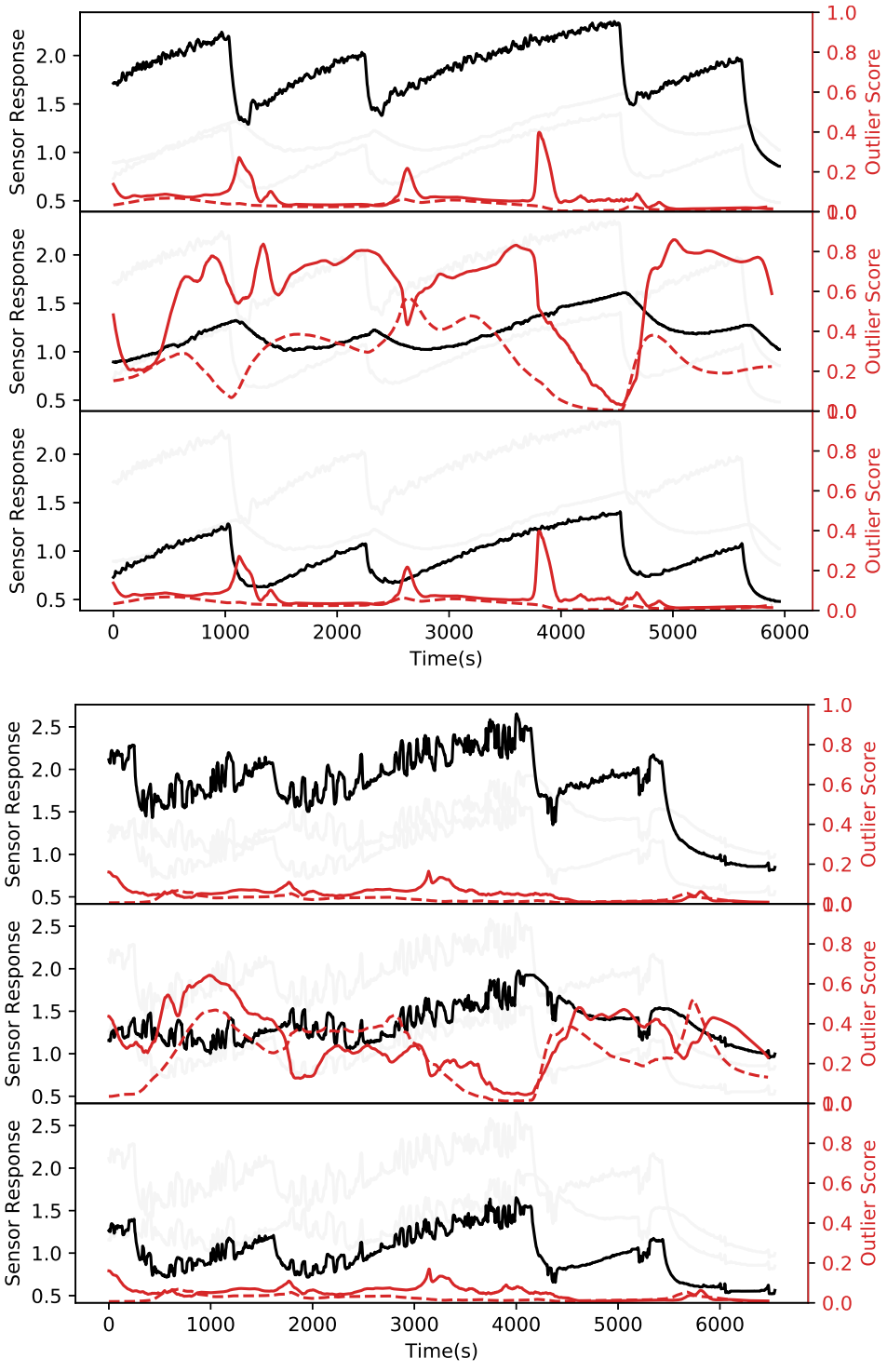


Figure 36. Outlier score results for the three sensors in two experiments used in testing. The second sensor (second rows) is the poisoned sensor. The learned-representations outlier score is in red, while the dashed red lines correspond to the outlier score with the cosine similarity metric applied directly to the input (no learning). Notice in the case of the second sensor, almost all the time the deep outlier score is significantly higher than in the baseline case. In the right experiment, both scores decrease at around time 4000 seconds. This is because the anomaly experienced at the previous discharge is no longer present in the 384-second-long segments. We consider the output then to be in-lier.

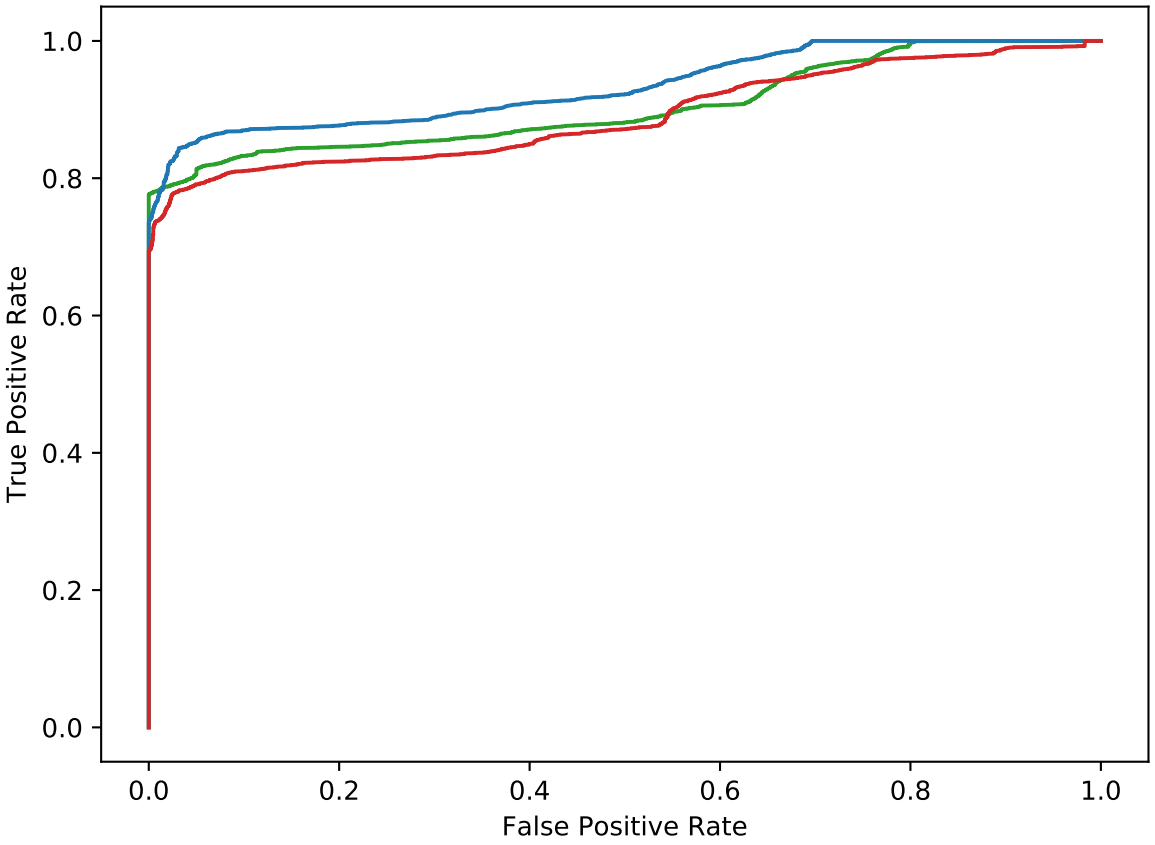


Figure 37. Receiver operating characteristic curve (ROC) for the contrastive-learning model (blue), shallow cosine-similarity-based model (green), and the shallow min-operator-based similarity (red).

5.5 Conclusion

In this chapter, we presented a framework for detecting anomalous sensor(s) in a chemical sensory system using a contrastive learning approach. In this approach, we adapt the standard multi-view contrastive learning loss function such that the model learns to maximize similarity among in-distribution samples (good sensors' readouts) while at the same time maximizing dissimilarity between the in-distribution and out-of-distribution samples. We gathered data from three commercial Tin Oxide (SnO2) sensors by exposing them to Ammonia in an environment-controlled experiment. We train a temporal CNN on 4 sets of measurements, and test it on 6 other sets. Our results show that we can identify the anomalous sensor among the three sensors with an AUC score of 93.6%, compared to 90.1% in the baseline case.

APPENDICES

Appendix A

VOC IR DATASET SOURCES

In this appendix we provide detailed lists of the different data sets used in this work. Many of source videos contain multiple scenes. Our own video clips are available on Google Drive via the following link `https://drive.google.com/open?id=1Sm0FwgUgP_j1TNBIxzUi8Pr7TvIdiuh0`.

Scene ID	Hyperlink	Contains leak?
Scene 1	`https://www.youtube.com/watch?v=G7boBAAoQPA`	No
Scene 2	`https://www.youtube.com/watch?v=XpszOB67NAI`	No
Scene 3	`https://www.youtube.com/watch?v=YU-8pg\_fFWQI`	No
Scene 4	`http://www.cantronics.com/images/stories/download/video/TR4700-Video-Bridge-Far-2010.avi`	No
Scene 5	`http://www.cantronics.com/images/stories/download/video/TR4700-Video-Bridge-Near-2010.avi`	No
Scene 6	`http://www.cantronics.com/images/stories/download/video/TR4700-Video-Road-Near-2010.avi`	No
Scene 7	`https://www.youtube.com/watch?v=yqz4-Neo0IY`	Yes
Scene 8	`https://www.youtube.com/watch?v=Zumb-sTzgHY`	Yes
Scene 9	`https://www.youtube.com/watch?v=GHXon3g2cAM`	Yes
Scene 10	`https://www.youtube.com/watch?v=FqD41AODUyY`	Yes
Scene 11	`https://www.youtube.com/watch?v=LPCte7Bn7Qw`	Yes
Scene 12	`https://www.youtube.com/watch?v=D5FjEAYsvv8`	Yes
Scene 13	`https://www.youtube.com/watch?v=S3Mzay6j1q4`	Yes

TABLE XIX

HYPERLINKS OF THE VIDEOS FROM WHICH WE OBTAINED TEMPORAL 1-D SIGNALS. HERE, WE USED ONLY ONE SCENE PER VIDEO TO EXTRACT TEMPORAL 1-D SIGNALS.

Appendix A (Continued)

Video ID	Hyperlink	Contains leak?
Scene 1-4	https://www.youtube.com/watch?v=G7boBAAoQPA	No
Scene 5-7	https://www.youtube.com/watch?v=XpszOB67NAI	No
Scene 8-9	https://www.youtube.com/watch?v=YU-8pg_fFWQ	Yes
Scene 10-11	https://www.youtube.com/watch?v=GHXon3g2cAM	Yes
Scene 12	https://www.youtube.com/watch?v=FqD41AODUyY	Yes
Scene 13-15	https://www.youtube.com/watch?v=Sqg43EeBkY8	Yes
Scene 16-20	https://www.youtube.com/watch?v=YZlnV3o4TRk	Yes
Scene 21-24	https://www.youtube.com/watch?v=lcTPioWkeCc\&t=149s	No
Scene 25	https://www.youtube.com/watch?v=4RVhik7eIzk	No
Scene 26	https://www.youtube.com/watch?v=5k8GORDmpog	No
Scene 27	https://www.youtube.com/watch?v=623iAfO8D_Q	No
Scene 28	https://www.youtube.com/watch?v=p4Q2jBVxrPw	Yes
Scene 29	https://www.youtube.com/watch?v=UeN18sEPSK0	Yes
Scene 30	https://www.youtube.com/watch?v=QZ9iPz7HCig	Yes
Scene 31	https://www.youtube.com/watch?v=B_lOx0Cc7Z4	Yes
Scene 32	https://www.youtube.com/watch?v=9PDJAEWg3Ok	Yes

TABLE XX

HYPERLINKS OF THE VIDEOS FROM WHICH WE EXTRACTED TWO DATASETS: SCENE 1-20 ARE THE DATA SET USED TO ESTABLISH THE CONFIDENCE SCORE AS IN TABLE IX. SCENE 1-20 ARE ALSO USED AS VALIDATION DATASET FOR THE SPATIO-TEMPORAL CNN AS IN TABLE XI. SCENE 21-32 ARE THE TEST DATA SET USED IN THE JOINT EVALUATION OF BOTH 1-D AND 2-D CNNS AS IN TABLE X

Appendix A (Continued)

Video ID	Hyperlink	Number of Scenes	Contains leak?
Video 1	https://www.youtube.com/watch?v=G7boBAAoQPA	15	No
Video 2	https://www.youtube.com/watch?v=Xpsz0B67NAI	5	No
Video 3	https://www.youtube.com/watch?v=YU-8pg_fFWQ	1	No
Video 4	https://www.youtube.com/watch?v=xsMMvLL0B_k	3	No
Video 5	https://www.youtube.com/watch?v=xsMMvLL0B_k	3	No
Video 6	https://www.youtube.com/watch?v=Sqg43EeBkY8	3	No
Video 7	http://www.cantronics.com/images/stories/download/video	1	No
Video 8	http://www.cantronics.com/images/stories/download/video	1	No
Video 9	http://www.cantronics.com/images/stories/download/video	1	No
Video 10	https://www.youtube.com/watch?v=yqz4-Neo0IY	1	Yes
Video 11	https://www.youtube.com/watch?v=Zumb-sTzgHY	1	Yes
Video 12	https://www.youtube.com/watch?v=GHXon3g2cAM	3	Yes
Video 13	https://www.youtube.com/watch?v=FqD41AODUyY	1	Yes
Video 14	https://www.youtube.com/watch?v=YZlnV3o4TRk	1	Yes
Video 15	https://www.youtube.com/watch?v=D5FjEAYsvv8	4	Yes
Video 16	https://www.youtube.com/watch?v=S3Mzay6j1q4	2	Yes
Video 17	https://www.youtube.com/watch?v=Z_4wa3UW-2o	2	Yes

TABLE XXI

VIDEO LINKS FOR THE SPATIO-TEMPORAL TRAINING DATA SET. THERE ARE A TOTAL OF 33 SCENES WHICH CONTAIN NO VOC-LEAK AND 15 SCENES WHICH CONTAIN VOC LEAK. THE SCENES VARY IN LENGTH.

Appendix B

COPYRIGHT PERMISSIONS

Computationally Efficient Spatio-Temporal Dynamic Texture Recognition for Volatile Organic Compound (VOC) Leakage Detection in Industrial Plants

Author: Diaa Badawi
Publication: IEEE Journal of Selected Topics in Signal Processing
Publisher: IEEE
Date: May 2020

Copyright © 2020, IEEE

Thesis / Dissertation Reuse

The IEEE does not require individuals working on a thesis to obtain a formal reuse license, however, you may print out this statement to be used as a permission grant:

Requirements to be followed when using any portion (e.g., figure, graph, table, or textual material) of an IEEE copyrighted paper in a thesis:

1) In the case of textual material (e.g., using short quotes or referring to the work within these papers) users must give full credit to the original source (author, paper, publication) followed by the IEEE copyright line © 2011 IEEE.
2) In the case of illustrations or tabular material, we require that the copyright line © [Year of original publication] IEEE appear prominently with each reprinted figure and/or table.
3) If a substantial portion of the original paper is to be used, and if you are not the senior author, also obtain the senior author's approval.

Requirements to be followed when using an entire IEEE copyrighted paper in a thesis:

1) The following IEEE copyright/ credit notice should be placed prominently in the references: © [year of original publication] IEEE. Reprinted, with permission, from [author names, paper title, IEEE publication title, and month/year of publication]
2) Only the accepted version of an IEEE copyrighted paper can be used when posting the paper or your thesis on-line.
3) In placing the thesis on the author's university website, please display the following message in a prominent place on the website: In reference to IEEE copyrighted material which is used with permission in this thesis, the IEEE does not endorse any of [university/educational entity's name goes here]'s products or services. Internal or personal use of this material is permitted. If interested in reprinting/republishing IEEE copyrighted material for advertising or promotional purposes or for creating new collective works for resale or redistribution, please go to http://www.ieee.org/publications_standards/publications/rights/rights_link.html to learn how to obtain a License from RightsLink.

If applicable, University Microfilms and/or ProQuest Library, or the Archives of Canada may supply single copies of the dissertation.

BACK CLOSE WINDOW

Appendix B (Continued)

IEEE
Requesting permission to reuse content from an IEEE publication

Real-Time Low-Cost Drift Compensation for Chemical Sensors Using a Deep Neural Network With Hadamard Transform and Additive Layers

Author: Diaa Badawi
Publication: IEEE Sensors Journal
Publisher: IEEE
Date: 15 August 2021

Copyright © 2021, IEEE

Thesis / Dissertation Reuse

The IEEE does not require individuals working on a thesis to obtain a formal reuse license, however, you may print out this statement to be used as a permission grant:

Requirements to be followed when using any portion (e.g., figure, graph, table, or textual material) of an IEEE copyrighted paper in a thesis:

1) In the case of textual material (e.g., using short quotes or referring to the work within these papers) users must give full credit to the original source (author, paper, publication) followed by the IEEE copyright line © 2011 IEEE.
2) In the case of illustrations or tabular material, we require that the copyright line © [Year of original publication] IEEE appear prominently with each reprinted figure and/or table.
3) If a substantial portion of the original paper is to be used, and if you are not the senior author, also obtain the senior author's approval.

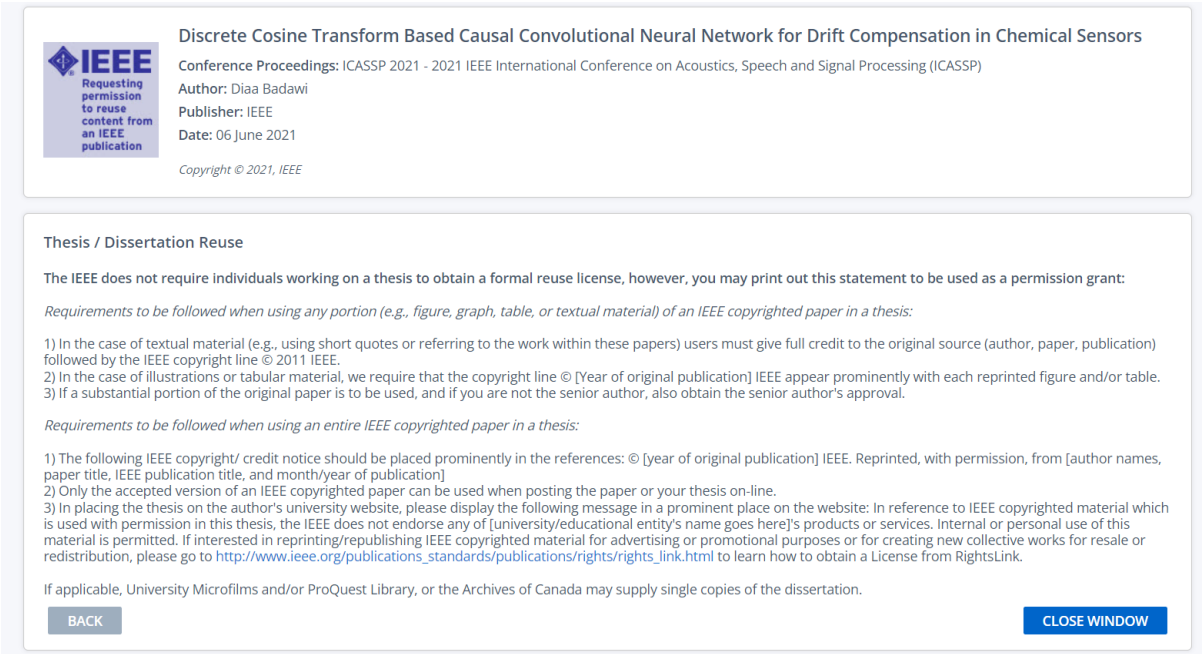
Requirements to be followed when using an entire IEEE copyrighted paper in a thesis:

1) The following IEEE copyright/ credit notice should be placed prominently in the references: © [year of original publication] IEEE. Reprinted, with permission, from [author names, paper title, IEEE publication title, and month/year of publication]
2) Only the accepted version of an IEEE copyrighted paper can be used when posting the paper or your thesis on-line.
3) In placing the thesis on the author's university website, please display the following message in a prominent place on the website: In reference to IEEE copyrighted material which is used with permission in this thesis, the IEEE does not endorse any of [university/educational entity's name goes here]'s products or services. Internal or personal use of this material is permitted. If interested in reprinting/republishing IEEE copyrighted material for advertising or promotional purposes or for creating new collective works for resale or redistribution, please go to http://www.ieee.org/publications_standards/publications/rights/rights_link.html to learn how to obtain a License from RightsLink.

If applicable, University Microfilms and/or ProQuest Library, or the Archives of Canada may supply single copies of the dissertation.

BACK CLOSE WINDOW

Appendix B (Continued)



Discrete Cosine Transform Based Causal Convolutional Neural Network for Drift Compensation in Chemical Sensors

Conference Proceedings: ICASSP 2021 - 2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)
Author: Diaa Badawi
Publisher: IEEE
Date: 06 June 2021

Copyright © 2021, IEEE

Thesis / Dissertation Reuse

The IEEE does not require individuals working on a thesis to obtain a formal reuse license, however, you may print out this statement to be used as a permission grant:

Requirements to be followed when using any portion (e.g., figure, graph, table, or textual material) of an IEEE copyrighted paper in a thesis:

1) In the case of textual material (e.g., using short quotes or referring to the work within these papers) users must give full credit to the original source (author, paper, publication) followed by the IEEE copyright line © 2011 IEEE.
2) In the case of illustrations or tabular material, we require that the copyright line © [Year of original publication] IEEE appear prominently with each reprinted figure and/or table.
3) If a substantial portion of the original paper is to be used, and if you are not the senior author, also obtain the senior author's approval.

Requirements to be followed when using an entire IEEE copyrighted paper in a thesis:

1) The following IEEE copyright/ credit notice should be placed prominently in the references: © [year of original publication] IEEE. Reprinted, with permission, from [author names, paper title, IEEE publication title, and month/year of publication]
2) Only the accepted version of an IEEE copyrighted paper can be used when posting the paper or your thesis on-line.
3) In placing the thesis on the author's university website, please display the following message in a prominent place on the website: In reference to IEEE copyrighted material which is used with permission in this thesis, the IEEE does not endorse any of [university/educational entity's name goes here]'s products or services. Internal or personal use of this material is permitted. If interested in reprinting/republishing IEEE copyrighted material for advertising or promotional purposes or for creating new collective works for resale or redistribution, please go to http://www.ieee.org/publications_standards/publications/rights/rights_link.html to learn how to obtain a License from RightsLink.

If applicable, University Microfilms and/or ProQuest Library, or the Archives of Canada may supply single copies of the dissertation.

BACK CLOSE WINDOW

Appendix B (Continued)

Detecting Anomaly in Chemical Sensors via Regularized Contrastive Learning

Conference Proceedings: ICASSP 2022 - 2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)
Author: Diaa Badawi
Publisher: IEEE
Date: 23 May 2022

Thesis / Dissertation Reuse

The IEEE does not require individuals working on a thesis to obtain a formal reuse license, however, you may print out this statement to be used as a permission grant:

Requirements to be followed when using any portion (e.g., figure, graph, table, or textual material) of an IEEE copyrighted paper in a thesis:

1) In the case of textual material (e.g., using short quotes or referring to the work within these papers) users must give full credit to the original source (author, paper, publication) followed by the IEEE copyright line © 2011 IEEE.
2) In the case of illustrations or tabular material, we require that the copyright line © [Year of original publication] IEEE appear prominently with each reprinted figure and/or table.
3) If a substantial portion of the original paper is to be used, and if you are not the senior author, also obtain the senior author's approval.

Requirements to be followed when using an entire IEEE copyrighted paper in a thesis:

1) The following IEEE copyright/ credit notice should be placed prominently in the references: © [year of original publication] IEEE. Reprinted, with permission, from [author names, paper title, IEEE publication title, and month/year of publication]
2) Only the accepted version of an IEEE copyrighted paper can be used when posting the paper or your thesis on-line.
3) In placing the thesis on the author's university website, please display the following message in a prominent place on the website: In reference to IEEE copyrighted material which is used with permission in this thesis, the IEEE does not endorse any of [university/educational entity's name goes here]'s products or services. Internal or personal use of this material is permitted. If interested in reprinting/republishing IEEE copyrighted material for advertising or promotional purposes or for creating new collective works for resale or redistribution, please go to http://www.ieee.org/publications_standards/publications/rights/rights_link.html to learn how to obtain a License from RightsLink.

If applicable, University Microfilms and/or ProQuest Library, or the Archives of Canada may supply single copies of the dissertation.

BACK CLOSE WINDOW

CITED LITERATURE

1. National Institute of Standards and Technology. Toluene. `https://webbook.nist.gov/cgi/cbook.cgi?ID=C108883&Type=IR-SPEC&Index=2#IR-SPEC`, last accessed: 2019-06-30.

2. The Oil and Gas Threat Map. Infrared video. `https://oilandgasthreatmap.com/about/infrared/`, last accessed: 2019-06-30.

3. D. Badawi, H. Pan, S. C. Cetin, and A. Enis Çetin. Computationally efficient spatio-temporal dynamic texture recognition for volatile organic compound (voc) leakage detection in industrial plants. *IEEE Journal of Selected Topics in Signal Processing*, 14(4):676–687, 2020.

4. Hongyi Pan, Diaa Badawi, Xi Zhang, and Ahmet Enis Cetin. Additive neural network for forest fire detection. *Signal, Image and Video Processing*, pages 1–8, 2019.

5. Chengmo Yang, Patrick Cronin, Agamyrat Agambayev, Sule Ozev, A Enis Cetin, and Alex Orailoglu. A crowd-based explosive detection system with two-level feedback sensor calibration. In *Proceedings of the 39th International Conference on Computer-Aided Design*, pages 1–9, 2020.

6. Diaa Badawi, Tuba Ayhan, Sule Ozev, Chengmo Yang, Alex Orailoglu, and Ahmet Enis Cetin. Detecting gas vapor leaks using uncalibrated sensors. *IEEE Access*, 7:155701–155710, 2019.

7. Diaa Badawi, Agamyrat Agambayev, Sule Ozev, and A Enis Cetin. Discrete cosine transform based causal convolutional neural network for drift compensation in chemical sensors. In *ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 8012–8016. IEEE, 2021.

8. Diaa Badawi, Agamyrat Agambayev, Sule Ozev, and A Enis Cetin. Real-time low-cost drift compensation for chemical sensors using a deep neural network with hadamard transform and additive layers. *IEEE Sensors Journal*, 2021.

9. Hongyi Pan, Diaa Badawi, and Ahmet Enis Cetin. Block walsh-hadamard transform based binary layers in deep neural networks. *arXiv preprint arXiv:2201.02711*, 2022.

CITED LITERATURE (Continued)

10. Diaa Badawi, Ishaan Bassi, Sule Ozev, and Ahmet Enis Cetin. Detecting anomaly in chemical sensors via regularized contrastive learning. In *ICASSP 2022-2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 86–90. IEEE, 2022.

11. Hongyi Pan, Diaa Badawi, Ishaan Bassi, Sule Ozev, and Ahmet Enis Cetin. Detecting anomaly in chemical sensors via l1-kernels based principal component analysis. *arXiv preprint arXiv:2201.02709*, 2022.

12. AJ Turner, Daniel J Jacob, Kevin J Wecht, Johannes D Maasakkers, E Lundgren, Arlyn E Andrews, Sebastien C Biraud, Hartmut Boesch, Kevin W Bowman, Nicholas M Deutscher, et al. Estimating global and north american methane emissions with high spatial resolution using gosat satellite data. *Atmospheric Chemistry and Physics*, 15(12):7049–7069, 2015.

13. Robert Parker, Hartmut Boesch, Austin Cogan, Annemarie Fraser, Liang Feng, Paul I Palmer, Janina Messerschmidt, Nicholas Deutscher, David WT Griffith, Justus Notholt, et al. Methane observations from the greenhouse gases observing satellite: Comparison to ground-based tccon data and model calculations. *Geophysical Research Letters*, 38(15), 2011.

14. Siraput Jongaramrungruang, Andrew K Thorpe, Georgios Matheou, and Christian Frankenberg. Methanet–an ai-driven approach to quantifying methane point-source emission from high-resolution 2-d plume imagery. *Remote Sensing of Environment*, 269:112809, 2022.

15. KC Liddiard. Thin-film resistance bolometer ir detectors. *Infrared Physics*, 24(1):57–64, 1984.

16. Herbert H Hill Jr, William F Siems, and Robert H St. Louis. Ion mobility spectrometry. *Analytical Chemistry*, 62(23):1201A–1209A, 1990.

17. Margaret Amy Ryan, Hanying Zhou, Martin G Buehler, Kenneth S Manatt, Victoria S Mowrey, Shannon P Jackson, Adam K Kisor, Abhijit V Shevade, and Margie L Homer. Monitoring space shuttle air quality using the jet propulsion laboratory electronic nose. *IEEE sensors journal*, 4(3):337–347, 2004.

18. Alexander Vergara, Shankar Vembu, Tuba Ayhan, Margaret A Ryan, Margie L Homer, and Ramón Huerta. Chemical gas sensor drift compensation using classifier ensembles. *Sensors and Actuators B: Chemical*, 166:320–329, 2012.

CITED LITERATURE (Continued)

19. Kow-Ming Chang, Chih-Tien Chang, Kuo-Yi Chao, and Chia-Hung Lin. A novel ph-dependent drift improvement method for zirconium dioxide gated ph-ion sensitive field effect transistors. *Sensors*, 10(5):4643–4654, 2010.

20. Denglong Ma, Jianqiang Deng, and Zaoxiao Zhang. Comparison and improvements of optimization methods for gas emission source identification. *Atmospheric Environment*, 81:188–198, 2013.

21. Arvind P Ravikumar, Jingfan Wang, and Adam R Brandt. Are optical gas imaging technologies effective for methane leak detection? *Environmental science & technology*, 51(1):718–724, 2017.

22. Zhang Yong, Zhang Liyi, Han Jianfeng, Ban Zhe, and Yang Yi. An indoor gas leakage source localization algorithm using distributed maximum likelihood estimation in sensor networks. *Journal of Ambient Intelligence and Humanized Computing*, 10(5):1703–1712, 2019.

23. Denglong Ma, Jianmin Gao, Zaoxiao Zhang, Hong Zhao, and Qingsheng Wang. Locating the gas leakage source in the atmosphere using the dispersion wave method. *Journal of Loss Prevention in the Process Industries*, 63:104031, 2020.

24. Porsteinn B Jónsson, Jeonghyeon Wang, and Jinwhan Kim. Scalar field reconstruction based on the gaussian process and adaptive sampling. In *2017 14th International Conference on Ubiquitous Robots and Ambient Intelligence (URAI)*, pages 442–445. IEEE, 2017.

25. Martin Asenov, Marius Rutkauskas, Derryck Reid, Kartic Subr, and Subramanian Ramamoorthy. Active localization of gas leaks using fluid simulation. *IEEE Robotics and Automation Letters*, 4(2):1776–1783, 2019.

26. Michael Hutchinson, Hyondong Oh, and Wen-Hua Chen. A review of source term estimation methods for atmospheric dispersion events using static or mobile sensors. *Information Fusion*, 36:130–148, 2017.

27. RW Gerchberg. Super-resolution through error energy reduction. *Optica Acta: International Journal of Optics*, 21(9):709–720, 1974.

28. Irina F Gorodnitsky and Bhaskar D Rao. Sparse signal reconstruction from limited data using focuss: A re-weighted minimum norm algorithm. *IEEE Transactions on signal processing*, 45(3):600–616, 1997.

CITED LITERATURE (Continued)

29. Arnold Lent and Heang Tuy. An iterative method for the extrapolation of band-limited functions. *Journal of Mathematical Analysis and Applications*, 83(2):554–565, 1981.

30. Athanasios Papoulis. A new algorithm in spectral analysis and band-limited extrapolation. *IEEE Transactions on Circuits and systems*, 22(9):735–742, 1975.

31. Dan C Youla and Heywood Webb. Image restoration by the method of convex projections: Part 1-theory. *IEEE transactions on medical imaging*, 1(2):81–94, 1982.

32. Patrick L Combettes. The foundations of set theoretic estimation. *Proceedings of the IEEE*, 81(2):182–208, 1993.

33. PL Combettes. The convex feasibility problem in image recovery. In *Advances in imaging and electron physics*, volume 95, pages 155–270. Elsevier, 1996.

34. James C Robinson. *An Introduction to Functional Analysis*. Cambridge University Press, 2020.

35. Heinz H Bauschke, Patrick L Combettes, et al. *Convex analysis and monotone operator theory in Hilbert spaces*, volume 408. Springer, 2011.

36. Alice Lucas, Michael Iliadis, Rafael Molina, and Aggelos K Katsaggelos. Using deep neural networks for inverse problems in imaging: beyond analytical methods. *IEEE Signal Processing Magazine*, 35(1):20–36, 2018.

37. Karol Gregor and Yann LeCun. Learning fast approximations of sparse coding. In *Proceedings of the 27th international conference on international conference on machine learning*, pages 399–406, 2010.

38. Steven Diamond, Vincent Sitzmann, Felix Heide, and Gordon Wetzstein. Unrolled optimization with deep priors. *arXiv preprint arXiv:1705.08041*, 2017.

39. Gregory Ongie, Ajil Jalal, Christopher A Metzler, Richard G Baraniuk, Alexandros G Dimakis, and Rebecca Willett. Deep learning techniques for inverse problems in imaging. *IEEE Journal on Selected Areas in Information Theory*, 1(1):39–56, 2020.

40. Nasir Ahmed, T_ Natarajan, and Kamisetty R Rao. Discrete cosine transform. *IEEE transactions on Computers*, 100(1):90–93, 1974.

CITED LITERATURE (Continued)

41. C Chamzas and Wen Xu. An improved version of papoulis-gerchberg algorithm on band-limited extrapolation. *IEEE transactions on acoustics, speech, and signal processing*, 32(2):437–440, 1984.

42. Ovidiu Calin. *Deep learning architectures*. Springer, 2020.

43. Zhou Wang, Eero P Simoncelli, and Alan C Bovik. Multiscale structural similarity for image quality assessment. In *The Thrity-Seventh Asilomar Conference on Signals, Systems & Computers, 2003*, volume 2, pages 1398–1402. Ieee, 2003.

44. Fisher Yu and Vladlen Koltun. Multi-scale context aggregation by dilated convolutions. *arXiv preprint arXiv:1511.07122*, 2015.

45. Bruce D Lucas, Takeo Kanade, et al. An iterative image registration technique with an application to stereo vision. Vancouver, 1981.

46. Nobuyuki Otsu. A threshold selection method from gray-level histograms. *IEEE transactions on systems, man, and cybernetics*, 9(1):62–66, 1979.

47. Joseph Watson, Kousuke Ihokura, and Gary SV Coles. The tin dioxide gas sensor. *Measurement Science and Technology*, 4(7):711, 1993.

48. United States Environmental Protection Agency. Leak detection and repair a best practices guide. `https://www.epa.gov/sites/production/files/2014-02/documents/ldarguide.pdf`. urldate: 2016-12-112. Accessed: 2019-06-30.

49. National Service Center for Environmental Publications . Inspection manual: Federal equipment leak regulations for the chemical manufacturing industry 3 VOLUME SET.

50. Fatih Erden, E Birey Soyer, B Ugur Toreyin, and A Enis Cetin. Voc gas leak detection using pyro-electric infrared sensors. In *2010 IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 1682–1685. IEEE, 2010.

51. Ahmet Enis Cetin and Behcet Ugur Toreyin. Method, device and system for determining the presence of volatile organic compounds (voc) in video, April 30 2013. US Patent 8,432,451.

CITED LITERATURE (Continued)

52. New York Times. Methane emissions soared to a record in 2021, noaa says - the new york times, 2022. `https://www.nytimes.com/2022/04/07/climate/methane-emissions-record.html`, urldate: 2022-04-07.

53. B Ugur Toreyin and A Enis Cetin. Volatile organic compound plume detection using wavelet analysis of video. In *2008 15th IEEE International Conference on Image Processing*, pages 1836–1839. IEEE, 2008.

54. B Uğur Töreyin, Yiğithan Dedeoğlu, Uğur Güdükbay, and A Enis Cetin. Computer vision based method for real-time fire and flame detection. *Pattern recognition letters*, 27(1):49–58, 2006.

55. Y Hakan Habiboglu, Osman Gunay, and A Enis Cetin. Real-time wildfire detection using correlation descriptors. In *2011 19th European Signal Processing Conference*, pages 894–898. IEEE, 2011.

56. Osman Gunay, Behçet Ugur Toreyin, Kivanc Kose, and A Enis Cetin. Entropy-functional-based online adaptive decision fusion framework with application to wildfire detection in video. *IEEE Transactions on Image Processing*, 21(5):2853–2865, 2012.

57. B Uğur Töreyin, Yiğithan Dedeoğlu, and A Enis Cetin. Wavelet based real-time smoke detection in video. In *2005 13th European Signal Processing Conference*, pages 1–4. IEEE, 2005.

58. Turgay Celik and Hasan Demirel. Fire detection in video sequences using a generic color model. *Fire Safety Journal*, 44(2):147–158, 2009.

59. Yusuf Hakan Habiboğlu, Osman Günay, and A Enis Çetin. Covariance matrix-based fire and flame detection method in video. *Machine Vision and Applications*, 23(6):1103–1113, 2012.

60. BU Toreyin, Yigithan Dedeoglu, A Enis Cetin, Sándor Fazekas, Dmitry Chetverikov, Tomer Amiaz, and Nahum Kiryati. Dynamic texture detection, segmentation and analysis. In *Conference On Image And Video Retrieval: Proceedings of the 6 th ACM international conference on Image and video retrieval*, volume 9, pages 131–134, 2007.

61. Renaud Péteri, Sándor Fazekas, and Mark J Huiskes. Dyntex: A comprehensive database of dynamic textures. *Pattern Recognition Letters*, 31(12):1627–1632, 2010.

CITED LITERATURE (Continued)

62. A Enis Cetin and Fatih Porikli. Special issue on dynamic textures in video. *Machine Vision and Applications*, 22(5):739–740, 2011.

63. B Ugur Toreyin and A Enis Cetin. Computer vision based forest fire detection. In *2008 IEEE 16th Signal Processing, Communication and Applications Conference*, pages 1–4. IEEE, 2008.

64. Turgay Celik, Hasan Demirel, Huseyin Ozkaramanli, and Mustafa Uyguroglu. Fire detection using statistical color model in video sequences. *Journal of Visual Communication and Image Representation*, 18(2):176–185, 2007.

65. A Enis Çetin, Kosmas Dimitropoulos, Benedict Gouverneur, Nikos Grammalidis, Osman Günay, Y Hakan Habiboglu, B Ugur Töreyin, and Steven Verstockt. Video fire detection–review. *Digital Signal Processing*, 23(6):1827–1843, 2013.

66. A Enis Cetin, Bart Merci, O Gûnay, B Uğur Töreyin, and Steven Verstockt. Methods and techniques for fire detection: signal, image and video processing perspectives. 2017.

67. Steven Verstockt, Peter Lambert, Rik Van de Walle, Bart Merci, and Bart Sette. State of the art in vision-based fire and smoke dectection. 2:285–292, 2009.

68. Panagiotis Barmpoutis, Kosmas Dimitropoulos, Kyriaki Kaza, and Nikos Grammalidis. Fire detection from images using faster r-cnn and multidimensional texture analysis. pages 8301–8305, 2019.

69. Byoung Chul Ko, Kwang-Ho Cheong, and Jae-Yeal Nam. Fire detection based on vision sensor and support vector machines. *Fire Safety Journal*, 44(3):322–329, 2009.

70. Feiniu Yuan. A fast accumulative motion orientation model based on integral image for video smoke detection. *Pattern Recognition Letters*, 29(7):925–932, 2008.

71. Suleyman Aslan, Ugur Gudukbay, B Ugur Toreyin, and A Enis Cetin. Early wildfire smoke detection based on motion-based geometric image transformation and deep convolutional generative adversarial networks. In *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 8315–8319. IEEE, 2019.

72. Arman Afrasiyabi, Diaa Badawi, Baris Nasir, Ozan Yildi, Fatios T Yarman Vural, and A Enis Çetin. Non-euclidean vector product for neural networks. In *2018 IEEE*

CITED LITERATURE (Continued)

International Conference on Acoustics, Speech and Signal Processing (ICASSP), pages 6862–6866. IEEE, 2018.

73. Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. Mobilenetv2: Inverted residuals and linear bottlenecks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4510–4520, 2018.

74. DY Chen and Pak Kwong Chan. An intelligent isfet sensory system with temperature and drift compensation for long-term monitoring. *IEEE Sensors Journal*, 8(12):1948–1959, 2008.

75. Luc Bousse, Nico F De Rooij, and Piet Bergveld. Operation of chemically sensitive field-effect sensors as a function of the insulator-electrolyte interface. *IEEE Transactions on Electron Devices*, 30(10):1263–1270, 1983.

76. Clinton ZD Goh, Pantelis Georgiou, Timothy G Constandinou, Themistoklis Prodromakis, and Christofer Toumazou. A cmos-based isfet chemical imager with auto-calibration capability. *IEEE Sensors Journal*, 11(12):3253–3260, 2011.

77. Blake C Jacquot, Nini Munoz, Darren W Branch, and Edwin C Kan. Non-faradaic electrochemical detection of protein interactions by integrated neuromorphic cmos sensors. *Biosensors and Bioelectronics*, 23(10):1503–1511, 2008.

78. Alisa Rudnitskaya. Calibration update and drift correction for electronic noses and tongues. *Frontiers in chemistry*, 6:433, 2018.

79. Lei Zhang, Yan Liu, Zhenwei He, Ji Liu, Pingling Deng, and Xichuan Zhou. Anti-drift in e-nose: A subspace projection approach with drift reduction. *Sensors and Actuators B: Chemical*, 253:407–417, 2017.

80. Tao Liu, Dongqi Li, and Jianjun Chen. An active method of online drift-calibration-sample formation for an electronic nose. *Measurement*, page 108748, 2020.

81. Yang Tao, Chunyan Li, Zhifang Liang, Haocheng Yang, and Juan Xu. Wasserstein distance learns domain invariant feature representations for drift compensation of e-nose. *Sensors*, 19(17):3703, 2019.

82. Dongliang Huang and Henry Leung. Reconstruction of drifting sensor responses based on papoulis–gerchberg method. *IEEE Sensors Journal*, 9(5):595–604, 2009.

CITED LITERATURE (Continued)

83. Dheeraj Kumar, Sutharshan Rajasegarar, and Marimuthu Palaniswami. Automatic sensor drift detection and correction using spatial kriging and kalman filtering. In *2013 IEEE International Conference on Distributed Computing in Sensor Systems*, pages 183–190. IEEE, 2013.

84. Dheeraj Kumar, Sutharshan Rajasegarar, and Marimuthu Palaniswami. Geospatial estimation-based auto drift correction in wireless sensor networks. *ACM Transactions on Sensor Networks (TOSN)*, 11(3):1–39, 2015.

85. Shenglan Ma, Jun Li, Hong Hao, and Shaofei Jiang. Structural response recovery based on improved multi-scale principal component analysis considering sensor performance degradation. *Advances in Structural Engineering*, 21(2):241–255, 2018.

86. Cosimo Distante, Marco Leo, and Krishna C Persaud. Wavelet transform for electronic nose signal analysis. *Discrete Wavelet Transforms: Biomedical Applications*, page 177, 2011.

87. Lei Zhang and Xiongwei Peng. Time series estimation of gas sensor baseline drift using arma and kalman based models. *Sensor Review*, 2016.

88. Tolga Ergen, Ali H Mirza, and Suleyman Serdar Kozat. Energy-efficient lstm networks for online learning. *IEEE Transactions on Neural Networks and Learning Systems*, 2019.

89. Pai Peng, Xiaojin Zhao, Xiaofang Pan, and Wenbin Ye. Gas classification using deep convolutional neural networks. *Sensors*, 18(1):157, 2018.

90. Shaojie Bai, J Zico Kolter, and Vladlen Koltun. An empirical evaluation of generic convolutional and recurrent networks for sequence modeling. *arXiv preprint arXiv:1803.01271*, 2018.

91. Lubna Shibly Mokatren, Ahmet Enis Cetin, and Rashid Ansari. Deep layered lms predictor. *arXiv preprint arXiv:1905.04596*, 2019.

92. A Enis Cetin, Omer N Gerek, and Sennur Ulukus. Block wavelet transforms for image coding. *IEEE Transactions on Circuits and Systems for Video Technology*, 3(6):433–435, 1993.

93. A Enis Cetin, Omer N Gerek, and Yasemin Yardimci. Equiripple fir filter design by the fft algorithm. *IEEE Signal Processing Magazine*, 14(2):60–64, 1997.

CITED LITERATURE (Continued)

94. Hamid Krim, Dewey Tucker, Stephane Mallat, and David Donoho. On denoising and best signal representation. *IEEE transactions on information theory*, 45(7):2225–2238, 1999.

95. A Enis Cetin and Mohammad Tofighi. Projection-based wavelet denoising [lecture notes]. *IEEE Signal Processing Magazine*, 32(5):120–124, 2015.

96. Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

97. Curtis R Vogel and Mary E Oman. Iterative methods for total variation denoising. *SIAM Journal on Scientific Computing*, 17(1):227–238, 1996.

98. Mohammad Tofighi, Onur Yorulmaz, Kivanç Köse, Deniz Cansen Yıldırım, Rengül Çetin-Atalay, and A Enis Cetin. Phase and tv based convex sets for blind deconvolution of microscopic images. *IEEE Journal of Selected Topics in Signal Processing*, 10(1):81–91, 2015.

99. CS Withers. Mercer's theorem and fredholm resolvents. *Bulletin of the Australian Mathematical Society*, 11(3):373–380, 1974.

100. Jssai Schur. Bemerkungen zur theorie der beschränkten bilinearformen mit unendlich vielen veränderlichen. *Journal für die reine und angewandte Mathematik*, 1911(140):1–28, 1911.

101. Christopher M Bishop and Nasser M Nasrabadi. *Pattern recognition and machine learning*, volume 4. Springer, 2006.

102. Jean Gallier. *Geometric methods and applications: for computer science and engineering*, volume 38. Springer Science & Business Media, 2011.

103. Hanying Zhou, Margie L Homer, Abhijit V Shevade, and Margaret A Ryan. Nonlinear least-squares based method for identifying and quantifying single and mixed contaminants in air with an electronic nose. *Sensors*, 6(1):1–18, 2006.

104. LTD Zhengzhou Winsen Electronics Technology Co. Ammonia gas sensor (model mq137 manual, 01 2018. `http://https://www.winsen-sensor.com/d/files/semiconductor/mq137.pdf`, 2018-01-18.

CITED LITERATURE (Continued)

105. L Carmel, S Levy, D Lancet, and D Harel. A feature extraction method for chemical sensors in electronic noses. *Sensors and Actuators B: Chemical*, 93(1-3):67–76, 2003.

106. Chengmo Yang, Patrick Cronin, Agamyrat Agambayev, Sule Ozev, A. Enis Cetin, and Alex Orailoglu. A crowd-based explosive detection system with two-level feedback sensor calibration. In *ICCAD*, 2020.

107. Lei Zhang, Fengchun Tian, Shouqiong Liu, Lijun Dang, Xiongwei Peng, and Xin Yin. Chaotic time series prediction of e-nose sensor drift in embedded phase space. *Sensors and Actuators B: Chemical*, 182:71–79, 2013.

108. DY Chen and Pak Kwong Chan. An intelligent isfet sensory system with temperature and drift compensation for long-term monitoring. *IEEE Sensors Journal*, 8(12):1948–1959, 2008.

109. Yuan Yao, Abhishek Sharma, Leana Golubchik, and Ramesh Govindan. Online anomaly detection for sensor systems: A simple and efficient approach. *Performance Evaluation*, 67(11):1059–1075, 2010.

110. Matt Calder, Robert A Morris, and Francesco Peri. Machine reasoning about anomalous sensor data. *Ecological Informatics*, 5(1):9–18, 2010.

111. Laura Erhan, M Ndubuaku, Mario Di Mauro, Wei Song, Min Chen, Giancarlo Fortino, Ovidiu Bagdasar, and Antonio Liotta. Smart anomaly detection in sensor systems: A multi-perspective review. *Information Fusion*, 2020.

112. Tie Luo and Sai G Nagarajan. Distributed anomaly detection using autoencoder neural networks in wsn for iot. In *2018 ieee international conference on communications (icc)*, pages 1–6. IEEE, 2018.

113. Zhirong Wu, Yuanjun Xiong, Stella X Yu, and Dahua Lin. Unsupervised feature learning via non-parametric instance discrimination. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3733–3742, 2018.

114. Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. Momentum contrast for unsupervised visual representation learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9729–9738, 2020.

CITED LITERATURE (Continued)

115. Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In *International conference on machine learning*, pages 1597–1607. PMLR, 2020.

116. Jihoon Tack, Sangwoo Mo, Jongheon Jeong, and Jinwoo Shin. Csi: Novelty detection via contrastive learning on distributionally shifted instances. *arXiv preprint arXiv:2007.08176*, 2020.

117. Hongyi Pan, Diaa Badawi, Erdem Koyuncu, and A Enis Cetin. Robust principal component analysis using a novel kernel related with the l1-norm. *arXiv preprint arXiv:2105.11634, presented in EUSIPCO 2021*, 2021.

VITA

NAME — Diaa H J Badawi

EDUCATION — MSc. in Electrical and Electronics Engineering, Bilkent University, Turkey, 2018
BSc. in Communication Engineering, An-Najah National University, Palestine, 2015

EXPERIENCE — Research Assistant, University of Illinois Chicago, 2018–2022
Teaching Assistant, University of Illinois Chicago, 2018–2022
Interim Research Engineer, Qualcomm Inc., California, 2020
Research Assistant, Bilkent University, 2016–2018
Teaching Assistant, Bilkent University, 2016-2018
Software Engineer, Asal Technology, Palestine, 2015 – 2016

PUBLICATIONS — **Journal Publications**

Pan, H., Badawi, D., & Cetin, A. E. (2022). Block Walsh-Hadamard Transform Based Binary Layers in Deep Neural Networks. ACM Transactions on Embedded Computing Systems (TECS).
Badawi, D., Agambayev, A., Ozev, S., & Cetin, A. E. (2021). Real-time low-cost drift compensation for chemical sensors using a deep neural network with hadamard transform and additive layers. IEEE Sensors Journal, 21(16), 17984-17994.
Nasrin, S., Badawi, D., Cetin, A. E., Gomes, W., & Trivedi, A. R. (2021). Mf-net: Compute-in-memory sram for multibit precision inference using memory-immersed data conversion and multiplication-free operators. IEEE Transactions on Circuits and Systems I: Regular Papers, 68(5), 1966-1978.
Pan, H., Badawi, D., Zhang, X., & Cetin, A. E. (2020). Additive neural network for forest fire detection. Signal, Image and Video Processing, 14(4), 675-682.
Pan, H., Badawi, D., & Cetin, A. E. (2020). Computationally efficient wildfire detection method using a deep convolutional network pruned via fourier analysis. Sensors, 20(10), 2891.

VITA (Continued)

Badawi, D., Pan, H., Cetin, S. C., & Çetin, A. E. (2020). Computationally efficient spatio-temporal dynamic texture recognition for volatile organic compound (voc) leakage detection in industrial plants. IEEE Journal of Selected Topics in Signal Processing, 14(4), 676-687.
Badawi, D., Ayhan, T., Ozev, S., Yang, C., Orailoglu, A., & Cetin, A. E. (2019). Detecting gas vapor leaks using uncalibrated sensors. IEEE Access, 7, 155701-155710.

Conference Publications

Badawi, D., Bassi, I., Ozev, S., & Cetin, A. E. (2022, May). Detecting Anomaly in Chemical Sensors via Regularized Contrastive Learning. In ICASSP 2022-2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP) (pp. 86-90). IEEE.
Pan, H., Badawi, D., Miao, R., Koyuncu, E., & Cetin, A. E. (2022, May). Multiplication-avoiding variant of power iteration with applications. In ICASSP 2022-2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP) (pp. 5608-5612). IEEE.
Pan, H., Badawi, D., Chen, C., Watts, A., Koyuncu, E., & Cetin, A. E. (2022). Deep Neural Network With Walsh-Hadamard Transform Layer for Ember Detection During a Wildfire. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (pp. 257-266).
Pan, H., Badawi, D., Koyuncu, E., & Cetin, A. E. (2021, August). Robust Principal Component Analysis Using a Novel Kernel Related with the ℓ_1-Norm. In 2021 29th European Signal Processing Conference (EUSIPCO) (pp. 2189-2193). IEEE.
Badawi, D., Agambayev, A., Ozev, S., & Cetin, A. E. (2021, June). Discrete Cosine Transform Based Causal Convolutional Neural Network for Drift Compensation in Chemical Sensors. In ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP) (pp. 8012-8016). IEEE.
Chen, Y. H., Twing, A. H., Badawi, D., Danavi, J., McCauley, M., & Cetin, A. E. (2020, May). Atrial Fibrillation Risk Prediction from Electrocardiogram and Related Health Data with Deep Neural Network. In ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP) (pp. 1269-1273). IEEE.

VITA (Continued)

Afrasiyabi, A., Badawi, D., Nasir, B., Yildi, O., Vural, F. T. Y., & Çetin, A. E. (2018, April). Non-euclidean vector product for neural networks. In 2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP) (pp. 6862-6866). IEEE.