Chapter 3

Multidimensional Multiconvolution-Based Feature Extraction Approach for Drift Tolerant Robust Classifier for Gases/Odors

3.1 Abstract

To the best of our knowledge, Convolutional Neural Network (CNN) was first used to classify gases/odors in 2017. However, several other neural network-based methods have been used for classification and drift corrections for decades. While applying these methods, various additional statistical algorithms are used to compensate for the drift. Due to such multistage procedures, these methods are not well-suited for

real-time applications. Hence, we have developed an end-to-end hybrid architecture of the CNNs suitable for real-time. Throughout the chapter, it has been called a "Drift Tolerant Robust Classifier (DTRC)." It can automatically extract salient multidimensional features from the drifted raw sensor array responses capable of efficiently classifying the gases/odors. While classifying with such extracted features, DTRC also curtails the drift impacts and outperforms the various state-of-the-art peers. The architecture of DTRC comprises three blocks performing one-, two-, and three-dimensional (1D, 2D, and 3D) operations. DTRC does not use any additional statistical algorithm to compensate for the drift, making it compatible with real-time applications. A publicly available benchmark drifted-dataset has been used to prove the efficacy of DTRC. The experimental results show the outperformance of DTRC.

3.2 Introduction

Persaud and Dodd [19] pioneered a model nose (electronic nose) that mimics an artificial olfaction system. It consists of several gas sensor elements in the form of an array popularly known as a "gas sensor array." With the help of such systems, the classification and quantification of gases/odors are determined. However, to achieve the following goals, the accurate performance is severely affected by drift. Furthermore, the corresponding data analytics for classification and quantification is performed on the generated unique signature patterns through gas sensor elements employed in the array. A model nose performs the task in three broad stages.

Data Acquisition Stage: Acquisition of unique signature patterns of the observed gases/odors through the gas sensor array.

Pre-processing Stage: Significant features are extracted from the captured raw

signature patterns. Also, feature selection, scaling, normalization, transformation, etc., are performed in this stage.

Classification or Quantification Stage: Either classification or quantification of gases/odors or both tasks are achieved with the help of suitable pattern recognition techniques. The desired higher performance of the determined task depends on the pre-processing stage or/and selection of the classifier/quantifier model.

A schematic block diagram of the model nose depicting the three stages described above is shown in Fig. 3.1. The application of model nose-based artificial olfaction systems has increased in several significant industries [159] since its conception for coal industries.

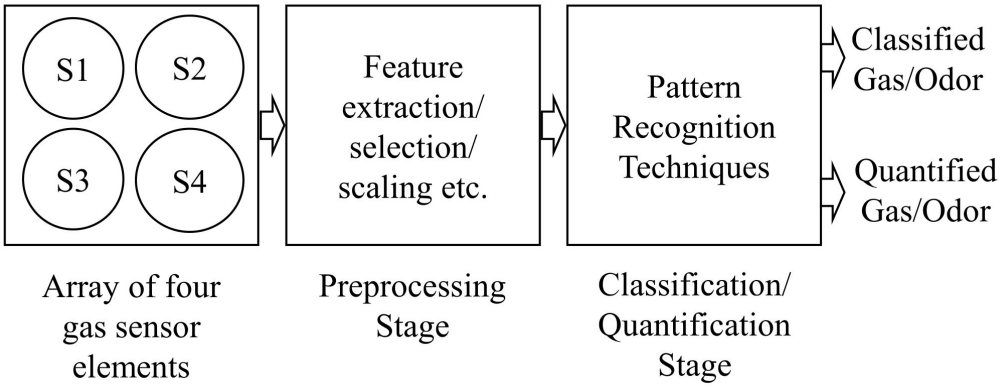

FIGURE 3.1: Three Stages of A Model Nose

Several Artificial Neural Network (ANN)-based and statistical approaches have been used conventionally to classify gases/odors for decades. A significant review of the pattern recognition techniques can be seen through [152, 160, 161]. Due to the inclusion of some hand-crafted pre-processing, the statistical methods are suitable only for laboratory-based experiments. Thus, the manual intervention in data analytics makes them inefficient for real-time applications. Fortunately, ANN-based approaches are promising candidates for gas/odor classification-based

real-time applications. It is possible only due to their capability of automatic pre-processing for feature extraction instead of manual pre-processing. However, ANN-based classifiers are prone to overfitting due to high-level interconnections among fully connected layers. Using some analysis space transformation approaches [152, 161], higher classification performances for gases/odors can be achieved without overfitting.

With the intelligent inclusion of various mathematical functions, the fully connected layers lead to the evolution of different advanced layers in the neural networks, viz., convolutional layers, and pooling layers. These advanced layers extended the conventional ANNs to CNNs. It is evident from the recently published literature [162–164] that CNNs can achieve high-performance for gas/odor classification. They also can counter the impact of overfitting and the absence of pre-processed data. In contrast, feature scaling, selection, extraction, and normalization-based data pre-processing techniques are the backbones of statistical pattern recognition approaches that contribute to obtaining higher accuracies. A variety of significant pre-processing methods can be been through [165].

Depending on the dimensions of the filters or kernels used for linear convolution, CNNs can extract temporal, spatial, and spatio-temporal features. Usually, CNN architectures sequentially possess convolutional, pooling, and fully connected layers. Convolutional layers are capable of automatic feature extraction, pooling layers are used for down-sampling of correlated features that help avoid overfitting, and fully connected layers-based multilayer perceptrons networks are used for classification. The automatic extraction of the highly distinguishable features by CNNs using multidimensional kernels (1-D, 2-D, and 3-D) serves similar to the traditional pre-processing stage. Although, ANNs have been very popular in gas sensing applications for decades. To the best of the author's knowledge, the very first attempt

for CNN application in gas sensing was found in 2017 [166]. Furthermore, interesting scientific publications on CNNs for gas/odor classification have recently been published [162–164, 166].

The conventional data (pre-)processing techniques for electronic nose include feature generation, feature extraction, feature selection, and dimension reduction. Also, in traditional approaches capturing and processing the whole data (dynamic and static) consumes more resources. Hence, these approaches are unsuitable for gas sensing applications in resource-constrained paradigms. While solving these issues, Qi et al. (2017) in [166] use a simplified CNN approach to classify the gases/odors that use fewer responses (sampled up to 15 seconds). They have implemented a simple CNN inspired by LeNet-5 [167] to classify seven Chinese liquors. Moreover, the use of CNN also extracts salient features automatically without requiring any preprocessing methods. With this ideology, CNN outperforms traditional approaches (LDA, SVM, BP-NN) and provides a high classification rate of 95.7 percent instead of 92.9. They have used two 2D convolutional layers, two pooling layers, a rasterization layer (flatten layer), a fully connected layer, and a softmax layer in their network. Further, their dataset consists of 211 samples of the considered varieties of Chinese liquors obtained by using a 10 element MOX gas sensor array. Their dataset also did not exhibit any drift in sensor characteristics. They have quoted that the sensors' original responses were first converted into grayscale images for their use in a 2D-CNN as input. Although, they have not discussed how this conversion was implemented.

Moreover, Peng et al. (2018) have proposed a 2D Deep CNN "GasNet" on four gases (carbon monoxide, methane, hydrogen, and ethylene), achieving 95.20% classification accuracy over a single batch of data, without any drift in the sensor characteristics [162]. Their network consists of six convolutional blocks and one

global average pooling layer. Further, they have used three types of layers (convolution, batch normalization, and activation) in each of the six similar convolutional blocks. Their dataset consists of 1200 samples of the considered gases obtained by an 8 element MOX gas sensor array. Each of these sensor responses consisted of 1000 vectors in a time series, with 8 elements in each vector. They have claimed their work to be the first of its kind using a 2D-CNN. Further, the GasNet proposed by Peng et al. (2018) comprises 38 layers, making the CNN highly complex. Further, Zhao et al. (2019) have proposed a 1D Deep CNN on certain mixtures of gases (ethylene, CO, methane), achieving 96.30% recognition accuracy over a single batch of drift-free sensor response datasets [163]. They have used a network with four convolutional stages, one dropout layer, one pooling layer, and two fully connected layers. They have further used three more layers (convolutional, regularization, activation) in each of the four convolutional stages. Their dataset consists of 593 samples of binary mixtures of Ethylene with Methane (CH4) and Carbon Monoxide (CO) obtained by using a 16 element MOX gas sensor array. Wei et al. (2019) have implemented the modified pioneer architecture of CNN "LeNet-5" [167] to classify CO, Methane, and its mixtures, achieving 98.67% identification accuracy with the unused downsampled test dataset dimension 16×12 [164]. However, they tried many other dimensions for down-sampling, making the classification process highly complex. Their dataset also did not exhibit any drift in sensor characteristics. They have used improved LeNet-5 network with two convolutional layers, two pooling layers, and two fully connected layers. Further, their dataset consists of 480 samples of the considered gases and their mixture obtained by using a 12 element MOX gas sensor array. The gases/odors have been classified using drift-free gas sensor array responses in these discussed works. Also, it has been observed that the authors have utilized the capability of either one-dimensional or two-dimensional convolution-based CNNs to classify the considered gases/odors.

In this chapter, we will use a hybrid architecture of CNN that utilizes the one-, two-, and three-dimensional convolution capability. The corresponding convolutional blocks automatically extract salient features from the gas sensor array responses. With this approach, we have novelly used three-dimensional convolution to extract spatio-temporal features to classify the gases/odors efficiently. The features extracted from 3D convolution contain more comprehensive information than the temporal or spatial features obtained from 1D or 2D convolution. Accordingly, the utilized hybrid CNN, well-published in [168], is an end-to-end multidimensional multiconvolution architecture. It is referred to as a "Drift Tolerant Robust Classifier (DTRC)" throughout the chapter/thesis. Our DTRC can classify gases/odors efficiently, even using drifted gas sensor array responses. This approach has several merits compared to peers developed to curb drift effects while classifying the considered gases/odors. The architecture of DTRC is depicted in Fig. 3.2. The merits of DTRC can be highlighted as follows:

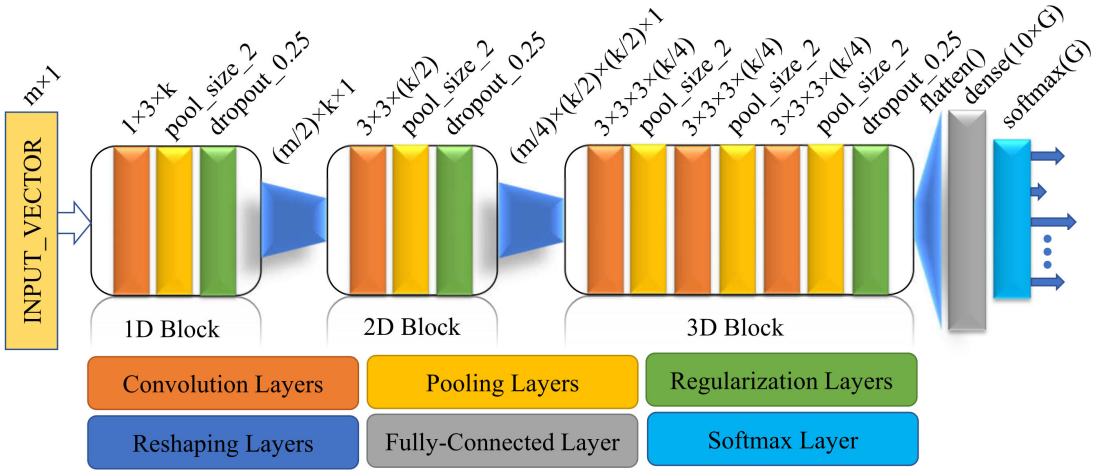

FIGURE 3.2: The End-to-End Hybrid Convolutional Neural Network (CNN) Architecture of Drift Tolerant Robust Classifier (DTRC)

☞ The DTRC can deliver higher performance without using any statistical algorithms for drift correction.

- ☞ The architecture of DTRC is straightforward to implement and understand the working.
- ☞ DTRC is a novel multidimensional multiconvolution end-to-end architecture of CNN capable of classifying gases/odors even using drifted gas sensor array responses.
- ☞ It is also efficient for real-time applications being an end-to-end model.

3.3 Corresponding Gas Sensor Array and Data

The performance of our DTRC has been demonstrated on a widely popular dataset among the gas sensing research community. This dataset is publicly available on the website of the Machine Learning Repository: Center for Machine Learning and Intelligent Systems, governed by the University of California Irvine [169]. Furthermore, this dataset was recorded by Vergara et al. (2012) by observing the responses of a gas sensor array periodically for up to three years (36 months). This dataset has been represented by ten batches (see Table 3.1) which collectively consist of 13,910 samples [86, 170]. These samples were recorded at the exposure of different concentrations (see Table 3.2) for the considered gases. Moreover, the data contained in all ten batches have been captured in different months for 36 months, as shown in Table 3.3. Thereby, the whole data is inherently affected by drift.

Further, to obtain the pre-processed data used to classify/quantify the considered gases/odors, eight salient features have been extracted from the raw responses, as shown in Fig. 3.3 [86]. These eight features include raw and normalized steady-state features and six transient features. These transient features have been obtained through the maximum and minimum exponential moving average (ema_α)

TABLE 3.1: Batch-wise Distribution of Samples for Considered Gases/Odors

Batches ↓	Gases/Odors →					
	Ethanol	Ethylene	Ammonia	Acetaldehyde	Acetone	Toluene
1	83	30	70	98	90	74
2	100	109	532	334	164	5
3	216	240	275	490	365	0
4	12	30	12	43	64	0
5	20	46	63	40	28	0
6	110	29	606	574	514	467
7	360	744	630	662	649	568
8	40	33	143	30	30	18
9	100	75	78	55	61	101
10	600	600	600	600	600	600
Total	1641	1936	3009	2926	2565	1833
Grand Total	**13910**					

TABLE 3.2: Concentration Ranges for Considered Gases/Odors

Gases/Odors	Ethanol	Ethylene	Ammonia	Acetaldehyde	Acetone	Toluene
Concentration Ranges (ppmv)	10-600	10-300	50-1000	5-500	12-1000	10-100

TABLE 3.3: Month IDs Corresponding to Batch ID

Batch ID	1	2	3	4	5	6	7	8	9	10
Month IDs	1,2	3,4,8-10	11-13	14,15	16	17-20	21	22,23	24,30	36

values corresponding to the injection and recovery phase at $\alpha = 0.1, 0.01, 0.001$. The aforementioned features' corresponding expressions have been given in equation (3.1)-(3.8). The used gas sensor array to capture this dataset consists of 16 sensor elements (see Table 3.4). Hence, each feature vector in the pre-processed dataset embraces $128(= 8 * 16)$ elements. The research community utilizes this comprehensive benchmark dataset for various purposes, viz., drift correction, classification, feature selection, and quantification.

steady-state feature

$$\Delta R = max_k r[k] - min_k r[k] \tag{3.1}$$

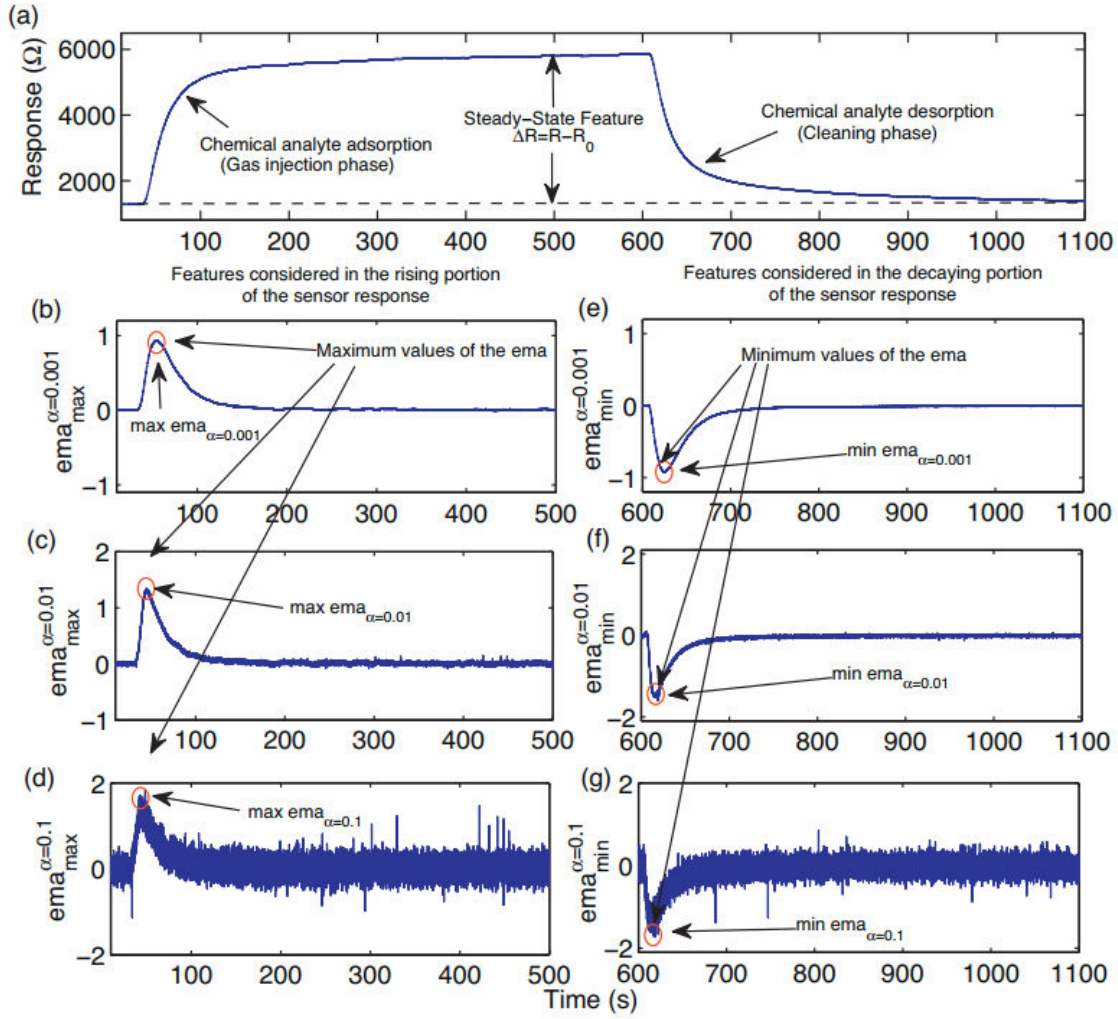

FIGURE 3.3: A Pictorial Representation of Extracted Salient Features from Raw Data → (a) Raw response of MOX gas sensor for Acetaldehyde at 30ppmv concentration. The graph shows three phases: baseline, injection phase, and recovery phase. (b)-(d) exponential moving average of injection phase for $\alpha = 0.1, 0.01, 0.001$. (e)-(g) exponential moving average of recovery phase for $\alpha = 0.1, 0.01, 0.001$.

normalized steady-state feature

$$\|\Delta R\| = \frac{max_k r[k] - min_k r[k]}{min_k r[k]} \tag{3.2}$$

exponetial moving average-based injection phase trasient features

$$max_k ema_{\alpha=0.1}(r[k]) \tag{3.3}$$

$$max_k ema_{\alpha=0.01}(r[k]) \tag{3.4}$$

$$max_k ema_{\alpha=0.001}(r[k]) \tag{3.5}$$

exponetial moving average-based recovery phase trasient features

$$min_k ema_{\alpha=0.1}(r[k]) \tag{3.6}$$

$$min_k ema_{\alpha=0.01}(r[k]) \tag{3.7}$$

$$min_k ema_{\alpha=0.001}(r[k]) \tag{3.8}$$

where R and $r[k]$ represent the resistance and resistance at discrete time index k, respectively.

Table 3.4: Types of Gas Sensor Elements Used in Gas Sensor Array

Type of Gas Sensor	TGS2600	TGS2602	TGS2610	TGS2620
Used Units	4	4	4	4

3.4 Design & Training of DTRC

The innovative idea behind to design of DTRC was that if only 1D or 2D CNN are capable of delivering the high-performance classification of gases/odors, then utilizing 1D, 2D, and 3D convolution simultaneously, the extracted salient features can be learned by a hybrid architecture of CNN. By using such extracted features, the hybrid CNN will be capable of providing high performance even using drift-affected

datasets. With this idea, CNN's proposed novel hybrid architecture consists of a dedicated block for each 1D, 2D, and 3D convolution-based feature extraction, as shown in Fig. 3.2. These blocks operate automatically in feed-forward kind conjunction and extract temporal, spatial, and spatio-temporal features from calibrated response vectors consisting of 128 data points. The expressions implementing 1D, 2D, and 3D convolution operations are given by equations (3.9)-(3.11).

$$C(p) = \sum_{n=1}^{S} I(p+n) \times F(n) \tag{3.9}$$

where I is the input having size $(M)_I$ and F is the kernel having size $(S)_F$. $C(p)$ represents the convolution at location (p) calculated over the range $1 \leqq p \leqq M - (S-1) + 1$. In this case, I represents one-dimensional datasets like time series datasets.

$$C(p,q) = \sum_{n_1,n_2=1}^{S} I(p+n_1, q+n_2) \times F(n_1, n_2) \tag{3.10}$$

where I is the input having size $(M \times N)_I$, and F is the kernel having size $(S \times S)_F$. $C(p,q)$ represents the convolution at location (p,q) calculated over the range $1 \leqq p \leqq M-(S-1)+1$, $1 \leqq q \leqq N-(S-1)+1$. In this case, I represents two-dimensional datasets like single channel images (binray or grayscale images).

$$C(p,q,r) = \sum_{n_1,n_2,n_3=1}^{S} I(p+n_1, q+n_2, r+n_3) \times F(n_1, n_2, n_3) \tag{3.11}$$

where I is the input having size $(M \times N \times P)_I$, and F is the kernel having size $(S \times S \times S)_F$. $C(p,q,r)$ represents the convolution at location (p,q,r) calculated over the range $1 \leqq p \leqq M-(S-1)+1$, $1 \leqq q \leqq N-(S-1)+1$, and $1 \leqq$

$r \leqq P-(S-1)+1$. In this case, I represents three-dimenaional datasets like multichannel images (RGB or multispcetral or hyperspectral images).

Similarly, pooling is performed on 1D, 2D, and 3D datasets following equations (3.12)-(3.14).

$$Z(p) = \frac{1}{K} \times \sum_{n=p\times K-(K-1)}^{p\times K} I(n) \tag{3.12}$$

where $Z(p)$ represents the pooling performed at location (p) with pooling size K on a one-dimensional dataset.

$$Z(p,q) = \frac{1}{K\times K} \times \sum_{n_1=p\times K-(K-1),n_2=q\times K-(K-1)}^{p\times K,q\times K} I(n_1,n_2) \tag{3.13}$$

where $Z(p,q)$ represents the pooling performed at location (p,q) with pooling size K on a two-dimensional dataset.

$$Z(p,q,r) = \frac{1}{K\times K\times K} \times \sum_{n_1=p\times K-(K-1),n_2=q\times K-(K-1),n_3=r\times K-(K-1)}^{p\times K,q\times K,r\times K} I(n_1,n_2,n_3) \tag{3.14}$$

where $Z(p,q,r)$ represents the pooling performed at location (p,q,r) with pooling size K on a three-dimensional dataset.

For DTRC, the layerwise details on connecting parameters among the dedicated block for 1D, 2D, and 3D operations have been given in Table 3.5. The outcomes from the 3D block are forwarded to the fully connected layer, which has the number of neurons ten times the number of target gases/odors. Subsequently,

the fully connected layer outcomes are forwarded to the softmax layer that provides probability fractions for all outputs to be predicted. The categorical cross-entropy loss function and the Adam optimizer [171] have been used to compile the DTRC. The loss function and weight updating expressions have been shown through equations (3.15), and (3.16). Moreover, Glorot's uniform distribution [172] has been used to initialize the kernel values (wights), while all the biases are initialized to zero. Convolution and pooling operations have been performed using a single stride to go through all the variations to extract features from the inputs. Default zero-padding has been used during the convolution to keep the input and output dimensions the same while using convolution. The pooling has been performed by taking averages with a window of size 2. Including the regularization layers, DTRC uses a booster shot to avoid overfitting by using early stopping with the patience of 20 epochs for maximum accuracy.

TABLE 3.5: Input and Output Shapes Through 1D, 2D, and 3D Blocks

Blocks →	1D Block	2D Block	3D Block
Input Size	128×1	$64 \times 64 \times 1$	$32 \times 32 \times 32 \times 1$
(C, P, R)	$(1, 1, 1)$	$(1, 1, 1)$	$(3, 3, 1)$
Kernels	64	32	$16, 16, 16$
Output Size	64×64	$32 \times 32 \times 32$	$4 \times 4 \times 4 \times 16$
Reshape	$64 \times 64 \times 1$	$32 \times 32 \times 32 \times 1$	1024
$(C : Convolutional, P : Pooling, R : Regularization) Layers$			

$$L(t_a, t_p) = -\sum_{c=1}^{C}(t_a) log(t_p) \tag{3.15}$$

where t_a and t_p are the actual and predicted targets.

$$W_t = W_{t-1} + \alpha \frac{\hat{V}_t}{\sqrt{\hat{S}_t} + \epsilon} \tag{3.16}$$

where W, α, $\hat{V}_t$, $\hat{S}_t$, and ϵ are the weights, learning rate, estimators of corrected bias for a first and second moment, and a fuzz factor to avoid the true divide.

As discussed, the used dataset consists of 13910 samples of considered gases/odors distributed in ten batches, where 128 data points represent each sample vector in each of the batches. Starting with this shape of sample vectors and automatically changing the size of inputs and outputs according to Table 3.5, we achieve output in the form of a fully connected layer with 1024 neurons at the end of the execution of the 3D block. Up to this stage, the hyperbolic tangent (***tanh***) function has been used as an activation function. Thus, the obtained fully connected layer is further forwarded to another dense layer with ten times the number of neurons to the number of targets. The subsequent outcomes of this dense layer are fed into the softmax layer to classify the considered gases/odors. In this context, DTRC is trained using the functions (as represented through equations (3.9)-(3.16)) and hyperparameters (e.g., $learning\_rate = 0.001$, $epochs = 100$, and $batch\_size = 32$). Rest hyperparameters are taken with default values. Consequently, the DTRC outperforms by utilizing the simultaneous significance of salient features extracted using multidimensional multiconvolution.

3.5 Results

With this inference, the DTRC is first trained separately on each batch, using 80% samples with tenfold cross-validation. Then the trained DTRC was evaluated on randomly chosen 20% samples of the corresponding batch. Thus, the obtained performance of DTRC has been given in Table 3.6. Subsequently, the trained DTRC is also tested for upcoming batches, and the corresponding results are shown in Table 3.7. At the same time, the differential performance of DTRC with Vergara et al.

(2012) [86] has been highlighted in Fig. 3.4. Up to this analysis, DTRC provides better classification results than the classification results obtained by Vergara et al. (2012) [86].

TABLE 3.6: Batch-wise Classification Accuracy Obtained by DTRC for 20% Samples of Individual Batches

Batch-1 to 5	96.63	100	100	100	97.50
Batch-6 to 10	98.70	99.72	93.22	100	99.31

TABLE 3.7: Batch-wise Classification Accuracy Obtained by DTRC for Subsequent Batches

	Tested Batches (2-10) →								
Trained Batches (1-9) ↓	2	3	4	5	6	7	8	9	10
1	40.27	59.08	63.35	34.52	43.57	32.49	10.20	24.68	25.36
2		97.67	100	76.65	78.82	69.95	31.52	76.42	51.83
3			90.68	66.50	81.40	70.18	19.93	74.80	48.70
4				91.37	63.50	60.52	15.94	37.40	33.13
5					69.89	57.60	15.22	29.54	33.77
6						74.56	70.75	41.06	39.31
7							89.80	71.06	48.03
8								87.66	32.14
9									35.83

In addition, to compare the performance of DTRC with state-of-the-art peers, we have adopted another two settings of experimentation. Accordingly, DTRC is trained again to the settings mentioned below.

Setting 1: $Batch-(n)$ dataset has been used to train the DTRC, and then the trained DTRC is tested on $Batch-(n+1)$; where $n = 1, 2, \ldots, 9$.

Setting 2: $Batch-1$ and $Batch-2$ data have been used to train the DTRC, and then the trained DTRC is tested on $Batch-(n)$; where $n = 3, 4, \ldots, 10$.

Under these settings, the performance of DTRC is compared with several following state-of-the-art approaches.

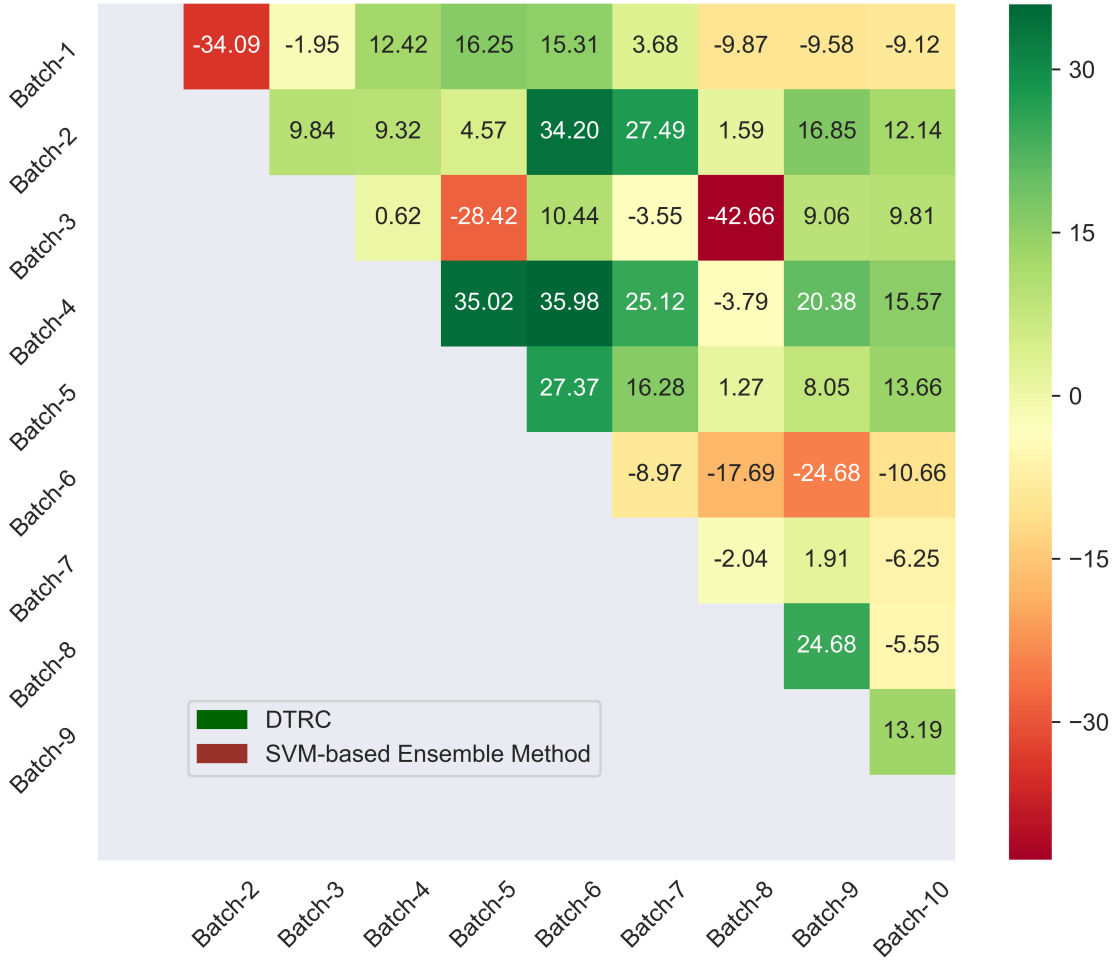

FIGURE 3.4: Differential Performance of DTRC *w.r.t.* Vergara et al. (2012)

The conventional approaches to dealing with the drift are costly and tedious. Hence, being inspired by semi-supervised domain adaptation, Liu et al. (2013) have proposed domain adaptation-based approaches to deal with the drift [173]. In their experiment, the radial basis function (RBF) kernel-based multiclass SVM (SVM-RBF) as a supervised method and in the form of a semi-supervised method RBF kernel-based manifold regularization (ML-RBF) have been used to classify the gases/odors. Moreover, it is reported in [174, 175] that the RBF kernel-based

extreme learning machine (ELM-RBF) provides better-generalized performance than traditional SVM. Zhang et al. (2017) have utilized domain regularized component analysis (DRCA) [176] for drift adaptation. Furthermore, Leon-Medina et al. (2020) have used the insight of transfer learning with the joint distribution adaptation (JDA) method to counter the drift issues [177]. In addition to these methods, Badawi et al. (2019) have used neural networks for the same purposes. They have utilized multiple combinations of various neural networks to suppress the drift impacts. These neural networks and combinations include multilayer perceptron (MLP), additive neural network (AddNet) with MLP (AddNet-MLP), discriminator of a generative adversarial neural network (DiscGAN), the combination AddNet-DiscGAN, and domain adaptation [178].

The performance comparisons of DTRC with the above-mentioned state-of-the-art methods have been shown in Fig. 3.5 and Fig. 3.6, respectively.

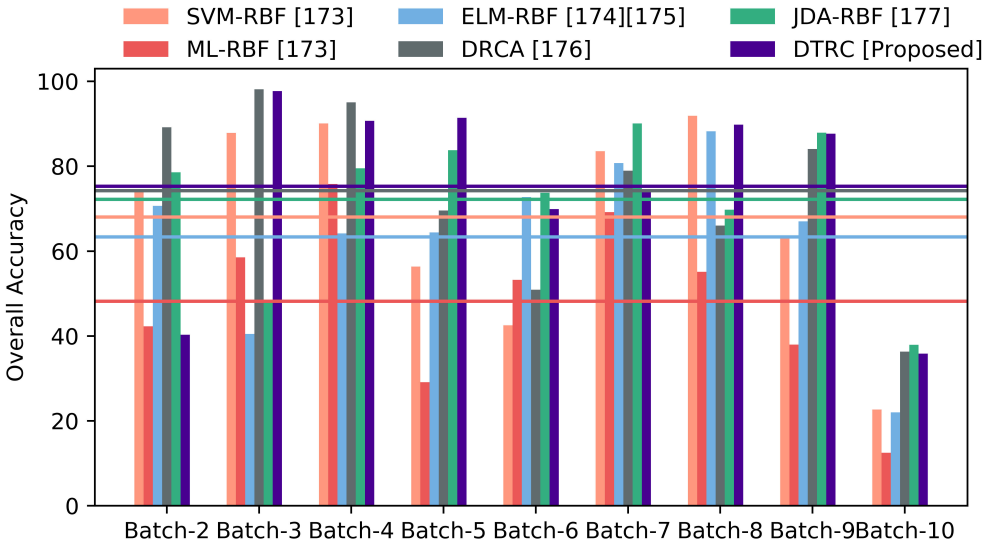

FIGURE 3.5: Performance Comparison of DTRC *w.r.t.* State-of-The-Art Methods Under Setting-1

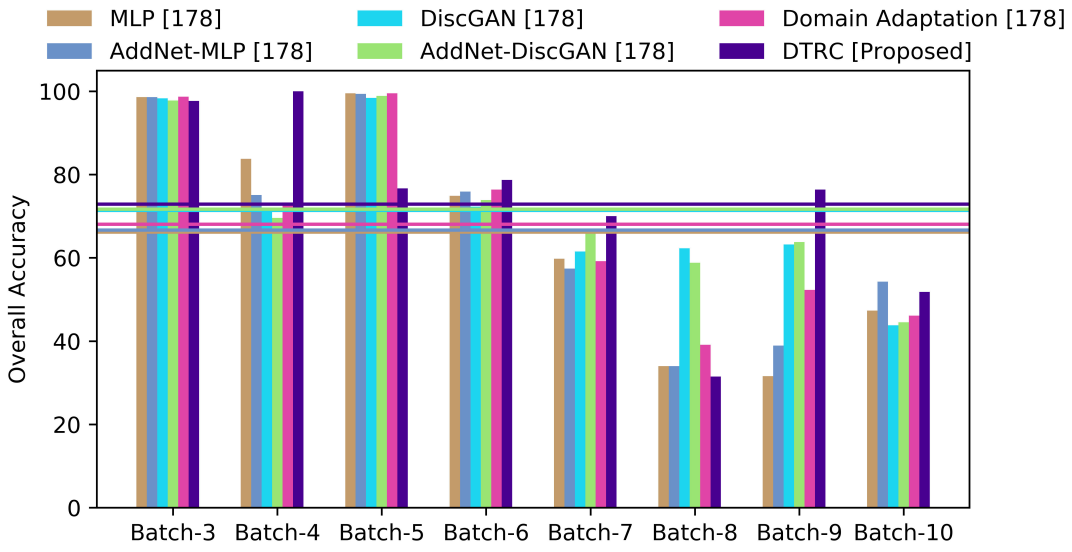

FIGURE 3.6: Performance Comparison of DTRC *w.r.t.* State-of-The-Art Methods Under Setting-2

3.6 Discussion

The DTRC is first trained and tested using 80 percent and 20 percent samples, respectively. During the training, ten-fold cross-validation was applied to counter the intra-batch drift automatically. Thus, the experimental results obtained in this case have been shown in Table 3.6. The DTRC provides high classification results on all individual batches.

The absolute difference between the performance of DTRC and SVM-based ensemble classifier [86] has been shown in Fig. 3.4 to represent the outperformance of DTRC. Wherever the positive absolute differences have been mentioned in contrast to the negative absolute differences, show the outperformance of DTRC. Accordingly, the graded color shades have also been used, from dark green to deep red. The green shade is darker and the better is the performance of DTRC than the SVM-based ensemble classifier. It is evident from Fig. 3.4 that the DTRC outperforms the SVM-based ensemble classifier at 29 out of 45 instances, especially when the batches are

significantly affected by drift.

The multidimensional multiconvolution-based strategy of DTRC uses saliency of temporal, spatial, and spatio-temporal features. In contrast, the recent literature on CNN to classify the gases/odors have used either 1D-CNN or 2D-CNN only. Moreover, the implementation of DTRC is straightforward. Even with a hybrid architecture, DTRC shows the layer-wise less complex architecture. The layer-wise comparison of DTRC and some state-of-the-art CNN architecture used for gas classification have been given in Table 3.8.

TABLE 3.8: A Layer-wise Comparison of Some State-of-The-Art CNN Architectures Used for Gas Classification

Layers ↓	Zhao et al. [163]	Peng et al. [162]	Wei et al. [164]	DTRC
C	19-1D	12-2D	2-2D	1-1D, 1-2D, 3-3D
F	1	1	2	1
T	44	38	12	23
$(C : Convolutional, F : Fully - Connected, T : Total) Layers$				

With this outperformance, it is evident that DTRC can deal with drift issues while classifying the gases/odors without incorporating additional statistical algorithms for drift correction. Also, being an end-to-end hybrid CNN architecture, DTRC is more suitable for real-time applications than multistage solutions for the same purposes.

Moreover, DTRC has also been compared with several state-of-the-art gas/odor classification methods that use the same drift-affected dataset. The comparative analysis of performances under Setting-1 and Setting-2 have been shown in Fig. 3.5 and Fig. 3.6, respectively. In these figures, grouped bars show the batch-wise actual classification accuracies, while horizontal lines have been used to represent the overall average accuracies for the tested batches. For Setting-1, DTRC outperforms for most of the batches (e.g., 3–6, 8–10). Moreover, under Setting-2, DTRC outperforms on batches 3, 4, 6, 7, 9, and 10. DTRC has done so without using any data

pre-processing for drift correction. In contrast, other methods provide enhanced performance after applying drift correction additionally. Using the referred severely drift-affected dataset DTRC achieves overall average classification accuracy of 75.30 and 72.90 percent, respectively, for Setting-1 and Setting-2. With this accuracy, DTRC outperforms all the compared state-of-the-art methods. It is not easy to perform well on all batches simultaneously due to the imbalanced distribution of samples. These batches hold 445, 1244, 1586, 161, 197, 2300, 3613, 294, 470, and 3600 samples, respectively from 1 to 10.

3.7 Conclusion

As evident from the architecture of DTRC, our proposed hybrid CNN approach simultaneously use the saliency of 1D-, 2D-, and 3D convolutions-based extracted features. The 1D convolution is achieved by operating 1D kernels or filters on 1D input data vectors. But, only linear, temporal, or spectral variations can be recorded using these kernels. In contrast, to record the spatial variations of data in the 2D feature space, 2D filters are required. Further, 3D kernels can capture the synergic variations of data in both directions, spectrally and spatially. It is worth noticing that to operate 2D and 3D kernels; we require input data with corresponding representations. Suppose we use some additional pre-processing to transform the data. In that case, it increases the computational complexity significantly, whereas we have already planned to use a relatively complex hybrid architecture to squeeze the insight of multidimensional multiconvolution. To escape from this dilemma, we have strategically designed DTRC.

The need for input data transformation according to the dimensionality of convolution operation bypassed using a reshaped layer within hybrid architecture.

In this way, we have reduced the risk of increasing computational load. In the second attempt, the computational complexity of the architecture is diminished by strategically selecting the number of kernels while using 1D, 2D, and 3D blocks. As the dimensionality of the convolution operation increased, we reduced the number of used kernels to half so that the homogeneity of computational load kept maintained throughout the architecture as possible. It is the point to be noted that the second and third blocks need less effort as their inputs are already well-extracted features instead of raw data vectors. With this synergetic capability, DTRC outperforms several well-published methods and even bypasses the necessity of drift correction statistical algorithms. This capability and its end-to-end architecture make it suitable for real-time applications.