Towards Using Reservoir Computing Networks for Noise-Robust Image Recognition

Azarakhsh Jalalvand, Wesley De Neve, Rik Van de Walle, Jean-Pierre Martens
Data Science Lab, ELIS, Ghent University–iMinds, B–9000 Ghent, Belgium
Email: Azarakhsh.Jalalvand@UGent.be

***Abstract*—Reservoir Computing Network (RCN) is a special type of the single layer recurrent neural networks, in which the input and the recurrent connections are randomly generated and only the output weights are trained. Besides the ability to process temporal information, the key points of RCN are easy training and robustness against noise. Recently, we introduced a simple strategy to tune the parameters of RCN resulted in an effective and noise-robust RCN-based model for speech recognition. The aim of this work is to extend that study to the field of image processing. In particular, we investigate the potential of RCNs in achieving a competitive performance on the well-known MNIST dataset by following the aforementioned parameter optimizing strategy. Moreover, we achieve good noise robust recognition by utilizing such a network to denoise images and supplying them to a recognizer that is solely trained on clean images. The conducted experiments demonstrate that the proposed RCN-based handwritten digit recognizer achieves an error rate of 0.81 percent on the clean test data of the MNIST benchmark and that the proposed RCN-based denoiser can effectively reduce the error rate on the various types of noise.**

***Index Terms*—Reservoir computing networks, recurrent neural networks, text recognition, image classification, image denoising**

I. Introduction

Thanks to the advances in the structure of the neural networks, such as Convolutional Neural Network (CNN) [1] and Deep Neural Networks (DNN), along with the more powerful computational hardware, image processing have become more elegant than ever. For instance, in many devices the traditional keyboards are being replaced with more stylish modes such as touchscreens that handle the handwriting to input text. In this regard, noise and the variability in the images of the same character are among the challenges that the automatic handwriting recognition (HWR) system may have to deal with.

Recently, new training methods, permitting a better exploitation of multiple hidden layers, were discovered and gave rise to the emergence of DNNs, coming in different flavors such as Deep Belief Networks (DBNs) [2] and Deep Boltzmann Machines (DBMs) [3].

It is, however, generally acknowledged that conventional and modern neural networks such as DNNs perform well, but that they are still hard to train; it takes a lot of time and the hyperparameters of the training process must be set properly. More recent approaches such as *dropout* [4] and *maxout* [5] are examples of efforts to both facilitate improved training and improved effectiveness of the models. Consequently, it was possible to achieve some impressive results in, for example, traditional isolated and clean digit dataset MNIST [1][1].

Despite of good performance in clean condition, many of these approaches still dramatically fail to recognize digits from noisy samples. In [2], for instance, it was shown that the digit error rate of a DBN, trained on clean samples, increases from 1.03% for clean test digits to 33.8% when the digits are partially masked by square blocks and to 66.1% when the digits are surrounded by a black border (see Fig. 1 and Table III). These results were improved to 8.7% and 1.9%, respectively, by training the DBN with noisy images. Examples of the noise types are depicted in Fig. 1. In another study [6], [7], a stacked sparse DBN-based denoising auto-encoder (SSDA) is trained to denoise the images.

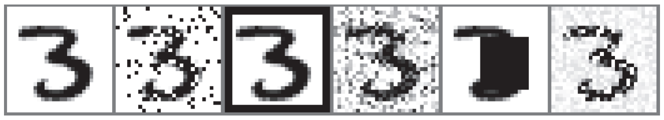


Fig. 1. From left to right, a clean MNIST sample and its corresponding noisy versions: salt & pepper, border, Gaussian, block, and speckle, respectively.

Besides the noise-robustness, incapability of processing the temporal information is another challenge in expanding the application of these techniques to process sequential data such as continuous text and videos. Some more complex CNNs have been proposed to address this weakness with the purpose of video classification [8].

The aim of the present paper is to show the potential of reservoir computing networks (RCNs) [9] in achieving good performance in handwriting recognition from noise corrupted images. RCNs, as a special type of recurrent neural networks, are shown to offer an elegant and noise robust alternative model in the field of speech recognition and enhancing speech features [10], [11] with a very simple and yet effective training procedure.

An RCN is peculiar in the sense that it consists of a hidden layer of recurrently connected non-linear neurons with fixed (that means non-trained) coefficients – called a reservoir – and an output layer of linear neurons with trained coefficients which 'read out' the outputs of the reservoir.

The rest of this paper is organized as follows. Section II briefly recalls the general principles underlying RCNs. Then,

[1]Some reference results along with results produced by this research effort have been listed in Table II.

we propose RCN architectures for performing HWR (Section III). In Sections IV and V, we describe an experimental study of these architectures for the recognition of clean handwritten digits. In the second part of the paper, we focus on the noisy digit recognition (Section VI). The paper ends with a number of conclusions, as well as a number of ideas for future research.

II. Reservoir Computing Network (RCN)

In its simplest form, an RCN is a neural network with two particular computational layers: (1) a hidden layer (pool) of recurrently interconnected non-linear neurons, called *reservoir*, driven by inputs and by delayed feed-backs of its outputs and (2) an output layer of linear neurons, called *readouts*, driven by the hidden neuron outputs (Fig. 2). A fundamental point is that the input weights and the recurrent weights are initialized from random distributions, and only the output weights are optimized (trained) for solving the targeted problem.

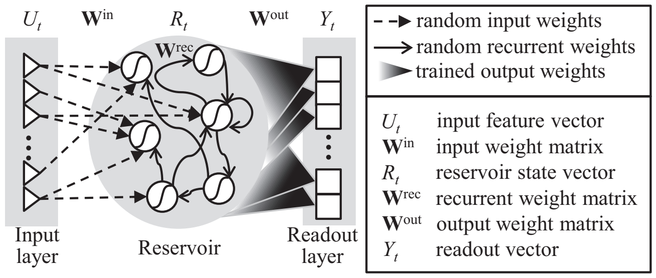

U_t
$\mathbf{W}^{\text{in}}$
R_t
$\mathbf{W}^{\text{out}}$
Y_t
$\mathbf{W}^{\text{rec}}$
Input layer
Reservoir
Readout layer
random input weights
random recurrent weights
trained output weights
U_t input feature vector
$\mathbf{W}^{\text{in}}$ input weight matrix
R_t reservoir state vector
$\mathbf{W}^{\text{rec}}$ recurrent weight matrix
$\mathbf{W}^{\text{out}}$ output weight matrix
Y_t readout vector

Fig. 2. A basic RCN consists of a reservoir and a readout layer. The reservoir is composed of interconnected **non-linear** neurons with **fixed** random weights. The readout layer consists of **linear** neurons with **trained** weights.

If U_t, R_t and Y_t represent the reservoir inputs, the reservoir outputs and the readouts at time t, the RCN equations can be written as follows [9]:

$$R_t = (1-\lambda)R_{t-1} + \lambda\, f_{res}(\mathbf{W}^{in}U_t + \mathbf{W}^{rec}R_{t-1}) \qquad (1)$$

$$Y_t = \mathbf{W}^{out}R_t \qquad (2)$$

with $0 < \lambda \leq 1$, with f_{res} being the non-linear activation function of the reservoir neurons (*hyperbolic tangent* in this work) and with $\mathbf{W}^{in}$, $\mathbf{W}^{rec}$ and $\mathbf{W}^{out}$ being the input, recurrent and output weight matrices, respectively.

Each individual input is normalized so that it has a zero mean and unit variance over the training examples. The initialized input and recurrent weights to the reservoir nodes are assigned from a normal distribution and they are characterized by four parameters [11]: (1) the largest singular value of the input weight matrix $\mathbf{W}^{in}$, (2) the maximal absolute eigenvalue of the recurrent weight matrix $\mathbf{W}^{rec}$, (3) the number of inputs driving each reservoir neuron and (4) the number of delayed reservoir outputs driving each reservoir neuron. The first two parameters control the absolute and the relative importance of the inputs and the delayed reservoir outputs in the reservoir neuron activation. The latter two control the sparsity of the input and the recurrent weight matrices.

The output weights are such that they minimize the mean squared error (MSE) between the *computed* readouts Y_t and the *desired* readouts D_t over the training examples. If an RCN is trained for recognition, the desired output D_t is a unit vector with one non-zero entry encoding the desired class at time t. If it is trained for feature denoising, D_t is the desired clean feature vector at time t. In both cases, the output weights emerge as the solution of an over-determined set of linear equations.

An RCN can be considered as an extension of the Extreme Learning Machine (ELM) proposed in [12]. An ELM is a two-layer MLP with a randomly fixed hidden layer of non-linear neurons followed by an output layer of linear neurons whose weights are determined so as to minimize the mean squared difference between the computed and the desired outputs. According to [11]–[13], the non-trained random weights, in combination with the MSE optimization criterion, prevent RCNs and ELMs from overfitting the training data and hence, let the system generalize better to unseen data than a system with a fully trained parameters.

III. RCN-based architectures for image processing

In many neural network-based image processing systems, the input is a pixel array of the whole image [7], [14]. However, in order to exploit the dynamical system properties of an RCN, we need to create a sequential input stream. This can be achieved by scanning the image column-wise (horizontal scanning) or row-wise (horizontal scanning).

The readouts of the RCN that will be encompassed in the recognizer correspond to the ten digits and to the white space which is present in each digit image. By introducing this white space and by envisioning an image as a digit surrounded by white space, we can achieve that the digit readouts will mainly react to features that are typical for the digit they represent.

A. Basic architecture

A trivial procedure leading to the desired input stream is horizontal scanning: the image is scanned column-wise from left to right and the subsequent columns (called frames) form the input vector sequence (see Fig. 3).

The digit scores are obtained by accumulating the digit readouts across time (the Σ component) and a winner-take-all algorithm returns the winner digit with the highest readout activity.

One could also benefit from bi-directional processing [11], which means that each RCN contains two reservoirs; The forward reservoir that processes the inputs $U_{1\to T}$ whereas the backward reservoir that processes the inputs $U_{T\to 1}$. The outputs of the latter reservoir are then time reversed before combining them with the outputs of the forward reservoir. A deep (cascade) RCN is obtained by stacking multiple RCNs, as depicted in Fig. 3. Each layer of the deep RCN is a basic bi-directional RCN.

The layers are trained one after the other using the same desired outputs in every layer. The argument for cascading

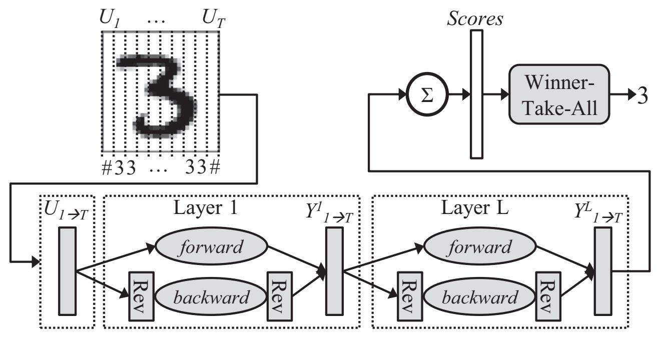


Fig. 3. Architecture of a deep RCN-based digit recognizer leveraging bi-directional processing in each layer.

layers is that the new layer can correct some of the mistakes made by the preceding layers because it offers additional temporal modeling capacity and a new inner space in which to perform the classification.

B. More complex architectures

Given that it is also suitable for continuous HWR, horizontal image scanning seems to be an obvious choice. However, for isolated digit recognition, one can also consider vertical scanning, as well as a combined scanning approach. The ones we propose here are depicted in Fig. 4.

Combination of input features: A simple combination strategy is to supply the RCN with the concatenation of one row and one column at each time step (see Fig. 4(a)). Obviously,

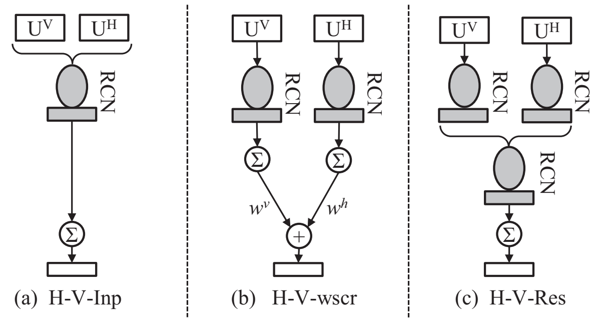


Fig. 4. Different ways of combining horizontal (H) and vertical scanning (V) in a system: (a) supply the RCN with one row and one column of the image, (b) compute a weighted sum of the digit scores (accumulations over time) emerging from an H-RCN and a V-RCN and (c) supply the H-RCN and V-RCN outputs to another RCN and accumulate the scores of the readouts of this RCN.

this approach presumes a square image, leading to an identical number of frames per scanning direction.

Weighted sum of scores: Another strategy is to make two independent parallel systems: one using horizontal (H-RCN) and one using vertical scanning (V-RCN). The digit scores can then be obtained as a linear combination of the scores emerging from the two sub-systems (see Fig. 4(b)). The advantage of this approach is that it can also be applied to rectangular images.

Combination of readouts: The third option is to supply the combined readouts of the V-RCN and the H-RCN to the final digit recognition RCN (see Fig. 4(c)). Obviously, this approach again presumes a square image.

IV. Experimental setup

In this section, we present the experimental framework that was set-up in order to assess the potential of the proposed system configurations.

A. MNIST corpus

The MNIST corpus [1] consists of clean handwritten isolated digit samples, grouped into two datasets: a training set consisting of 60K samples and a test set consisting of 10K samples. Each sample is represented by a 28×28 gray-scale encoded pixel array. The original pixel codes (between 0 and 255) are interpreted as real numbers between 0 and 1. Many studies sub-divide the development set into a training set of 50,000 images and a validation set of 10,000 images. We report the digit error rate (DER%) on the validation or test set as the recognition performance measure.

Some studies extend the training dataset by deforming the original training images and by considering the deformed images as extra training examples, but here we refrain from doing so because our main objective is to show that an RCN-based system has potential to become an alternative to other state-of-the-art systems and it is difficult to make a fair comparison of results obtained with an extended training set without knowing exactly which deformations were applied in the systems one wants to compare with.

In order to conduct experiments on noise robustness, we construct a multi-condition dataset by dividing the dataset into six equally large parts. One part is left unaltered and serves as a clean dataset. The images of the other five parts are corrupted with noise, one noise type per part. As in [2] and [7], the considered noise types are Gaussian, Salt & Pepper, Speckle, Block and Border.

The front-end scans the image either horizontally (H) or vertically (V) and per scanning step t, the column vector (if H) or the row vector (if V) is a 28-dimensional vector X_t. However, it is common in neural networks to obtain the input feature vector, U_t, by extending X_t with its first and second derivatives in the scanning direction, or by stacking the vectors $X_{t-k}, .., X_{t+k}$. Both approaches have the advantage of providing the system with a glimpse of the future. In our experiments, we use frame stacking with $k = 2$.

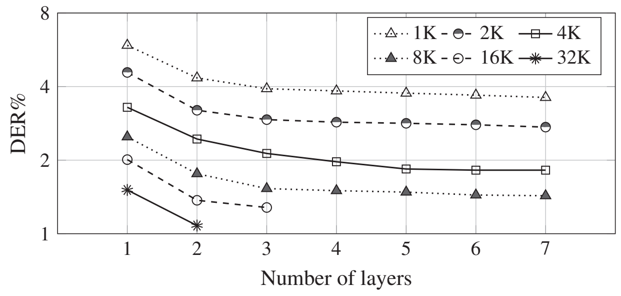
V. Experimental results

In this section, we assess the performance of our systems as a function of the reservoir size (the number of neurons in the reservoir), the depth of the RCN (the number of layers) and the direction of scanning in the front-end. Unless stated otherwise, all RCNs are bi-directional and an RCN with a reservoir of size N actually encompasses two independent reservoirs of size $N/2$ working in parallel. The number of trainable parameters of such an RCN is the dimension of W^{out} which connects the reservoir nodes to the readouts. In this digit recognition task, there are 11 classes (ten digits and the white space) leading to $11 \times (N+1)$ trainable parameters, in which the extra 1 represents a bias for each readout node.

A. Deep versus wide

First, we compare single- and multi-layer RCNs with horizontal image scanning. In multi-layer RCN, the reservoir size is kept the same in each layer. The results depicted in Fig. 5 support the following conclusions:

- Any single-layer system can be improved by adding extra layers and the relative reduction of the DER that can be attributed to adding a second layer is about 25%, irrespective of the reservoir size.
- The positive impact of adding layers decreases quickly with the depth of the RCN. In general, there is no point in creating systems with more than three layers.
- Even though a multi-layer system does not yield a much lower DER than a single-layer system encompassing the same number of trainable parameters, the former is easier and faster to design. In fact, the memory load and the training time are roughly proportional to the square of the reservoir size, meaning that for the training of one-layer RCN with a 32K reservoir, one needs four times more memory and two times more time than for the training of a two-layer RCN with a 16K reservoir in each layer.



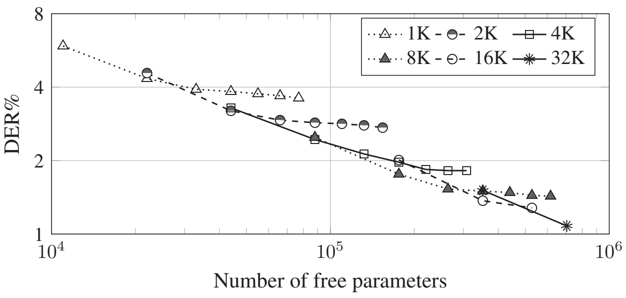


Fig. 5. DER (in %) on the validation set as a function of the reservoir size and the number of layers (top) and the same results, but as a function of the number of trainable parameters (bottom).

B. Scanning directions

In a second experiment, we assess the impact of the image scanning direction and the scanning combination strategy on the system performance.

TABLE I
COMPARING DIFFERENT INPUT SCANNING OPTIONS.

	H	V	H-V-inp	H-V-wscr	H-V-res
DER%	1.52	1.39	1.48	1.30	**1.18**

TABLE II
COMPARING THE PERFORMANCE OF THE RCN WITH THE REFERENCE SYSTEMS ON TRAINED AND TESTED ON THE ORIGINAL MNIST DATASET.

Model	DER%
MLP [15]	1.60
SVM [16]	1.40
MLP + dropout [17]	1.05
DBN [2]	1.03
DBM [3]	0.95
CNN [1]	0.95
MLP + maxout + dropout [5]	0.94
ELM [18]	0.86
RCN (This work)	**0.81**
DBM + dropout [4]	0.79
CNN + maxout + dropout [5]	0.45

Table I lists the results of five systems: (1) H: one 2-layer system with 5K reservoirs and horizontal scanning, (2) V: one 2-layer system with 5K reservoirs and vertical scanning, (3) H-V-inp: one 2-layer system with 5K reservoirs driven by the concatenation of the two scanning directions, (4) H-V-wscr: two 2-layer systems with 2.5K reservoirs (one per scanning direction) whose digit scores are linearly combined, and (5) H-V-res: two 2-layer systems with 2K reservoirs (one per scanning direction) followed by a single-layer system with a 2K reservoir.

As shown by our results, system H-V-res, clearly outperforms both single scanning systems. This indicates that the H and the V readouts for a frame together form a richer feature space for the final classification of the frames.

C. Final result

Based on the above findings, we designed a system of type H-V-res that consists of two 2-layer systems comprising a 16K reservoir in each layer, followed by a single layer RCN encompassing a 16K reservoir. This system has 880K trainable parameters and it achieves a DER of 0.81% on the MNIST test set (see Table II), showing that it is competitive with formerly reported systems working with the same inputs and being trained on the same training samples.

VI. NOISE ROBUSTNESS

In this section, we study the noise robustness of our RCN systems. First, we consider systems that recognize digits from raw noisy images and later, we consider systems that recognize digits from denoised images.

A. Recognition for raw noisy images

In this case, we distinguish two experimental settings: one in which the system is trained on clean images only (clean training) and one in which the system is trained on a mix of

TABLE III

DER (IN %) PER NOISE TYPE FOR THE CASES OF CLEAN AND MULTI-CONDITION TRAINING. THE LAST ROW SHOWS THE DER OF A MULTI-COLUMN RCN-BASED RECOGNIZER COMPRISING TWELVE SUB-SYSTEMS EACH TRAINED ON ONE NOISE CONDITION AND ONE DIRECTION.

	System	Clean	Gaussian	S & P	Speckle	Block	Border	Average
Clean	DBN-2010 [2]	1.03	-	-	-	33.78	66.14	-
	DBN-2013 [7]	1.09	**29.17**	**18.63**	**8.11**	25.72	90.05	**28.80**
	V	1.11	57.04	56.27	72.96	24.97	85.49	49.64
	H	1.28	31.43	40.91	45.91	25.41	**60.99**	34.32
	H-V-res	**0.81**	32.10	38.91	49.32	**21.85**	79.34	37.06
Multi	DBN-2010 [2]	1.68	-	-	-	8.72	1.95	-
	V	1.88	4.73	6.06	7.38	9.50	2.45	5.33
	H	2.28	4.12	5.17	5.65	9.10	2.42	4.79
	H-V-res	**1.50**	**3.08**	**3.75**	**4.32**	**6.82**	**1.75**	**3.54**

clean samples and samples corrupted by the five noise types that are also present in the test set (multi-condition training).

The results of our experiments are listed in Table III. For comparison with the state-of-the-art, the table also includes the results for DBN systems we could find in the literature. In the clean training case, the presence of noise induces a dramatic increase of the DER in all systems. None of the systems stands out on all conditions. The DBN system wins in three of the six conditions, the RCN in the other three, be it that on average the DBN system yields the lowest DER. It is fair to say that RCNs degenerate at more or less the same pace as DBNs when the mismatch between the training and the test conditions increases. We interpret this as a positive result because deep neural networks are acknowledged for their good noise robustness and because the research on RCNs is still in its initial phase.

In the multi-condition training case, the effect of the noise is much more moderate. The H-V-res system now yields an average error rate of only 3.54% and it outperforms the DBN systems in all conditions for which a comparison is possible. Combining two scanning directions seems to help significantly as long as there is no big mismatch between the training and the test conditions (this means clean test for clean training and all tests for multi-condition training). However, more research is needed to establish why the advantage of the combination disappears in mismatched conditions.

B. Recognizing connected digits

As described in Section III, the capability of processing temporal information makes it possible to recognize the digit by scanning the image. Consequently, one can train an RCN by scanning the isolated digits horizontally and operate this system on the connected samples without any extra pre-processing (e.g., digit segmentation). This is a noticeable discrepancy between RCNs and many other conventional neural networks. Fig. 6 depicts the output of a multi-conditionally trained RCN with horizontal scanning (the H system in Table III) which has been supplied with a concatenation of multiple noisy digits.

C. Removing the noise in the front-end

Denoising the input images in the front-end is another approach to reduce the mismatch between training and testing

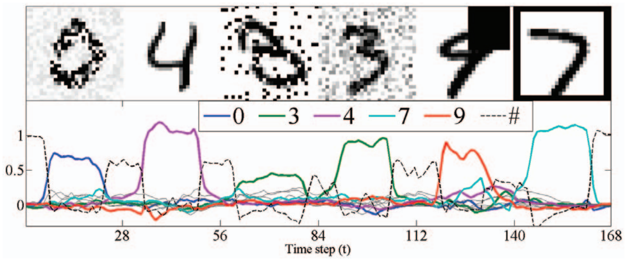


Fig. 6. The readouts of a multi-conditionally trained RCN with horizontal scanning supplied with a concatenation of multiple noisy digits.

conditions. In this phase, we propose an RCN-based denoising Auto-Encoder (DAE) to accomplish this.

For fixing the hyper-parameters of the DAE reservoirs, we follow the same strategy as before, but this time under the assumption that the dynamics of the targeted outputs are identical to the dynamics of the inputs. Moreover, we established that bi-directional processing is also helpful for this task but that it suffices to stack three (instead of five) successive frames in the DAE input. Since the output of the DAE is a denoised version of the input feature vector, the number of trainable parameters of such an RCN-based DAE of the size N is $28 \times (N + 1)$, with 28 being the number of pixels per column/row.

We introduce two DAE architectures: (1) **MixDAE**: a single noise-independent DAE that is trained to remove any kind of noise appeared in the training data and (2) **ComDAE**: a committee of five noise-specific DAEs (one for each noise type), followed by a noise-independent DAE which is driven by the concatenation of the outputs of the former five DAEs.

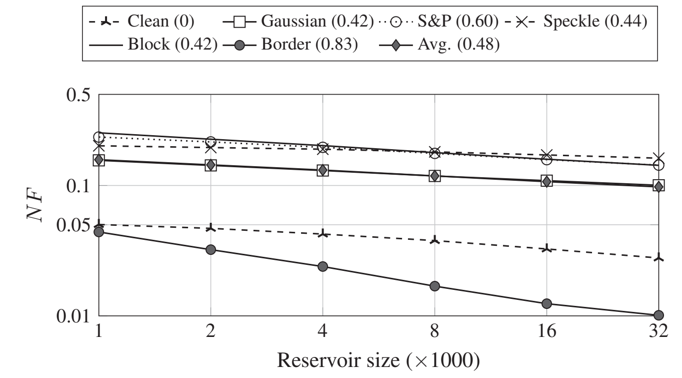
In order to quantify the amount of noise in the image, we define Noise Fraction (NF) as a function of the Pearson correlation coefficient (PCC), with $NF = 1 - PCC^2$. The values of NF are between 0 and 1, with NF = 0 denoting a clean image.

Fig. 7(a) shows the mean NF on the validation set as a function of the reservoir size and the noise type obtained after denoising the image by means of a single-layer MixDAE using horizontal scanning.

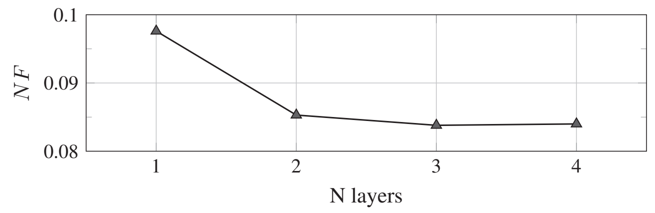
- With a reservoir of size 4K, the mean NF is already

smaller than 0.2 for all noise types. The mean NF is in all cases significantly smaller than the mean NF of the raw noisy images (this mean ranged between 0.4 and 0.83 depending on the noise type).
- The noise reduction improves very gradually as the reservoir size increases. There is no clear bend in the curve for any of the noise types.
- Border noise, the most problematic noise type in the previous experiments, is easy to remove almost completely. This follows from the fact that it is very easy to establish where it occurs and which clean pixel value the DAE has to predict there. Therefore, it is not surprising to find that the NF after denoising of an image corrupted by border noise is even lower than that of a clean image after denoising (there, the DAE output depends on the location of the digit in the image).
- Speckle noise is the only noise type for which the NF is almost independent of the size of the DAE.

The effect of adding layers to the average NF of a single-layer RCN with 32K reservoir is depicted in Fig. 7(b). Adding a second layer clearly induces an additional gain whilst further layers are not beneficial anymore.



(a) Optimizing the reservoir size for a single layer DAE. The figures between brackets in the legend represent the NF of the original noisy images.



(b) Optimizing the number of layers for a DAE with 32K neurons per layer.

Fig. 7. Optimizing the reservoir size and the number of layers for an RCN-based DAE.

Without reporting the results in detail, we mention that neither changing the scanning direction nor combining two scanning directions in an H-V-res like system leads to any significant improvement. Because the aim of denoising is to find and remove the noise patterns and the noise types encountered in this work are direction-independent.

Based on the above findings, we also considered a 2-layer MixDAE with 32K reservoirs in each layer as the reference (1.8M trainable parameters) against which we will compare the ComDAE. To make ComDAE equally complex as the MixDAE (in terms of trainable parameters), the former is composed of five 2-layer noise-specific DAEs with 6K reservoirs per layer and a single-layer noise-independent DAE with a 4K reservoir.

In a control experiment, we also consider an ideal situation by having pre-knowledge about the noise type of the input image. Therefore, we feed the image only to one particular DAE from the first stage of the ComDAE that has been trained for the same noise type. This so-called ideal DAE is denoted as IdlDAE.

Fig. 8 that summarizes the results obtained with these three DAEs, supports the following conclusions: (1) For Gaussian and S&P noise types, the ComDAE achieves a noise reduction that is nearly identical to that of the IdlDAE, but on three other types, the MixDAE is better than the ComDAE. (2) On average, there is little difference between the simple MixDAE and the much more complex ComDAE. Fig. 9 shows the performance of MixDAE on denoising some examples.

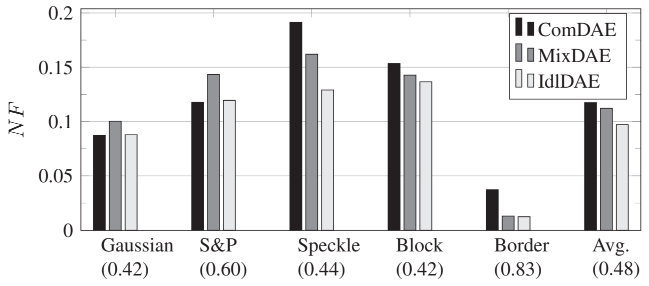


Fig. 8. The noise fraction (NF) for the output of the mixed, the combined and an 'ideal' DAE that has prior knowledge of the noise type. The NF of the raw noisy images are mentioned between brackets.

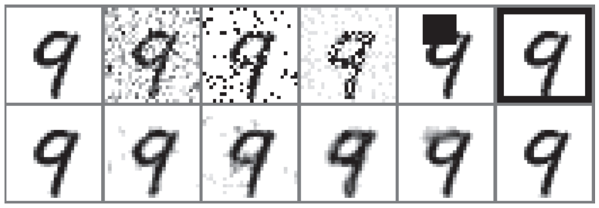


Fig. 9. One clean and five noise corrupted samples of digit 9 (top) and the corresponding outputs of the MixDAE.

D. Recognition for denoised images

In order to evaluate the influence of the RCN-based DAE on the recognition, we test the cascade of the MixDAE and the H-V-res system we formerly trained on clean images. The results obtained with this cascade are listed in Table IV. The table also includes the performance of the adaptive multi-column stacked sparse denoising auto-encoder (AMC-SSDA) reported in [7] and the RBM-based denoiser reported in [2]. It is clear that the

TABLE IV
THE INFLUENCE OF ADDING AN RCN-BASED DAE IN FRONT OF THE CLASSIFIER ON THE PERFORMANCE OF THE RCN-BASED RECOGNIZER (AS DER%) ON THE NOISY VERSION OF THE MNIST DATASET.

Classifier	DAE	Clean	Gaussian	S & P	Speckle	Block	Border	Average
DBN-2010 [2]	RBM-based	1.24	-	-	-	19.09	1.29	-
DBN-2013 [7]	AMC-SSDA	1.50	1.47	2.22	**2.09**	5.18	1.15	2.27
H-V-res	-	**0.81**	32.10	38.91	49.32	21.85	79.34	37.06
H-V-res	MixDAE	1.03	**1.33**	**1.86**	2.41	4.95	**0.89**	2.08
H-V-res (RT)	MixDAE	1.22	1.57	2.17	2.19	**3.94**	1.25	**2.06**

MixDAE introduces a dramatic gain in noise robustness of the H-V-res system at the cost of only a minor loss of accuracy in the case of clean images. Furthermore, the H-V-res system with MixDAE outperforms the AMC-SSDA system in five of the six conditions.

In theory, the just tested configuration is sub-optimal because it implies a mismatch between training and testing. Therefore, we also trained an H-V-res system on denoised training images (called H-V-res (RT)). However, to our surprise, the figures in Table IV show no significant improvement over the sub-optimal system. Apparently, there is no need to retrain the recognizer every time the DAE is improved (e.g., by taking a new noise type into account).

The results obtained for the H-V-res system embedding a mixed DAE show that image denoising in combination with clean training is more effective than multi-condition training, even though the latter is over optimistic because it is tested on noise types that were present during training. This is a remarkable result since a limited study in [2] involving border noise and block noise came to the opposite conclusion for a system encompassing sparse DBNs. In that study, a clean trained DBN, a multi-conditionally trained DBN, and a clean trained DBN supplied with the denoised images led to the DERs of 22.7%, 4.6% and 6.4%, respectively.

VII. CONCLUSION AND FUTURE WORK

The aim of this work was to investigate the potential of reservoir computing networks (RCNs) in the context of image processing, with a particular focus on handwritten digit recognition and image denoising.

Our preliminary study showed that a large enough RCN recognizer can be a competitor to conventional neural network-based recognizers in clean conditions. Moreover, we established that an RCN can be highly effective in denoising an image and that the combination of a denoiser and a recognizer outperforms a similar combination created by means of conventional deep neural network technology. Considering the key points and the performance of RCN, we believe that they are decent candidates to be merged to the conventional DNN-based image and video processing systems.

ACKNOWLEDGMENT

The research activities described in this paper were funded by Ghent University, iMinds, the Institute for the Promotion of Innovation by Science and Technology in Flanders (IWT), the Fund for Scientific Research-Flanders (FWO-Flanders), and the European Union.

REFERENCES

[1] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, “Gradient-based learning applied to document recognition,” *Proc. IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[2] Y. Tang and C. Eliasmith, “Deep networks for robust visual recognition,” in *Proc. ICML*, 2010, pp. 1055–1062.

[3] R. Salakhutdinov and G. E. Hinton, “Deep boltzmann machines,” in *AISTATS*, 2009, pp. 448–455.

[4] G. E. Hinton, N. Srivastava, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, “Improving neural networks by preventing co-adaptation of feature detectors,” *CoRR*, 2012. [Online]. Available: http://arxiv.org/abs/1207.0580

[5] I. J. Goodfellow, D. Warde-Farley, M. Mirza, A. C. Courville, and Y. Bengio, “Maxout networks,” in *Proc. ICML*, 2013, pp. 1319–1327.

[6] J. Xie, L. Xu, and E. Chen, “Image denoising and inpainting with deep neural networks,” in *Proc. NIPS*, 2012, pp. 350–358.

[7] F. Agostinelli, M. Anderson, and H. Lee, “Adaptive multi-column deep neural networks with application to robust image denoising,” *Proc. NIPS*, vol. 26, pp. 1–9, 2013.

[8] A. Karpathy, G. Toderici, S. Shetty, T. Leung, R. Sukthankar, and L. Fei-Fei, “Large-scale video classification with convolutional neural networks,” in *Proc. CVPR*, 2014.

[9] H. Jaeger, “The ‘‘echo state’’ approach to analysing and training recurrent neural networks - with an erratum note,” GMD Report 148, German National Research Center for Information Technology, Tech. Rep., 2001. [Online]. Available: http://www.faculty.jacobs-university.de/hjaeger/pubs/EchoStatesTechRep.pdf

[10] A. Jalalvand, K. Demuynck, and J.-P. Martens, “Feature enhancement with a reservoir-based denoising auto encoder,” in *International Symposium on Signal Processing and Information Technology, Proceedings*. Proc. ISSPIT, 2013, p. 6.

[11] A. Jalalvand, F. Triefenbach, K. Demuynck, and J.-P. Martens, “Robust continuous digit recognition using reservoir computing,” *Computer Speech and Language*, vol. 30, no. 1, pp. 135 – 158, 2015.

[12] G.-B. Huang, Q.-Y. Zhu, and C.-K. Siew, “Extreme learning machine: theory and applications,” *Neurocomputing*, vol. 70, pp. 489–501, 2006.

[13] G.-B. Huang, X. Ding, and H. Zhou, “Optimization method based extreme learning machine for classification,” *Neural Computation*, vol. 74, no. 13, pp. 155–163, 2010.

[14] D. C. Ciresan, U. Meier, L. M. Gambardella, and J. Schmidhuber, “Handwritten digit recognition with a committee of deep neural nets on GPUs,” *Neural Networks Tricks of the Trade*, 2011.

[15] P. Simard, D. Steinkraus, and J. C. Platt, “Best practices for convolutional neural networks applied to visual document analysis,” in *Proc. DAR*, Aug 2003, pp. 958–963.

[16] D. Decoste and B. Schölkopf, “Training invariant Support Vector Machines,” *Machine Learning*, vol. 46, no. 1–3, pp. 161–190, Mar. 2002.

[17] N. Srivastava, “Improving neural networks with dropout,” Master’s thesis, U. Toronto, 2013.

[18] J. Tang, C. Deng, and G.-B. Huang, “Extreme learning machine for multilayer perceptron,” *IEEE Trans. Neural Networks and Learning Systems*, pp. 1–1, 2015.