Deep Learning with Dense Random Neural Networks

Conference Paper *in* Advances in Intelligent Systems and Computing · October 2017

DOI: 10.1007/978-3-319-67792-7_1

2 authors:

Erol Gelenbe
Imperial College London
685 PUBLICATIONS **14,496** CITATIONS
SEE PROFILE

Yonghua Yin
Imperial College London
38 PUBLICATIONS **261** CITATIONS
SEE PROFILE

Deep Learning with Dense Random Neural Networks

Erol Gelenbe and Yonghua Yin

Intelligent Systems and Networks Group
Department of Electrical and Electronic Engineering
Imperial College, London SW7 2BT

Abstract. We exploit the dense structure of nuclei to postulate that in such clusters, the neuronal cells will communicate via soma-to-soma interactions, aswell as through synapses. Using the mathematical structure of the spiking Random Neural Network, we construct a multi-layer architecture for Deep Learning. An efficient training procedure is proposed for this architecture. It is then specialized to multi-channel datasets, and applied to images and sensor-based data.

1 Introduction

Deep learning has achieved great success [6], through multilayer architectures that extract high-level representations from raw data, and many techniques facilitate the training process such as layer-wise pre-training [23], stochastic gradient descent [3], dropout [33] and batch normalization [24]. Other work on extreme learning machines has been generalized to multilayer networks without fine-tuning of the whole network [35].

The spiking Random Neural Network (RNN), a stochastic integer-state "integrate and fire" model [13], where neurons interact with each other via excitatory and inhibitory spikes, has the property that its state probability distribution is given by an easily solvable system of non-linear equations and has been used for numerous applications [7, 9–12, 15, 17, 20, 22, 26, 29–32, 34] that exploit its recurrent structure. In recent work [21],the RNN is used for deep learning by exploiting the asymptotic behaviour of the network through simplified transfer functions with multilayer and ELM arcitectures [25, 35] resulting in high classification rates on real data sets with limited computational cost.

The human brain contains important areas composed of dense clusters of cells, such as the basal ganglia and various nuclei composed of similar or varieties of cells. We suggest that such clusters allow for direct communication between stomata, in addition to signalling through dendrites and synapses. We develop a multi-layer architecture where layers of densely packed RNN nuclei with a statistically structured interconnections that allow for great individual variability among neuron spiking times and interconnection patterns, and cells communicate [18] in a recurrent fully connected structure that uses synapses and soma-to-soma interaction. Communication between layers uses a conventional multi-layer feedforward structure, where first-layer cells receive excitation signals from external sources, and cells in each nucleus have an inhibitory projection to the next layer.

This RNN Multiple Layer Architecture (RNN-MLA) shown in Figure 1 is related to previous work [21], and this paper makes the following contributions:

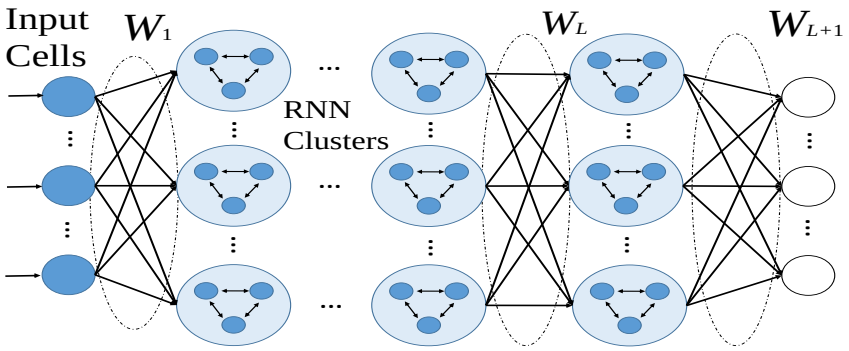


Fig. 1. Schematic representation of the RNN-MLA.

- We show how network parameters such as the number of cells per nucleus, the firing rate of the cells, and the external arrival rates of spikes hould be selected.
- The training procedure is improved by normalizing the inputs to nuclei and adjusting external arrival rates of spikes inside nuclei.
- The multi-channel RNN-MLA (MCRNN-MLA), extended from the RNN-MLA, is proposed to handle multi-channel classification datasets and applied to recognizing 3D objects, and distinguishing different chemical gases. The results indicate that the MCRNN-MLA is more accurate than state-of-the-art methods in most cases with high training efficiency.

2 The Mathematical Model

Though mall groups of spiking neurons can be represented conveniently using differential equation models, when large ensembles of hundreds or thousands of cells are represented, as in the nuclei that we consider, probability models are more convenient and tractable [14, 37]. Thus we will use the RNN, a stochastic and recurrent model [16] that mimics a very large "integrate and fire" system [4], with cells that are represented by a discrete cell state. The emission of spikes occurs from a neuron when its discrete state drops by 1 without the external arrival of an inhibitory spike. In the RNN-MLA architecture of Figure 1, nuclei in the first (input) layer cells receive excitatory spike trains from external sources, resulting in a linear cell activation $q(x) = x$. The successive L hidden layers ($L \geq 2$) are composed of dense nuclei (clusters) that receive inhibitory spike trains from cells in the previous layer, with a resultant activation function $q(x) = \zeta(x)$. Let us denote the connecting weight matrices between layers of the L-hidden-layer RNN-MLA by $W_1, \cdots, W_L \geq 0$ and output weight matrix by W_{L+1}. Given an input matrix X, a forward pass of X in the RNN-MLA can be described as:

$$Q_1 = \min(X, 1),\ Q_l = \zeta(Q_{l-1}W_{l-1}),\ l = 2, \cdots, L+1,\ O = Q_{L+1}W_{L+1}, \qquad (1)$$

where Q_1 is the 1st layer output, Q_l is the lth layer output, and O is the final RNN-MLA output. The element-wise operation $\min(x,y)$ produces the smaller value between x and y. For notation ease, we will use $\zeta(\cdot)$ as a term-by-term function for vectors and matrices.

2.1 Nuclei with Inhibitory Cells

If a nucleus is composed of statistically identical inhibitory cells, then using the RNN equations [13], the excitation (activation) probability q of any of the statistically identical cells as a function of the external inputs $[\lambda^+, \lambda^-]$ becomes:

$$q = \frac{\lambda^+}{r+\lambda^-+qr} = \frac{\sqrt{(r+\lambda^-)^2+4r\lambda^+}-(r+\lambda^-)}{2r}. \tag{2}$$

where r is the firing rate of each cell, and λ^+, λ^- are Poisson the externally arriving excitatory and inhibitory spike train rates.

2.2 Activation only through Some-to-Soma Interactions

Soma-to-soma interactions occur when a cell in a cluster, say C_1 fires, and provokes the simultaneous or quasi-synchronous firing of other cells C_2, ... ,C_m which are also excited, and possibly leading to the excitation of some other cell C_{m+1} which in turn may fire later. As a result, the excitation level of cells C_1, ... ,C_m drops, while the excitation level of cell C_{m+1} rises.

Clearly, there may be many such patterns of communication, and the simplest occurs as follows in an $n \geq 3$-cell network. Let us fix some cell, say C_1, and consider any other cell C_2, selected with probability $1/(n-1)$; if it is excited it will fire at rate r and cause some other cell other than C_1, say C_3 now selected with probability $1/(n-2)$, to fire and together they will excite C_1: this leads to the terms in the numerator of (3). The terms in the denominator of (3) result from the case where any excited cell such as C_2 fires at rate r and triggers the firing of C_1. In the RNN formalism, this leads to the following representation of the excitation probability q of cell C_1, assuming that all cells are homogenous and statistically identical, follows from the results in [19]:

$$q = \frac{\lambda^+ + (n-1)\frac{1}{n-1}qr(n-2)\frac{1}{n-2}q\frac{1}{n-1}}{r+\lambda^-+(n-1)\frac{1}{n-1}qr\frac{1}{n-1}}, \text{ or } q = \frac{\lambda^+}{r+\lambda^-}, \tag{3}$$

r which is mathematically equivalent to a singe cell with no interactions.

2.3 Random Selection of Soma-to-Soma Interactions

Again using [18, 19] consider a nucleus whose cells receive an inhibitory input from some external cell u of the form $q_u w_u^-$ where q_u is the state of the external cell, and w_u^- is the corresponding inhibitory weight. We then obtain [21] a model for the interconnect pattern with soma-to-soma interactions within a nucleus of n cells. When a given cell fires, it provokes repeated firing with probability p among the the subset of cells of size

Algorithm 1 Improved training procedure for the RNN-MLA

Get data matrix X and label matrix Y
for $l = 1, \cdots, L-1$ **do**
  solve Problem (8) for W_l with input X
  $W_l \leftarrow W_l / \max(\zeta(XW_l))/10$
  $X \leftarrow \zeta(XW_l)$
randomly generate W_L in range [0 1]
$W_{L+1} \leftarrow \text{pinv}(\zeta(XW_L))Y$

$n-1$ which contains the cell that first fired, and terminates with probability $(1-p)$ by exciting the nth cell which was not among the $n-1$ initial cells. Similarly, other cells may fire and deplete the potential of the nth cell. Assuming a homogenous and statistical identical population, we get:

$$q = \frac{\lambda^+ + rq(n-1)\sum_{l=0}^{\infty}\left(\frac{qp(n-1)}{n}\right)^l \frac{1-p}{n}}{r+\lambda^- + q_u w_u^- + rq(n-1)\sum_{l=0}^{\infty}\left(\frac{qp(n-1)}{n}\right)^l \frac{p}{n}}, \tag{4}$$

becoming:

$$q = \frac{\lambda^+ + \frac{rq(n-1)(1-p)}{n-qp(n-1)}}{r+\lambda^- + q_u w_u^- + \frac{rqp(n-1)}{n-qp(n-1)}} \tag{5}$$

which is a second degree polynomial in q:

$$\begin{aligned} q^2 p(n-1)(\lambda^- + q_u w_u^-) + q(n-1)\left(r(1-p) - \lambda^+ p\right) \\ -qn(r+\lambda^- + q_u w_u^-) + \lambda^+ n = 0. \end{aligned} \tag{6}$$

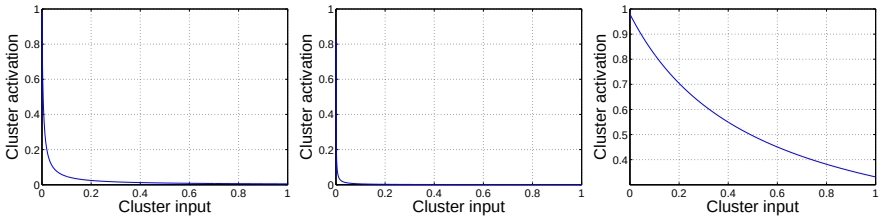
Its positive root which is less than one is $q(q_u w_u^-) \equiv \zeta(q_u w_u^-)$, since the value of q that we seek is a probability. Leting $x = q_u w_u^-$:

$$\zeta(x) = \frac{-(C-nx) - \sqrt{(C-nx)^2 - 4p(n-1)(\lambda^- + x)d}}{2p(n-1)(\lambda^- + x)}, \tag{7}$$

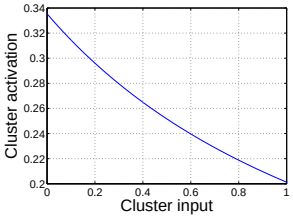
where $d = n\lambda^+$ and $C = \lambda^+ p + rp - \lambda^- n - r - \lambda^+ pn - npr$.

3 An Improved Training Procedure for the RNN-MLA

In the nucleus activation function $q(x) = \zeta(x)$ of (7), five parameters needed to be determined: the number of cells n, the probability for repeated firing p, the firing rate r, the external excitatory input λ^+ and external inhibitory input λ^-. They are: $n \geq 2$, $p \leq 1$, $\lambda^+ > 0, \lambda^- > 0, r > 0$ and $\lambda^- \geq \lambda^+$. For an intuitive illustration of the dense-nuclei activation $q(x) = \zeta(x)$, Figure 2, shows curves of $\zeta(x)$ versus the cluster input x under different values of r, λ^+, λ^- that satisfy their corresponding constraints, with $n = 20$ and $p = 0.05$.



(a) $r = 0.001,\ \lambda^+ = \lambda^- = 0.005$ (b) $r = 0.001,\ \lambda^+ = \lambda^- = 0.0005$ (c) $r = 0.1, \lambda^+ = \lambda^- = 0.5$



(d) $r = 10, \lambda^+ = \lambda^- = 0.5$

Fig. 2. Cluster activation $\zeta(x)$ versus input x under different parameters for a twenty-cell cluster ($n = 20$) with $p = 0.05$.

3.1 The Detailed Training Procedure

The weights W_l $(l = 1, \cdots, L-1)$ are determined by solving a reconstruction problem:

$$\min_{W_l} ||X - \text{adj}(\zeta(X\bar{W}))W_l||^2 + ||W_l||_{\ell_1},\ \text{s.t. } W_l \geq 0, \tag{8}$$

where $\bar{W} \geq 0$ is randomly generated, operation $\text{adj}(X)$ first maps its input into $[0\ 1]$ linearly, then uses the "zcore" MATLAB operation and finally adds a positive constant to remove negativity. The fast iterative shrinkage-thresholding algorithm (FISTA) in [2] is used to solve Problem (8) for W_l with the modification of setting negative elements in the solution to zero in each iteration. Weight matrix W_L is randomly generated in range $[0\ 1]$, while W_{L+1} is determined by the Moore-Penrose pseudo-inverse [21,25,35,38,40, 41] (denoted by "pinv"). The improved training procedure for the RNN-MLA is shown in Algorithm 1, where operation $\max(\cdot)$ produces the maximal element of its input. Note that, following the constraints deduced in Subsection **??**, parameters n, p, and r for the clusters are selected as $n = 20$, $p = 0.05$ and $r = 0.001$. In addition, numerous numerical tests show that $0.01, 0.005$ are generally good choices for $\lambda = \lambda^+ = \lambda^-$.

4 RNN-MLA for Multi-Channel Classification Datasets

We now adapt the RNN-MLA structure for multi-channel classification datasets, naming this structure the MCRNN-MLA. The use of the MCRNN-MLA is then shown on both multi-channel image and real-world classification datasets. For ease of illustration, we consider a 2-channel dataset (Channel-1 and 2) for a 2-channel L-hidden-layer MCRNN-MLA shown in Figure 3.

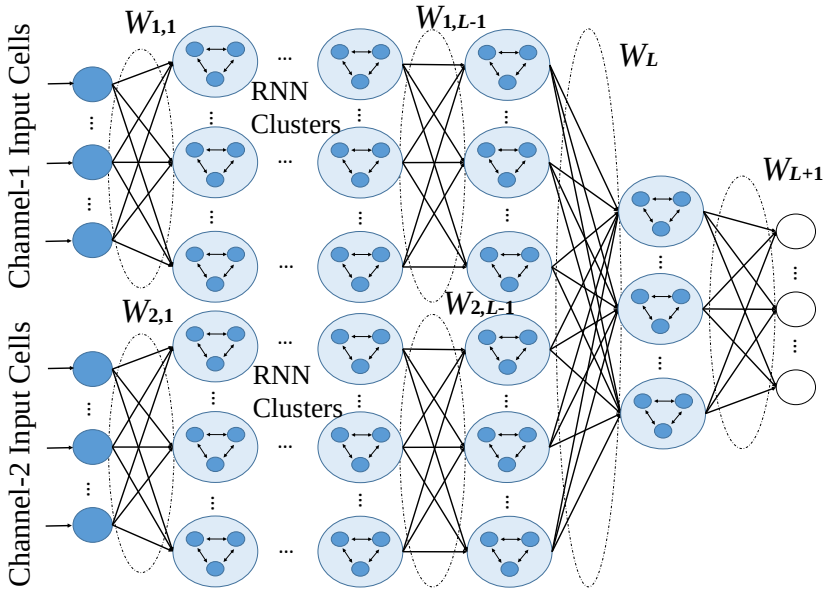


Fig. 3. Schematic representation of the MCRNN-MLA.

Let us denote the connecting weights between layers for only Channel-1 by $W_{1,1},\cdots,W_{1,L-1}\geq 0$, those for only Channel-2 by $W_{2,1},\cdots,W_{2,L-1}\geq 0$, those between the $L-1$ and L hidden layers by $W_L\geq 0$ and output weights by W_{L+1}. The weights $W_{c,l}\geq 0$ ($c=1,2; l=1,\cdots,L-1$) are determined by solving an reconstruction problem using the modified FISTA (described in Section 3):

$$\min_{W_{c,l}} ||X_c - \text{adj}(\zeta(X_c\bar{W}))W_{c,l}||^2 + ||W_{c,l}||_{\ell_1},\ \text{s.t. } W_{c,l}\geq 0, \tag{9}$$

where X_c is either the data from Channel-c or its layer encodings. The training procedure of a C-channel L-hidden-layer MCRNN-MLA is shown in Algorithm **??**.

4.1 Modifications to the MCRNN-MLA

We modify the MCRNN-MLA, calling it the MCRNN-MLA1, where the schematic representation of a C-channel L-hidden-layer B-branch MCRNN-MLA1 is shown in Figure 4. Let us denote the connecting weights to the lth hidden layer for Channel-c of Branch-b by $W_{c,l,b}\geq 0$ ($c=1,\cdots,C; l=1,\cdots,L-1; b=1,\cdots,B$), those for all channels between the $L-1$ and L hidden layers by $W_L\geq 0$ and output weights by W_{L+1}. The training procedure of the MCRNN-MLA1 is detailed in Algorithm **??**.

The second modification MCRNN-MLA2, is a simplified MCRNN-MLA1 obtained by removing the last hidden layer of the MCRNN-MLA1 that produces random mappings via random connections W_L. The schematic representation and training procedure are omitted because they are similar to the ones for the MCRNN-MLA1.

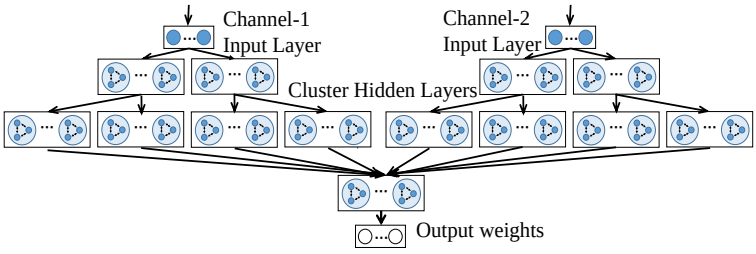


Fig. 4. Schematic representation of the MCRNN-MLA1.

5 Numerical Result Comparisons

We now move to numerical tests that use two multi-channel classification datasets: an image dataset and a real-world time-series. In the numerical experiments, we use the MCRNN-MLA, MCRNN-MLA1, MCRNN-MLA2, RNN-MLA with Algorithm 1, as well as the algorithm that was reported in [21], and the multi-layer perception (MLP), the convolutional neural network (CNN) and hierarchical extreme learning machine (H-ELM) [35]. The experiments are run in a personal computer with Intel i7-6700K CPU (4.00GHz), 16GB memory and GeForce GTX 1080 GPU. The MCRNN-MLA, MCRNN-MLA1, MCRNN-MLA2, RNN-MLA and H-ELM are implemented using MATLAB with only CPU. The MLP and CNN are implemented using Keras [5] with either Theano [36] or Tensorflow [1] as the backend: when the backend is Theano, only CPU is used; while Keras with Tensorflow uses both CPU and GPU.

5.1 Application 1: Recognizing 3D Objects

NORB Dataset The small NORB dataset [27] is intended for experiments in 3D object recognition from shape. The objects belong to 5 generic categories: four-legged animals, human figures, airplanes, trucks and cars. The instance numbers for both training and testing are 24300. There are two 96×96 images in each instance which are downsampled into 32×32. Examples of the two images of each class are given in Figure 5. All images are whitened using the code provided by [35]. The objective is to recognize the generic category of a given instance using the corresponding two images. The number of channels is 2.

Experimental Settings The structures for the RNN-MLA, MCRNN-MLA, MCRNN-MLA1, MRCRNN-MLA2, H-ELM and MLP (Tanh activation) are respectively 2048-500-2000-5, 2x1024-2x500-2000-5, 2x1024-80x2x100-10x160x10-8000-5, 2x1024-2x2x500-2x4x500-5, 2048-1000-1000-12000-5 and 2048-500-500-5. The CNN with Tanh activation is constituted by two convolution layers, a pooling layer, a fully-connected layer and an output layer (denoted as "input-conv-conv-pool-fc-output") and each convolution layer has 30 filters with kernel size 2x2. The dropout technique developed to prevent overfitting [33] is exploited for the MLP and CNN. The Adadelta optimizer [39] is used to train the MLP and CNN; the training methods for the rest tools are specified in

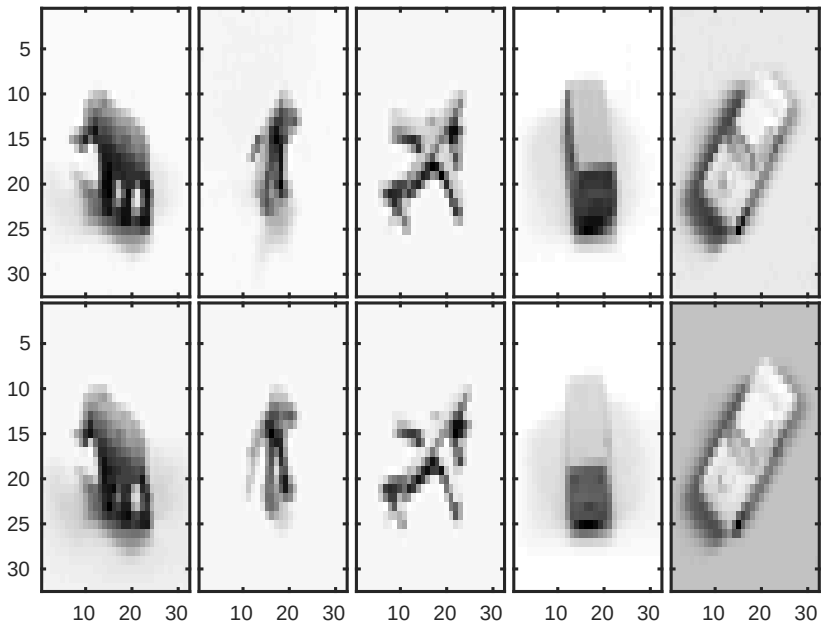


Fig. 5. Examples of two images of each class in the NORB dataset: four-legged animals, human figures, airplanes, trucks and cars.

this paper, [21] and [35]. For each method, five trials are conducted. For the MLP and CNN, there are 50 epochs in each trial and both Theano and Tensorflow are used.

Experimental Results The results of different methods on the training and testing accuracies and the training and testing time are summarised in Table 1. The proposed MCRNN-MLA obtains the highest testing accuracy (91.19%). With the aid of the dropout technique, the CNN achieves a comparable testing accuracy (91.09%) with the MCRNN-MLA. However, the computational complexity is much higher. Specifically, using only the CPU, the proposed MCRNN-MLA can be trained and applied (tested) respectively around 100 and 10 times faster than the CNN. Aided by the computational power of the GPU, the training of the CNN is still more than 15 times slower than that of the MCRNN-MLA. The testing accuracy of the improved RNN-MLA is slightly lower than that of the MCRNN-MLA, so are the training and testing time. Therefore, both the improved RNN-MLA and MCRNN-MLA proposed in this paper are good classifiers for the NORB dataset with high accuracy and training efficiency.

5.2 Application 2: Distinguishing Chemical Gases

Twin Gas Sensor Arrays (TGSA) Dataset The TGSA dataset includes 640 recordings of 5 twin 8-sensor detection units (Units 1 to 5) exposing to 4 different gases (Carbon

Table 1. Training and testing accuracies (%) and time (s) of different methods on NORB dataset for recognizing 3D objects.

Method	Accuracies (%)		Times (s)	
	Training	Testing	Training	Testing
MLP	89.43 ±10.49	75.70 ±9.07	424.60 ±37.47	1.46 ±0.09
MLP+dropout	82.19 ±17.54	64.61 ±19.08	457.85 ±2.64	2.07 ±0.08
CNN	100.00 ±0	89.59 ±0.24	14754.6 ±17.21	64.59 ±0.21
CNN+dropout	100.00 ±0	90.61 ±2.06	15001.3 ±28.68	65.67 ±0.36
MLP*	99.40 ±0.52	83.29 ±3.24	119.87 ±5.12	0.80 ±0.17
MLP+dropout*	99.41 ±0.38	83.28 ±1.08	123.44 ±3.16	0.93 ±0.08
CNN*	100.00 ±0	90.31 ±0.35	282.46 ±3.26	2.29 ±0.08
CNN+dropout*	100.00 ±0	91.09 ±0.49	292.23 ±1.64	2.31 ±0.06
H-ELM	99.80 ±0.10	87.36 ±0.85	30.79 ±1.41	5.80 ±0.21
Original RNN-MLA	99.94 ±0.03	87.11 ±0.44	12.12 ±0.22	3.85 ±0.19
Improved RNN-MLA	99.99 ±0	90.46 ±0.60	12.91 ±0.31	4.00 ±0.27
MCRNN-MLA	99.99 ±0.01	**91.19** ±0.63	15.11 ±0.26	5.95 ±0.21
MCRNN-MLA1	99.82 ±0.06	90.64 ±0.58	357.11 ±7.60	123.25 ±5.91
MCRNN-MLA2	99.51 ±0.11	90.83 ±0.34	54.89 ±0.39	8.65 ±0.28

*These experiments are conducted using the GPU.

Monoxide (CO), Ethanol, Ethylene and Methane) [8]. The duration of each recording is 600 seconds (100Hz sampling frequency) producing 480,000 (8x600x100) features. We use 30-second segments, and then each instance has 24,000 (8x3000) attributes. Examples of these 30-second segments of each class are given in Figure 6. The objective is to distinguish the gas types using the 30-second segments. The number of channels is 8. The following tasks are considered, in both of which two thirds of instances are used for training while the rest for testing:

- Task 1 (3,029 instances): build a specific classifier for Unit 1 to fulfill the objective.
- Task 2 (12,089 instances): build one classifier for all units to fulfill the objective.
- Tasks 3 to 6: build specific classifiers for Units 2 to 5 to fulfill the objective.

Motivation Due to the inherent variability of chemical gas sensors, a classification model for a specific sensor-array system may not be applicable to another system [8,28]. For example, a classifier built for Unit 1 can not be used for Unit 2 to Unit 5. Therefore, a fast and robust method to build accurate classifiers for sensor-array systems is required.

Experimental Settings The settings of different methods to handle Tasks 1 and 2 of the TGSA dataset are given in Table 2. To fit the time-series data into the CNN, we reshape it from 8x3000 to 8x50x60. The training techniques for these networks are the same as those used in Section 5.1. When Task 1 is handled, the trial number is 5 for the MLP and CNN (80 epoches in each trial). They are activated by the Tanh function. For the rest networks, the trial number is 20. When Task 2 is handled, 5 trials are conducted

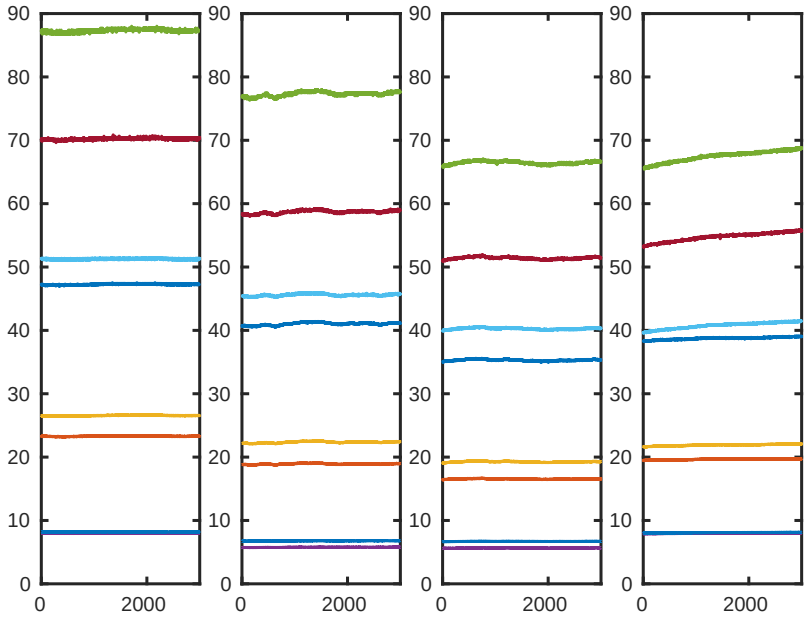


Fig. 6. Examples of eight 30-second segments of each class in the TGSA dataset: Carbon Monoxide, Ethanol, Ethylene, Methane.

for all methods, except the cases where only the CPU is used to train the MLP and CNN, in which the training is slow and one trial is conducted. The settings for Tasks 3 to 6 are the same as the ones for Task 1.

Experimental Results Table 3 presents the results of different methods to handle Task 1. The accuracies of the CNN and MLP are not satisfactory and the training time is long. The MCRNN-MLA1 is the most accurate classifier for Unit 1 (98.88% testing accuracy). However, its testing time is much higher than those of the H-ELM and MCRNN-MLA, where they achieve 96.83% and 98.27% accuracies respectively. Taking both the accuracy and computational complexity into account, the MCRNN-MLA is the best choice for Task 1.

For Task 2 with results given in Table 4, the highest testing accuracy is 93.66%, achieved by the MCRNN-MLA. For the H-ELM, adding more neurons into the last hidden layer may increase its accuracy to handle Task 2. However, its testing time increases as well, where the current one is 2.60s which is already higher than the one of the MCRNN-MLA (2.29s). Larger network sizes for the CNN may also help it to achieve higher accuracies for Tasks 1 and 2. However, the current accuracy gap between the CNN and MCRNN-MLA is large and the testing time (using only the CPU) of the CNN is more than ten times larger than that of the MCRNN-MLA.

The confusion matrixes of the MCRNN-MLA for Tasks 1 and 2 in the testing phase are given in Tables 5 and 6, respectively. For Task 1, Methane has the highest misclassi-

Table 2. Network structure for different methods to handle the TGSA dataset.

Method	Input Layer	Hidden Layers		Output Layer
		Task 1	Task 2	
MLP	24000	500-500	500-500	4
CNN	8x50x60	conv-conv-pool-fc	conv-conv-pool-fc	4
H-ELM	24000	500-500-8000	1000-1000-12000	4
Original RNN-MLA	24000	100-2000	50-5000	4
Improved RNN-MLA	24000	100-3000	50-5000	4
MCRNN-MLA	8x3000	8x200-8x100-2000	8x300-8x50-5000	4
MCRNN-MLA1	8x3000	2x8x200-2x16x100-2000	2x8x200-2x16x100-5000	4
MCRNN-MLA2	8x3000	2x8x200-2x16x100	2x8x200-2x16x100	4

fication rate; while, for Task 2, it is Ethylene. The MCRNN-MLA is exploited to handle Tasks 3 to 5, with results given in Table 7. For all units, the proposed MCRNN-MLA produces high classification rates (from 94.5% to as high as 98.99%) and the training is highly efficient, which can be completed in less than 10 seconds. Therefore we conclude that the MCRNN-MLA is the most efficient among the methods we compare to distinguish chemical gases based on the TGSA dataset.

6 Conclusions

In previous work [21] we proposed the RNN-MLA and demonstrated its usefulness in deep learning. In this paper, we have investigated the concept of dense nuclei of cells. We improved the training procedure for the RNN-MLA and proposed the novel MCRNN-MLA for classifying multi-channel datasets. Comparative numerical experiments were conducted using several different deep-learning methods based on both images and real-world data for multi-channel datasets. The results have shows that the proposed MCRNN-MLA significantly improves upon the results obtained with other methods, both for classification accuracy and efficiency of training.

References

1. Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., Citro, C., Corrado, G.S., Davis, A., Dean, J., Devin, M., Ghemawat, S., Goodfellow, I., Harp, A., Irving, G., Isard, M., Jia, Y., Jozefowicz, R., Kaiser, L., Kudlur, M., Levenberg, J., Mané, D., Monga, R., Moore, S., Murray, D., Olah, C., Schuster, M., Shlens, J., Steiner, B., Sutskever, I., Talwar, K., Tucker, P., Vanhoucke, V., Vasudevan, V., Viégas, F., Vinyals, O., Warden, P., Wattenberg, M., Wicke, M., Yu, Y., Zheng, X.: TensorFlow: Large-scale machine learning on heterogeneous systems (2015), `http://tensorflow.org/`, software available from tensorflow.org
2. Beck, A., Teboulle, M.: A fast iterative shrinkage-thresholding algorithm for linear inverse problems. SIAM journal on imaging sciences 2(1), 183–202 (2009)
3. Bousquet, O., Bottou, L.: The tradeoffs of large scale learning. In: Platt, J.C., Koller, D., Singer, Y., Roweis, S.T. (eds.) Advances in Neural Information Processing Systems 20, pp. 161–168. Curran Associates, Inc. (2008), `http://papers.nips.cc/paper/3323-the-tradeoffs-of-large-scale-learning.pdf`

Table 3. Training and testing accuracies (%) and time (s) of different methods on Task 1 of TGSA dataset for distinguishing chemical gases.

Method	Accuracies (%)		Times (s)	
	Training	Testing	Training	Testing
MLP	24.96 ±0.10	24.95 ±0.10	569.39 ±10.67	0.29 ±0.01
MLP+dropout	24.96 ±0.10	24.95 ±0.10	571.67 ±8.34	0.32 ±0.01
CNN	47.57 ±22.51	47.97 ±22.92	5428.14 ±125.44	6.90 ±0.17
CNN+dropout	52.45 ±27.59	51.98 ±27.13	5604.41 ±122.86	7.36 ±0.17
MLP*	24.96 ±0.10	24.95 ±0.10	81.96 ±0.64	0.15 ±0
MLP+dropout*	24.96 ±0.1	24.95 ±0.10	81.37 ±0.24	0.16 ±0
CNN*	42.79 ±17.93	42.38 ±17.52	98.8 ±0.33	0.22 ±0.01
CNN+dropout*	71.50 ±5.87	71.63 ±6.78	101.25 ±0.18	0.24 ±0.02
H-ELM	98.08 ±0.65	96.83 ±0.61	6.55 ±0.34	0.66 ±0.11
Original RNN-MLA	98.41 ±0.60	91.43 ±1.60	4.55 ±0.13	0.41 ±0.03
Improved RNN-MLA	94.68 ±0.65	91.74 ±0.46	10.54 ±0.13	0.62 ±0.04
MCRNN-MLA	99.85 ±0.10	98.27 ±0.74	9.27 ±0.43	0.37 ±0.03
MCRNN-MLA1	99.59 ±0.05	**98.88** ±0.13	40.90 ±0.43	7.18 ±0.26
MCRNN-MLA2	98.51 ±0.14	94.41 ±0.39	114.6 ±3.84	5.66 ±0.38

*These experiments are conducted using the GPU.

4. Brunel, N., Van Rossum, M.C.: Lapicques 1907 paper: from frogs to integrate-and-fire. Biological cybernetics 97(5-6), 337–339 (2007)
5. Chollet, F.: Keras. `https://github.com/fchollet/keras` (2015)
6. Ciresan, D.C., Meier, U., Gambardella, L.M., Schmidhuber, J.: Deep big simple neural nets for handwritten digit recognition. Neural Computation 22, 3207–3220 (2010)
7. Cramer, C.E., Gelenbe, E.: Video quality and traffic qos in learning-based subsampled and receiver-interpolated video sequences. Selected Areas in Communications, IEEE Journal on 18(2), 150–167 (2000)
8. Fonollosa, J., Fernández, L., Gutiérrez-Gálvez, A., Huerta, R., Marco, S.: Calibration transfer and drift counteraction in chemical sensor arrays using direct standardization. Sensors and Actuators B: Chemical (2016)
9. Gelenbe, E., Gellman, M., Lent, R., et al.: Autonomous smart routing for network qos
10. Gelenbe, E., Iasnogorodski, R.: A queue with server of walking type (autonomous service). Annales de L'Institut Henri Poincaré Section B, Calcul des Probabilités et Statistique 16, 63–73 (1980)
11. Gelenbe, E., Koçak, T.: Area-based results for mine detection. Geoscience and Remote Sensing, IEEE Transactions on 38(1), 12–24 (2000)
12. Gelenbe, E., Koubi, V., Pekergin, F.: Dynamical random neural network approach to the traveling salesman problem. In: Proceedings IEEE Symp. Systems, Man and Cybernetics. p. 630635. IEEE (1993)
13. Gelenbe, E.: Random neural networks with negative and positive signals and product form solution. Neural computation 1(4), 502–510 (1989)
14. Gelenbe, E.: Stability of the random neural network model. Neural computation 2(2), 239–247 (1990)
15. Gelenbe, E.: Steps toward self-aware networks. Communications of the ACM 52(7), 66–75 (2009)

Table 4. Training and testing accuracies (%) and time (s) of different methods on Task 2 of TGSA dataset for distinguishing chemical gases.

Method	Accuracies (%)		Times (s)	
	Training	Testing	Training	Testing
MLP	25.04	25.06	3098.65	1.42
MLP+dropout	24.86	24.83	2741.69	1.41
CNN	69.29	68.69	21577.89	28.16
CNN+dropout	25.04	25.06	23130.27	29.38
MLP*	24.98 ±0.12	24.96 ±0.12	322.76 ±0.55	0.64 ±0.02
MLP+dropout*	25.02 ±0.02	25.04 ±0.01	326.62 ±1.37	0.65 ±0.01
CNN*	55.47 ±30.37	55.16 ±30.08	400.17 ±4.03	0.88 ±0.03
CNN+dropout*	77.13 ±12.70	75.69 ±12.97	417.62 ±11.77	1.06 ±0.11
H-ELM	62.02 ±2.38	61.90 ±2.60	33.98 ±0.66	2.60 ±0.19
Original RNN-MLA	78.97 ±1.48	26.52 ±2.80	48.86 ±0.32	1.60 ±0.08
Improved RNN-MLA	88.07 ±0.53	87.63 ±0.83	48.36 ±0.10	1.67 ±0.09
MCRNN-MLA	93.60 ±0.32	**93.66** ±0.48	53.59 ±1.44	2.29 ±0.16
MCRNN-MLA1	90.97 ±0.18	90.46 ±0.51	62.18 ±0.74	3.90 ±0.18
MCRNN-MLA2	79.83 ±0.23	79.21 ±0.27	32.94 ±1.15	1.96 ±0.20

*These experiments are conducted using the GPU.

Table 5. Confusion matrix of MCRNN-MLA to classify the testing subdataset in Task 1 of TGSA dataset for distinguishing chemical gases.

True	Classified			
	CO	Ethanol	Ethylene	Methane
CO	251	0	0	0
Ethanol	0	253	0	0
Ethylene	0	3	249	1
Methane	6	3	6	238

16. Gelenbe, E., Fourneau, J.M.: Random neural networks with multiple classes of signals. Neural computation 11(4), 953–963 (1999)
17. Gelenbe, E., Hussain, K.F.: Learning in the multiple class random neural network. Neural Networks, IEEE Transactions on 13(6), 1257–1267 (2002)
18. Gelenbe, E., Timotheou, S.: Random neural networks with synchronized interactions. Neural Computation 20(9), 2308–2324 (2008)
19. Gelenbe, E., Timotheou, S.: Synchronized interactions in spiked neuronal networks. The Computer Journal 51(6), 723–730 (2008), https://doi.org/10.1093/comjnl/bxn004
20. Gelenbe, E., Wu, F.J.: Large scale simulation for human evacuation and rescue. Computers & Mathematics with Applications 64(12), 3869–3880 (2012)
21. Gelenbe, E., Yin, Y.: Deep learning with random neural networks. 2016 International Joint Conference on Neural Networks (IJCNN) pp. 1633–1638 (2016)
22. Ghalut, T., Larijani, H.: Non-intrusive method for video quality prediction over lte using random neural networks (rnn). In: Communication Systems, Networks & Digital Signal Processing (CSNDSP), 2014 9th International Symposium on. pp. 519–524. IEEE (2014)

Table 6. Confusion matrix of MCRNN-MLA to classify the testing subdataset in Task 2 of TGSA dataset for distinguishing chemical gases.

True	Classified			
	CO	Ethanol	Ethylene	Methane
CO	965	21	23	1
Ethanol	20	987	2	0
Ethylene	76	19	888	28
Methane	26	0	23	952

Table 7. Training and testing accuracies (%) and time (s) of MCRNN-MLA on Tasks 3 to 6 (Units 2 to 5) of TGSA dataset for distinguishing chemical gases.

Unit	Accuracies (%)		Times (s)	
	Training	Testing	Training	Testing
2	99.36 $\pm$0.25	97.28 $\pm$0.94	9.10 $\pm$0.26	0.37 $\pm$0.03
3	99.85 $\pm$0.10	96.77 $\pm$0.84	9.97 $\pm$1.17	0.39 $\pm$0.07
4	99.85 $\pm$0.05	94.87 $\pm$1.38	7.04 $\pm$0.47	0.19 $\pm$0.02
5	100.00 $\pm$0	98.99 $\pm$0.80	6.70 $\pm$0.17	0.18 $\pm$0.01

23. Hinton, G.E., Salakhutdinov, R.R.: Reducing the dimensionality of data with neural networks. Science 313(5786), 504–507 (2006)
24. Ioffe, S., Szegedy, C.: Batch normalization: Accelerating deep network training by reducing internal covariate shift. CoRR abs/1502.03167 (2015), http://arxiv.org/abs/1502.03167
25. Kasun, L.L.C., Zhou, H., Huang, G.B.: Representational learning with extreme learning machine for big data. IEEE Intelligent Systems 28(6), 31–34 (2013)
26. Larijani, H., Radhakrishnan, K.: Voice quality in voip networks based on random neural networks. In: Networks (ICN), 2010 Ninth International Conference on. pp. 89–92. IEEE (2010)
27. LeCun, Y., Huang, F.J., Bottou, L.: Learning methods for generic object recognition with invariance to pose and lighting. In: Computer Vision and Pattern Recognition, 2004. CVPR 2004. Proceedings of the 2004 IEEE Computer Society Conference on. vol. 2, pp. II–97–104. IEEE (2004)
28. Marco, S., Gutierrez-Galvez, A.: Signal and data processing for machine olfaction and chemical sensing: A review. IEEE Sensors Journal 12(11), 3189–3214 (Nov 2012)
29. Martínez, M., Morón, A., Robledo, F., Rodríguez-Bocca, P., Cancela, H., Rubino, G.: A grasp algorithm using rnn for solving dynamics in a p2p live video streaming network. In: Hybrid Intelligent Systems, 2008. HIS'08. Eighth International Conference on. pp. 447–452. IEEE (2008)
30. Mohamed, S., Rubino, G.: A study of real-time packet video quality using random neural networks. IEEE Transactions on Circuits and Systems for Video Technology 12(12), 1071–1083 (2002)
31. Radhakrishnan, K., Larijani, H.: Evaluating perceived voice quality on packet networks using different random neural network architectures. Performance Evaluation 68(4), 347–360 (2011)

32. Rubino, G., Varela, M.: A new approach for the prediction of end-to-end performance of multimedia streams. In: Quantitative Evaluation of Systems, 2004. QEST 2004. Proceedings. First International Conference on the. pp. 110–119. IEEE (2004)
33. Srivastava, N., Hinton, G.E., Krizhevsky, A., Sutskever, I., Salakhutdinov, R.: Dropout: a simple way to prevent neural networks from overfitting. Journal of Machine Learning Research 15(1), 1929–1958 (2014)
34. Stafylopatis, A., Likas, A.: Pictorial information retrieval using the random neural network. IEEE Transactions on Software Engineering 18(7), 590–600 (1992)
35. Tang, J., Deng, C., Huang, G.B.: Extreme learning machine for multilayer perceptron. IEEE transactions on neural networks and learning systems 27(4), 809–821 (2016)
36. Theano Development Team: Theano: A Python framework for fast computation of mathematical expressions. arXiv e-prints abs/1605.02688 (May 2016), http://arxiv.org/abs/1605.02688
37. Timotheou, S.: The random neural network: a survey. The computer journal 53(3), 251–267 (2010)
38. Yin, Y., Zhang, Y.: Weights and structure determination of chebyshev-polynomial neural networks for pattern classification. Software 11, 048 (2012)
39. Zeiler, M.D.: ADADELTA: an adaptive learning rate method. CoRR abs/1212.5701 (2012), http://arxiv.org/abs/1212.5701
40. Zhang, Y., Yin, Y., Guo, D., Yu, X., Xiao, L.: Cross-validation based weights and structure determination of chebyshev-polynomial neural networks for pattern classification. Pattern Recognition 47(10), 3414–3428 (2014)
41. Zhang, Y., Yin, Y., Yu, X., Guo, D., Xiao, L.: Pruning-included weights and structure determination of 2-input neuronet using chebyshev polynomials of class 1. In: Intelligent Control and Automation (WCICA), 2012 10th World Congress on. pp. 700–705. IEEE (2012)