On the Application of Reservoir Computing Networks for Noisy Image Recognition

Azarakhsh Jalalvand[a], Kris Demuynck[a], Wesley De Neve[a,b], Jean-Pierre Martens[a]

[a]*Data Science Lab, Ghent University–iMinds, B–9000 Ghent, Belgium*
[b]*Image and Video Systems Lab, KAIST, Daejeon, South Korea*

Abstract

Reservoir Computing Networks (RCNs) are a special type of 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 RCNs. Evaluation in the domain of noise robust speech recognition proved that this method was effective. The aim of this work is to extend that study to the field of image processing, by showing that the proposed parameter tuning procedure is equally valid in the field of image processing and conforming that RCNs are apt at temporal modeling and are robust with respect to noise. In particular, we investigate the potential of RCNs in achieving 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 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.

Keywords:
Reservoir computing networks, recurrent neural networks, text recognition, image classification, image denoising

1. Introduction

Thanks to the advances in the structure of the neural networks since the early 80's, such as introducing the concept of Deep Neural Networks (DNNs) [1, 2] and Convolutional Neural Network (CNN) [3] along with the more powerful computational hardware, image processing have become more elegant than ever. For instance, in recent devices the traditional keyboards are being replaced with modern interfaces such as touchscreens that input text via handwriting (e.g., TV touchpad remotes). Due to the different handwriting styles, there is a lot of variability in the images of the same character, making automatic handwriting recognition (HWR) a challenging task.

The presence of background noise is another source of variability the HWR system may have to deal with. In fact, in applications such as address recognition on parcels or full text recognition from digital scans of old manuscripts or typed documents, noise corrupted images such as the one depicted in Figure 1 are the norm.

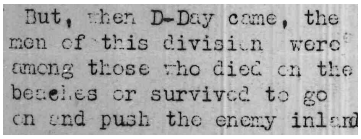


Figure 1: Part of the military newspaper "The Stars and Stripes" published in 1944.

In this paper, we show that reservoir computing networks (RCNs) have great potential for achieving good performance in HWR from noise corrupted images. We demonstrate this on the MNIST [3] dataset, a handwritten digit recognition task (HDR) used by many research groups to benchmark their technologies.

More than two decades ago, Multilayer Perceptrons (MLPs) [4] were among the first classifiers that were

Email address: Azarakhsh.Jalalvand@UGent.be (Azarakhsh Jalalvand)

tested on MNIST. In [3], an MLP with two computational layers of neurons was reported to reach a digit error rate (DER) of 2.95%, and a later study [5] reported a DER of 1.60%. Employing MLPs with more layers was long time believed to yield no significant improvement. However, new training methods, permitting a better exploitation of multiple hidden layers, were recently discovered and gave rise to the emergence of Deep Neural Networks [6]. Differences in the details lead to DNNs of various types. Two of them, Deep Belief Networks (DBNs) and Deep Boltzmann Machines (DBMs) have also been tested on MNIST (see Table 1). Roughly speaking, they achieve a DER of about 1%.

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* [7] and *maxout* [8] are examples of efforts to both facilitate improved training and improved effectiveness of the models.

Nonetheless, long before deep neural networks became successful, significant improvement over a standard 2-layer MLP was achieved by means of a Convolutional Neural Network (CNN) [3] that acts like a feature extractor. In fact, one of the main points of criticism raised against an MLP was that its hidden neurons see the whole image and are therefore bound to overlook the local topological relations between adjacent pixels undoubtedly present in sub-regions of the image [3]. Hence, the idea was to scan the image, to filter the pixels appearing in the emerging sub-regions by means of trainable filters and to down-sample the filtered outputs so as to create a rich and compact feature representation that constitutes a more suitable input to the MLP-based classifier. The first results obtained with the CNN approach were already mentioned in [3]. With a DER of 0.95%, CNN-based systems can still be considered as state-of-the-art for HDR and the CNN-based features are being used in many complex visual understanding models [19].

Obviously, there is no reason why the concepts of a CNN and a DNN could not be combined. Deep Convolutional Neural Networks are the exponent of that idea, leading to a DER of 0.83%.

Another idea that induced a significant boost in HDR, was to enrich the original training dataset with new images, obtained by deforming the raw training images. In [5], for instance, elastic deformations were applied to the raw images achieving a convincing drop in DER from 1.6% to 0.7%. Since then, basically all novel methods have shown to benefit from such an enrichment of the training set (see right column of Table 1). By also introducing separate DNNs for different digit widths (6 classes), it was even possible to achieve human-competitive performance (0.23%) [18].

In spite of the spectacular performances achieved in clean conditions, all aforementioned approaches fail dramatically when recognizing digits from noisy samples. In [12], for instance, it was shown that a DBN trained on clean samples, fails completely when recognizing noisy samples. The DER raises 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 Figure 2). Consequently, new research has been directed towards improving the robustness of HDR against the presence of noise.

In general, one can roughly distinguish three approaches: (1) add noise to the training examples and perform a so-called *multi-conditional training* of the neural network, (2) make the classifier intrinsically more robust against the effects induced by noise, for example, by using a sparsely connected DBN rather than a conventional fully connected one [12], and (3) remove a large part of the noise from the input image before presenting it to the classifier. In [12], it was argued that due to the noise, a lot of neurons are driven into saturation and are therefore not contributing to the recognition anymore. By training it on noisy images, the standard DBN could be made much more effective. The DER could be reduced from 33.8% to 8.7% for the case of block noise and from 66.1% to 1.9% for the case of border noise.

In some other work [20, 21], a stacked sparse DBN-based denoising auto-encoder (SSDA) is trained to denoise the images. In such a system, one SSDA per noise type was trained and the denoised image is obtained as a linear combination of the individual SSDA outputs. Feeding these images to a DBN trained on clean samples induced a dramatic improvement. The average error rate was reduced from 34.3% (an average over five noise types) to 2.4%. Examples of the noise types are depicted in Figure 2 and Table 5 lists the improvements per noise type. As the combination weights are determined by a weight prediction module, the latter system was called an adaptive multi-column SSDA (AMC-SSDA) system.

Besides the noise-robustness, the incapability of processing temporal information is another challenge in expanding the application of these techniques to process sequential data such as continuous text and videos. Long short-term memory systems (LSTM) and some more complex CNNs have been proposed and used to address this weakness with the purpose of motion pic-

Table 1: Reference results on MNIST using the original training set and using an expanded version of the training set (for example, by applying deformation). The presented DERs are accompanied by a reference to the paper introducing the technique that was used.

System	DER% (Original training set)	DER% (Enriched training set)
LSTM[9, 10]	1.80	0.32
2-layer MLP [5]	1.60	0.70
MLP + dropout [11]	1.05	-
DBN [12]	1.03	-
DBM [13]	0.95	-
CNN [3]	0.95	0.80
MLP + maxout + dropout [8]	0.94	-
ELM [14]	0.86	-
DCN [15, 16]	0.83	0.35
DBM + dropout [7]	0.79	-
Large CNN [17]	0.60	0.39
CNN + maxout + dropout [8]	0.45	-
Multi-CNN [18]	-	0.23

Figure 2: From left to right, a clean MNIST sample and its corresponding noisy versions: salt & pepper, border, Gaussian, block, and speckle, respectively.

ture classification [9, 10, 22].

In this paper, the focus is on reservoir computing networks (RCNs) [23], which are a special type of recurrent neural networks. As was shown in [24, 25], RCNs in combination with the proposed simple but effective training procedure, can provide adequate solutions in the field of speech recognition and noise robust speech processing. The aim of this paper is to investigate if the same training procedure is applicable in the domain of image recognition and if the strong points of RCNs such as good temporal modeling and noise robustness transfer to the new domain as well.

Although developed in parallel, on the conceptual level, an RCN can be considered as an extension of the Extreme Learning Machine (ELM) proposed in [26]. An ELM is a two-layer MLP with a randomly initialized and afterwards fixed (i.e. non-trained) 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. Under these constraints, there exists a closed-form solution for the weights which can be obtained by inverting a squared matrix and performing some additional matrix multiplications.

ELMs have been proven to be effective, efficient and robust algorithms for pattern classification. In the past years, several versions of ELMs have been introduced to tackle the different challenges in the field of machine learning. For instance, due to the difficulty in obtaining the labeled data, Huang et, al. [27] proposed a semi-supervised ELM for classification and an unsupervised ELM for clustering. Other solutions to the dilemma of insufficient labeled data are domain adaptation and transfer learning [28]. In this respect, Zhang and Zhang [29] extended ELMs to handle domain adaptation problems for improving the transferring capability of ELM between multiple domains with very few labeled guide instances in target domain, and overcome the generalization disadvantages of ELM in multi-domains application.

An RCN differs from an ELM in the sense that its first layer, called the reservoir, consists of *recurrently connected* non-linear neurons with randomly initilialized and fixed (non-trained) coefficients. Similar to ELMs, the "andom and fixed" first layer feeds into an an output layer of linear neurons with trained coefficients. Since this layer reads from the reservoir to construct its output, it is called the 'read out' layer.

In our previous study, we conceived a straightforward strategy to design an RCN [25]: one only needs to specify the bandwidths of the RCN inputs and outputs, after which the method automatically produces good values for all the hyperparameters of the reservoir component of the RCN. The size of the reservoir remains a free parameter, and its optimal value depends

on the number of available training examples and the envisioned compromise between accuracy and computational cost. This method sped up the design of the robust spoken digit recognizer significantly. The aforementioned observations motivated us to experimentally investigate whether the devised techniques could also be successfully applied in a domain different from speech, namely the visual domain, and in particular handwriting recognition.

A summary of our results with RCNs on handwriting recognition appeared in [30]. This paper extends that contribution by (1) providing an empirical analysis on how to setup RCN architectures for performing HDR, (2) introducing a multi-column RCN-based model to process the noisy images, (3) comparing RCNs and ELMs with regard to their robustness against overfitting, and (4) having a short look at the recognition of connected digits. The remainder of this paper is organized as follows. Section 2 briefly recalls the general principles underlying RCNs. Section 3 introduces the various ways in which the two dimensional image can be fed to a temporal model such as an RCN. Next, we describe our experimental study of these architectures for the recognition of clean handwritten digits (Sections 4 and 5). In the second part of the paper, we focus on the noise-robustness of the proposed RCN architectures (Section 6). The paper ends with a summary and conclusions, as well as some ideas for future research.

2. 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 feedbacks of its outputs and (2) an output layer of linear neurons, called *readouts*, driven by the hidden neuron outputs (Figure 3). 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.

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 [23]:

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

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

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,

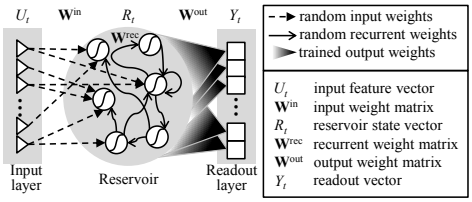


Figure 3: A basic RCN consists of a reservoir and a readout layer. The reservoir is composed of interconnected **non-linear** neurons with **fixed** random weights. The readout layer consists of **linear** neurons with **trained** weights.

respectively. The constant λ is called the leak rate because (if one makes abstraction of f_{res}) Equation (1) represents a leaky integration of the neuron activation.

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 [25]:

- α_U the largest singular value of $\mathbf{W}^{in}$,
- ρ (known as *spectral radius*) the maximal absolute eigenvalue of $\mathbf{W}^{rec}$,
- K^{in} the number of inputs driving each reservoir neuron, and
- K^{rec} 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.

3. RCN-based architectures for image processing

In many neural network-based image processing systems, the input is a pixel array of the whole image [21, 31]. 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.

3.1. 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 Figure 4).

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 [25], which means that each RCN contains two reservoirs: the forward reservoir that processes the inputs $U_{1\rightarrow T}$ whereas the backward reservoir that processes the inputs $U_{T\rightarrow 1}$. 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 Figure 4. Each layer of the deep RCN is a basic bi-directional RCN.

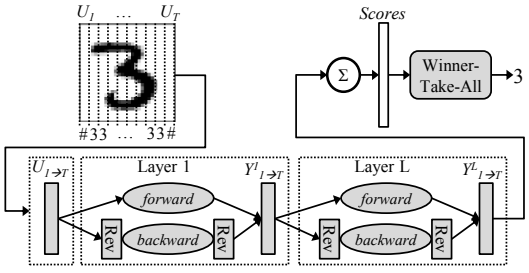


Figure 4: Architecture of a deep RCN-based digit recognizer leveraging bi-directional processing in each layer.

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

3.2. 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 Figure 5.

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 Figure 5(a)). Obviously, this approach presumes a square image, leading to an identical number of frames per scanning direction.

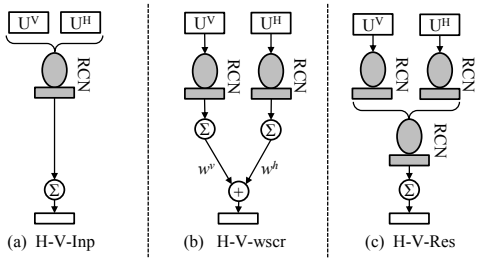


Figure 5: Different ways of combining horizontal (H) and vertical scanning (V) in a system: (a) supply the RCN with one row and one column of the image, (b) compute a weighted sum of the digit scores (accumulations over time) emerging from an H-RCN and a V-RCN and (c) supply the H-RCN and V-RCN outputs to another RCN and accumulate the scores of the readouts of this RCN.

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 Figure 5(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 Figure 5(c)). Obviously, this approach again presumes a square image.

4. 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.

4.1. MNIST corpus

The MNIST corpus [3] 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. The following noise types are chosen to be representatives of the challenges in the common image processing procedures [12, 21]. We consider **Salt & Pepper**, for which a certain amount of the pixels in the image are either black or white. This noise can, e.g., appear in charge coupled device (CCD) sensor outputs or in the transmission of the image. Principal sources of **Gaussian** noise in digital images arise during acquisition e.g. sensor noise caused by poor illumination, high temperature, transmission e.g. electronic circuit noise, etc. **Speckle** is a granular noise that inherently exists in and degrades the quality of the active radar, synthetic aperture radar (SAR), medical ultrasound and optical coherence tomography images. Apart from these technical noise types, usually the objects in an image are partially **blocked** by other elements or the objects are **bordered** in a frame (e.g., house numbers).

4.2. Front-end

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 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 $k = 2$.

4.3. RCN hyperparameters

The creation of a suitable RCN was studied in detail in [25]. In summary, the theory presented there leads to the following conclusions:

1. The input and recurrent weight matrices ($\mathbf{W}^{in}$ and $\mathbf{W}^{rec}$) can be very sparse. In particular, 5 to 10 elements per node are enough, regardless of the reservoir size and the input feature vector size.
2. The spectral radius, ρ, must be tuned to the bandwidth (normalized frequency, F) of the input activations of the reservoir (interpreted as time series):
$$\ln(\rho) = \frac{-F}{0.35} \tag{3}$$
3. The leak rate λ must be tuned to $T_{\min}$, the minimum time (in scan steps) the reservoir output is expected to remain constant:
$$\ln(1-\lambda) = \frac{-1}{T_{\min}} \tag{4}$$
4. The square of α_U follows from a function of the other parameters. This function is proportional to an auxiliary parameter V_{opt}, defined as the optimal reservoir output variance.

In [25] we argue that V_{opt} may be independent of the task, but since the objective of that work was only speech recognition, we did not present any experimental evidence for this argument yet. Here, we will show that $V_{opt} = 0.035$ which was found optimal for spoken digit recognition, is also valid for handwritten digit recognition.

4.4. Design of the reservoir in the first layer

The first step consists of determining the bandwidth of the input activations. In order to do so, it suffices to consider an arbitrary small reservoir (500 nodes) with memory-less neurons (no leaky integration) and no recurrent connection, to record a few hundred input activation patterns of 28 samples long (which is the number of scan steps), to determine the periodogram (the square of the magnitude of the DFT) of each recorded pattern and to calculate the envisioned power spectrum as the mean of these periodograms. To facilitate our experiment, we fed the reservoir with normalized pixels with a unit variance.

Figure 6 (left) shows the estimated power spectrum, $|B(f)|^2$, and its bandwidth, $F = 0.15$. For this value it follows from Equation 3 that $\rho = exp(-0.15/0.35) = 0.65$. Note that there is little difference in bandwidth between horizontal and vertical scanning.

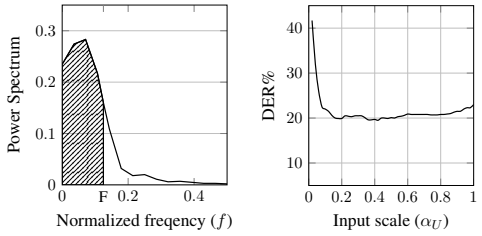


Figure 6: (left) Power spectrum of the input activation and (right) DER as a function of the input scaling factor α_U for a system encompassing a single layer RCN with 500 nodes.

Next, we have to identify the minimum number of scan steps a readout is supposed to remain constant. Given that white spaces are often not more than 4 pixels wide and that the core of a digit like '1' may be as narrow as that as well, we select $T_{\min} = 4$ as a realistic value. For this value, Equation 4 leads to $\lambda = 0.22$.

The third step consists of finding the proper input scale factor α_U. The equation for solving α_U as a function of the other reservoir parameters can be found in [25].

Here we verify whether this function leads to a good result in the simple case of a recognizer built with the reservoir we used for measuring the power spectrum of the input activation. For this reservoir, the function reduces to

$$\alpha_U^2 \; K^{in} \; V_U \; \phi_b(F) = V_{opt} = 0.035 \qquad (5)$$

with $K^{in} = 5$, $V_U = 1$ and

$$\phi_b(F) = \frac{\int_{-F}^{F} |B(f)|^2 \; df}{\int_{-0.5}^{0.5} |B(f)|^2 \; df} = 0.85 \qquad (6)$$

The result is $\alpha_U = 0.28$. In order to verify whether this is a suitable value, we reused the same reservoir in combination with a large number of input scaling factors. Figure 6 (right) shows the DER of the digit recognizer on a validation set as a function of this factor. It appears that 0.28 is in the middle of the optimal range from 0.2 to 0.5.

We also verified whether (ρ, λ) = (0.65, 0.22) were appropriate values. Again, we considered a reservoir with 500 nodes. Since it was already shown in [25] that λ and ρ can be optimized independently of each other, we made two lines of search: one along ρ (while $\lambda = 1$) and one along λ with the optimum ρ from the previous sweep. According to Figure 7, the values originating from the theory are close to the actual optimum points.

For the higher layers, the inputs are always close to the desired outputs. Therefore, after measuring the average bandwidth of the outputs of the first layer, we set $\rho = 0.4$ and $\alpha_U = 0.6$ for all further layers. Since the desired outputs are the same as the first layer, the same $\lambda = 0.22$ was set for the higher layers. A control experiment with a two-layer system confirmed that these settings are appropriate.

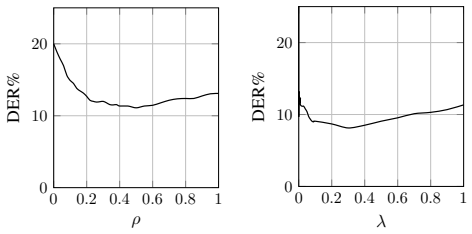


Figure 7: Control experiments to optimize ρ (left) and λ (right).

5. 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 bidirectional 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 $11 \times (N + 1)$, in which the extra 1 represents a bias for each readout node.

5.1. 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 Figure 8 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.

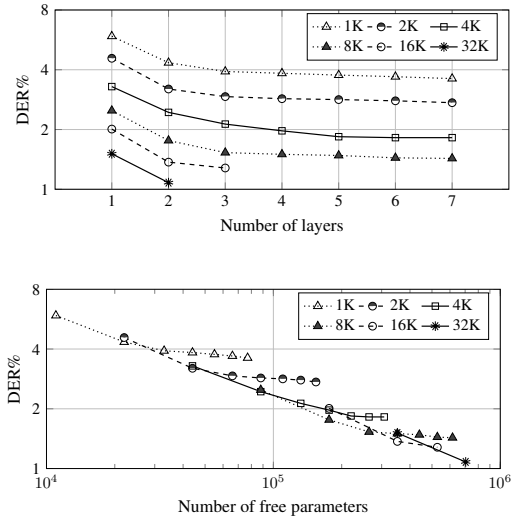


Figure 8: DER (in %) on the validation set as a function of the reservoir size and the number of layers (top) and the same results, but as a function of the number of trainable parameters (bottom).

5.2. Scanning directions

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

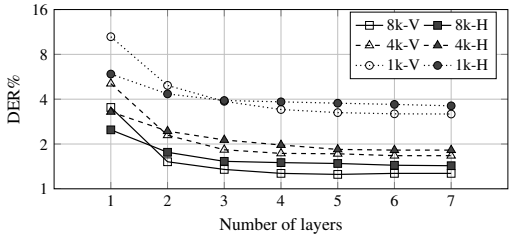


Figure 9: Comparing the effect of horizontal image scanning (H) with the vertical procedure (V) for three multi-layer RC-based systems having 1K, 4K, and 8K nodes per layer.

Table 2: 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**

Figure 9 clearly shows that in the case of a single-layer system, horizontal scanning outperforms vertical scanning; whereas for the deep systems, vertical scanning tends to produce slightly better results. The differences may be due to the fact that most digits occupy a smaller part of the image in horizontal than in vertical direction, as illustrated in Figure 10. This means on the one hand that a single column carries more information about the digit on average than a single row, which is beneficial for horizontal scanning. On the other hand, this means that the digit score in vertical scanning systems is based on more frames; what should normally favor this scanning direction. Apparently, the first phenomenon is more dominant than the second one in a one-layer system, whereas the second phenomenon is more dominant in the multi-layer systems.

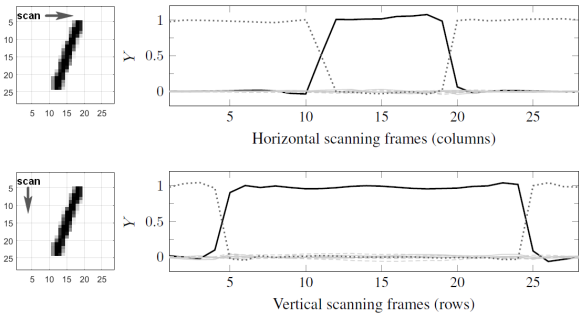


Figure 10: The readouts of reservoirs working with horizontal (top) and vertical (bottom) scanning for a sample of digit 1. The strong black line is the readout of digit 1, the dashed line is the readout of white space class.

Table 2 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

Table 3: Reference results on MNIST using the original training set. The presented DERs are accompanied by a reference to the paper introducing the technique that was used.

Model	DER%
MLP [5]	1.60
MLP + dropout [11]	1.05
DBN [12]	1.03
DBM [13]	0.95
CNN [3]	0.95
MLP + maxout + dropout [8]	0.94
ELM [14]	0.86
RCN (This work)	**0.81**
DBM + dropout [7]	0.79
CNN + maxout + dropout [8]	0.45

a richer feature space for the final classification of the frames.

5.3. 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 3), showing that it is competitive with formerly reported systems working with the same inputs and being trained on the same training samples. As depicted in Figure 11 the most errors occurs for digit '9' being recognized as '4' and '7' being recognized as '2'.

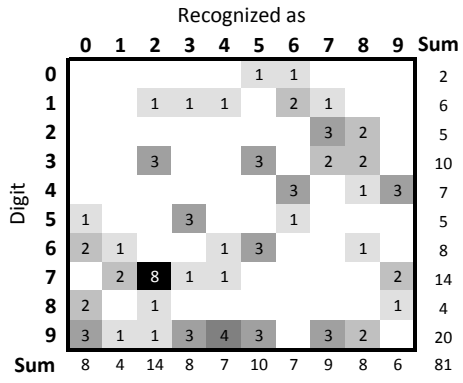


Figure 11: The confusion matrix for the performance of the RCN-based classifier on the MNIST dataset. The most errors occurs for digits '9' and '7', with '7' being recognized as '2'.

6. 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.

6.1. Recognition of noisy images

According to [25, 26, 32], the non-trained random weights, in combination with the MSE optimization criterion, reduces the chance of overfitting the training data in RCNs and ELMs and hence, let the system generalize better to noisy data than a system with fully trained parameters. In what follows, 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 clean samples and samples corrupted by the five noise types that are also present in the test set (multi-conditional training). We present DERs for each of the six subsets of the multi-conditional test set. Note that multi-conditional training is bound to yield an optimistic result as all noise types encountered in the test data were also present during system training. Nevertheless, we followed this recipe to have comparable results with other references.

Preliminary results

With regard to the effect of number of trainable parameters on the robustness of the RCN, we considered the single layer reservoir that scans the image horizontally (the H system) and the number of trainable parameters is controlled by changing the size of the reservoir. The experiments were conducted for both clean and multi-conditional training. Figure 12 shows the performance of RCNs of 500 to 16K nodes, that are tested on clean and a mixture of noisy samples. From these experiences one can conclude that a glimpse of overfitting is visible only for very large RCNs that are trained on clean and tested on noisy samples. This phenomenon was evident in our previous study on speech recognition in [25] only in the extremely mismatched conditions (i.e. clean training and testing on very noisy samples). According to multi-conditional training results, as well as trained-tested on clean samples, no overfitting occurs in case of matched conditions. A conclusion that is inline with other studies on RCNs.

Apart from the aforementioned common characteristics of RCNs and ELMs, RCNs benefit from the recurrent connections that add the ability of processing temporal information in a non-linear way. In order to study the effect of these connections to the robustness of the RCNs, we repeated the same experiments with

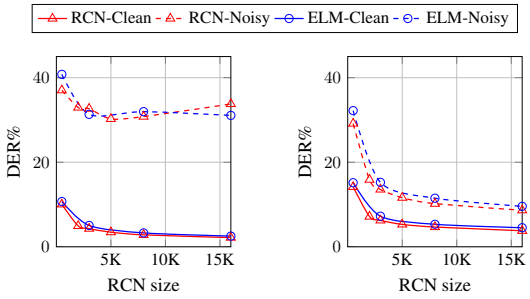


Figure 12: The effect of increasing the size of the hidden layer on the performance of single layer RCN and ELM using clean training (left) as well as multi-conditional training (right). The condition of the test samples (Clean or Noisy) is determined in the legend.

Table 4: Comparing the performance of a single layer 16K-node RCN with an equivalent ELM.

System	Clean training		Multi-cond. training	
	Clean	Noisy	Clean	Noisy
RCN	2.18	33.78	3.80	8.65
ELM	2.54	31.08	4.52	9.58

an equivalent ELM. That is actually the same RCN but without the recurrent connections. The performance of this system is depicted in Figure 12. Also the exact error rates of the 16K-node systems are listed in Table 4. These experiments show that both ELM and RCN follow the same trend of behavior with two main exceptions: (1) trained on clean and tested on noisy samples, the ELM suffers less from the overfitting and (2) RCN outperforms ELM in the matched conditions. Therefore, it can be confirmed that the robustness against overfitting mainly is caused by the random weights and the linear training approach, while the recurrent connections allows the RCN to focus on the less apparent relations between features and classes, and hence help more for matched conditions where such fine differences are more meaningful.

RCN-based noisy image recognizer

The results of our experiments using different RCN-based architectures (explained in Section 3) are listed in Table 5. For comparison with 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-conditional 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 (that means clean test for clean training and all tests for multi-conditional training).

None of the tested systems stands out on all noise types, but on average, the H-V-inp architecture is the winner because there is no noise type for which it completely breaks down. This system performs exceptionally well for the Border noise. The DBN-based system on the other hand is the winner for three of the five noise types.

In order to further investigate the difference between the performances of H-V-inp, H and V, we should take a look at Figure 13 which shows the readouts of the three systems for a noisy sample of digit 7 surrounded by a border. This figure shows that in the case of H and V, the winner is mainly determined by the digit readout that reaches the highest value at the beginning and the end of the scan, where only the black border is observed. In the case of horizontal scanning, this seems to be '1' which often comprise a more or less vertical line that bares a lot of resemblance with the black border. For the H-V-inp system, none of the digit readouts seems to match the black border and the correct winner is more likely to pop-up.

To confirm our hypothesis we investigated in more detail the confusions between the recognized and the correct digit in the case border noise is present. Table 6 shows the digits that were recognized in case of a wrong decision. For instance, it is indicated that with the H-RCN system, 8501 of the validation samples have been wrongly classified as digit 1.

Apparently, the H-system is strongly biased towards the digits exhibiting a strong vertical line ('1' and '4') whilst system V is biased towards digits with horizontal lines on top, bottom or center ('2', '4','5' and '7'). A simple solution to reduce the effect of the false positive reaction to a border, without seriously degrading the performance on the other noise types, is to bound the readouts to an acceptable interval such as (-0.05, 0.25). By doing so, the DER for Border noise can be

Table 5: DER (in %) per noise type for the cases of clean and multi-conditional training. The systems DBN-2010 and DBN-2013 refer to [12] and [21], respectively. (V) and (H) respectively refer to the RCNs that are supplied with vertical and horizontal scanning of the image as the input. Also see Section 3 for the combined approaches. 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	1.03	-	-	-	33.78	66.14	-
	DBN-2013	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-inp	1.18	29.46	40.94	30.70	22.12	**16.97**	**23.56**
	H-V-wscr	0.89	32.12	38.47	48.50	**21.79**	64.94	34.45
	H-V-res	**0.81**	32.10	38.91	49.32	21.85	79.34	37.06
Multi	DBN-2010	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-inp	2.28	4.20	5.22	5.23	8.96	2.63	4.75
	H-V-wscr	1.65	3.12	3.90	4.47	7.20	1.93	3.71
	H-V-res	**1.50**	**3.08**	**3.75**	**4.32**	**6.82**	**1.75**	**3.54**
	MC-RCN	2.82	4.54	6.07	6.22	9.82	3.23	5.45

Table 6: Distribution of the wrong decisions in case of border noise.

	0	1	2	3	4	5	6	7	8	9	Sum
H	0	**8501**	4	0	33	0	0	0	11	1	8549
V	0	0	**5180**	13	69	**788**	0	50	0	0	6099
H-V-inp	2	**1533**	2	1	98	17	0	41	2	1	1697

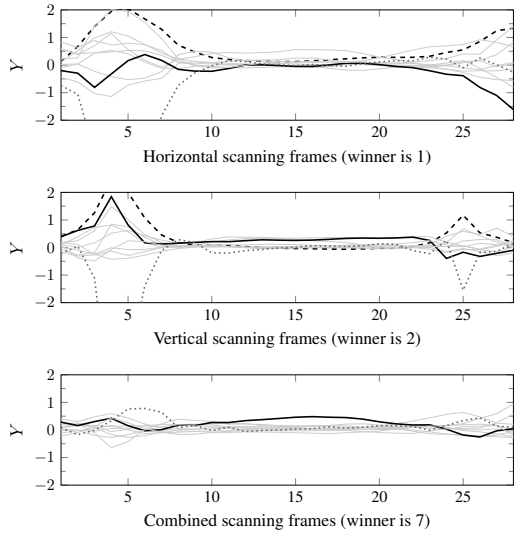


Figure 13: The readouts of the H, V, and H-V-inp recognizers trained on the clean dataset and fed with an image of digit 7 corrupted with border noise. The solid black, dashed black, and dotted gray lines belong to the readouts of digit 7, the winner digit and white space, respectively.

decreased to 64.93%, 34.79%, and 15.42% for the V, H, and H-V-inp systems, respectively. This, in its turn, leads to average DERs of 44.26%, 27.61%, and 22.09%, respectively.

For multi-conditional training, only two results are reported for the DBN system, but the figures in Table 6 do not contradict the conclusion that RCN-based systems are even more robust to noise than DBN systems which are generally acknowledged for their good robustness in comparison to other techniques.

Multi-Column RCN

For completeness, we also trained a two-stage multi-column RCN (MC-RCN) recognizer. The first stage encompasses twelve 2-layer RCNs with a 3K reservoir per layer, one RCN per noise type (6 noise types) and per scanning direction (2 directions). The outputs of these twelve RCNs drive a 2-layer RCN with a 4K reservoir per layer. The reservoir sizes were chosen such that the system has the same number of trainable parameters as the H-V-res system. Apparently, the MC-RCN does not even outperform the much simpler H and V systems (see Table 5). Our hypothesis is that the reservoirs in the first stage are too small to permit an accurate classification. This was confirmed by an experiment in which the reservoir size was increased to 8K (leading to 2M train-

able parameters) and in which the error rate dropped to 3.41%. Increasing the size of the H-V-res system on the other hand did not cause any significant improvement (error rate of 3.49%). This latter results proves that the MC-RCN, in spite of its much larger complexity, will in the end not significantly outperform the much simpler H-V-res architecture.

6.2. *Recognizing connected digits*

As described in Section 3, 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. Figure 14 depicts the output of a multi-conditionally trained RCN with horizontal scanning (the H system in Table 5) which has been supplied with a concatenation of multiple noisy digits.

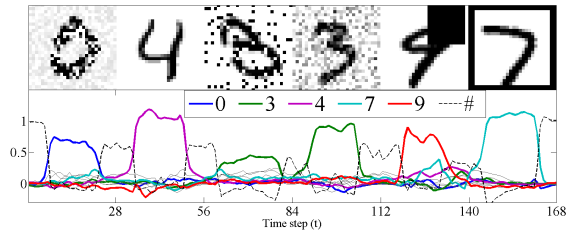


Figure 14: The readouts of a multi-conditionally trained RCN with horizontal scanning supplied with a concatenation of multiple noisy digits.

6.3. *Removing the noise in the front-end*

Multi-condition training is an approach to reduce the mismatch between training and testing. One could also reduce the noise in the front-end. 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) **Mixed-DAE**: a single noise-independent DAE that is trained to remove any kind of noise that appeared in the training data and (2) **Combined-DAE**: a committee of five noise-specific DAEs (one for each noise type as shown in Figure 15), 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.

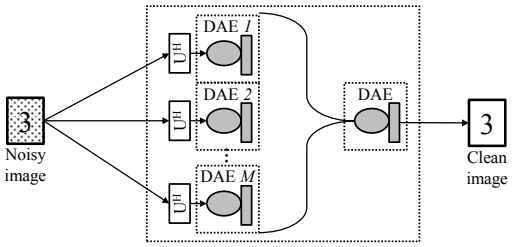


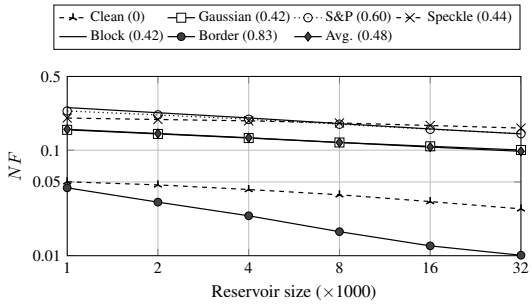
Figure 15: Architecture of the combined DAE: the outputs of a committee of noise-specific DAEs (M different noise types) drive a noise-independent DAE.

Figure 16(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 Mixed-DAE using horizontal scanning.

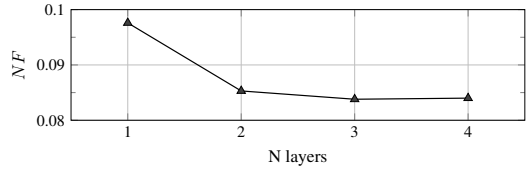
- With a reservoir of size 4K, the mean NF is already smaller than 0.2 for all noise types. The mean NF is in all cases significantly smaller than the mean NF of the raw noisy images (this mean ranged between 0.4 and 0.83 depending on the noise type).
- The noise reduction improves very gradually as 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 Figure 16(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.

Figure 16: 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 Mixed-DAE with 32K reservoirs in each layer as the reference (1.8M trainable parameters) against which we will compare the Combined-DAE. To make Combined-DAE equally complex as the Mixed-DAE (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 Combined-DAE that has been trained for the same noise type. This so-called ideal DAE is denoted as **Ideal-DAE**.

Figure 17 summarizes the results obtained with these three DAEs and supports the following conclusions: (1) For Gaussian and S&P noise types, the Combined-DAE achieves a noise reduction that is nearly identical to that of the Ideal-DAE, but on three other types, the Mixed-DAE is better than the Combined-DAE. (2) On average, there is little difference between the simple Mixed-DAE and the much more complex Combined-DAE. Figure 18 shows the performance of Mixed-DAE on denoising some examples.

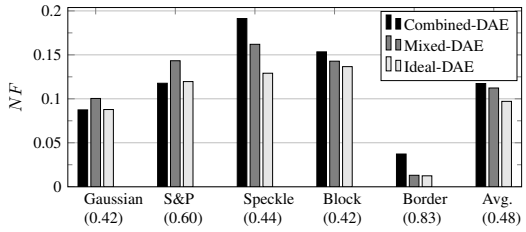


Figure 17: The noise fraction (NF) for the output of the mixed, the combined and an 'ideal' DAE that has prior knowledge of the noise type. The NF of the raw noisy images are mentioned between brackets.

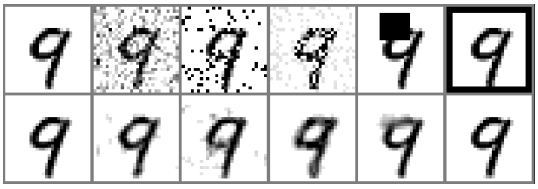


Figure 18: One clean and five noise corrupted samples of digit 9 (top) and the corresponding outputs of the Mixed-DAE.

6.4. Recognition for denoised images

In order to evaluate the influence of the RCN-based DAE on the recognition, we test the cascade of the Mixed-DAE and the H-V-res system we formerly trained on clean images. The results obtained with this cascade are listed in Table 7. The table also includes the performance of the adaptive multi-column stacked sparse denoising auto-encoder (AMC-SSDA) reported in [21] and the RBM-based denoiser reported in [12]. It is clear that the Mixed-DAE 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

Table 7: 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. The systems DBN-2010 and DBN-2013 refer to [12] and [21], respectively.

DAE	RBM-based	AMC-SSDA	-	Mixed-DAE	Mixed-DAE
Classifier	DBN-2010	DBN-2013	H-V-res	H-V-res	H-V-res (RT)
Clean	1.24	1.5	**0.81**	1.03	1.22
Gaussian	-	1.47	32.1	**1.33**	1.57
S & P	-	2.22	38.91	**1.86**	2.17
Speckle	-	**2.09**	49.32	2.41	2.19
Block	19.09	5.18	21.85	4.95	**3.94**
Border	1.29	1.15	79.34	**0.89**	1.25
Average	-	2.27	37.06	2.08	**2.06**

images. Furthermore, the H-V-res system with Mixed-DAE 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 7 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 [12] 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.

7. Conclusions, discussion 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. Key properties of RCNs that were observed on other task such as (noisy) speech recognition and that were validated in this paper on the (noisy) MNIST image dataset are: RNCs generalize well to unseen conditions and they are easy to train, especially so in combination with our training procedure [25] which provides near-optimal values for almost all hyperparamters with very little effort. The temporal processing capabilities of RCNs could also be readily used to coax the system in recognizing digits strings instead of single digits. The results obtained on the MNIST dataset do show that a large enough RCN recognizer can surpass conventional neural network-based recognizers in matched 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.

However, results on MNIST should not be generalized to image processing in general. As can be seen in Table 1, having intermediate representations of various complexity as obtained with deep convolutional nets, only provides small gains on MNIST. On other image recognition task such as CIFAR-10 [33], intermediate representations obtained/learned via 2-dimensional (2D) convolution seem to be crucial in obtaining state-of-the-art results [34]. Hence, we do expect that on tasks such as CIFAR-10, RCNs will need good intermediate representations based on some form of convolution as well in order to obtain competitive results. A straightforward approach would be to plug in features obtained with CNNs. However, this would require training a CNN before training the RCN, loosing one of the key benefits of RCNs, namely the ability for quick experimentation without needing to fine-tune a lot of hyperparameters. Alternative approaches that use bottom-up generated convolutional features obtained via Restricted Boltzmann Machines or k-means clustering [35], or via other unsupervised learning approaches, are in this aspect more appealing. A major, and as of yet unanswered question in this regard is if supervised learning is a pre-requisite for obtaining good intermediate representations. Yet another ap-

proach would be to make 2-dimensional convolution an inherent property of the RCN. In its current form, RCNs perform a form of 1-dimensional convolution by means of their recurrent nodes. Extending this to 2 dimensions would make RCNs more suitable for processing images, and 3 dimensions could enable fast learning for video processing. Combining bi-directional (horizontal and vertical) scanning as done in this paper and in [36] does provide some of the benefits of 2D convolution, but we expect that such setup will prove to be sub-optimal for larger image sizes. Extensions that better mimic the local properties of 2D convolution are however non-trivial and, to our best knowledge, no such extensions have been proposed yet.

Nevertheless, considering the strong points and the performance of RCN, we believe that, even in their current form, RCNs are good candidates to be merged to the conventional DNN-based image and video processing systems. For the more challenging image and video task, some feature engineering will be needed, but we expect that their fast and robust training combined with the easily pre-computed hyperparameters will still makes RCNs good candidates for quick prototyping and even for final 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] K. Fukushima, Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position, Biological Cybernetics 36 (4) (1980) 193–202. URL http://dx.doi.org/10.1007/BF00344251

[2] J. Weng, N. Ahuja, T. S. Huang, Cresceptron: a self-organizing neural network which grows adaptively, in: Proc. IJCNN, 1992, pp. 576–581.

[3] Y. LeCun, L. Bottou, Y. Bengio, P. Haffner, Gradient-based learning applied to document recognition, Proc. IEEE 86 (11) (1998) 2278–2324.

[4] D. E. Rumelhart, G. E. Hinton, R. J. Williams, Learning internal representations by error propagation, Parallel Distributed Processing: Explorations in the Microstructure of Cognition 1 (1986) 318–362.

[5] P. Simard, D. Steinkraus, J. C. Platt, Best practices for convolutional neural networks applied to visual document analysis, in: Proc. DAR, 2003, pp. 958–963.

[6] J. Schmidhuber, Deep learning in neural networks: An overview, Neural Networks 61 (2015) 85 – 117. doi:http://dx.doi.org/10.1016/j.neunet.2014.09.003. URL http://www.sciencedirect.com/science/article/pii/S0893608014002135

[7] G. E. Hinton, N. Srivastava, A. Krizhevsky, I. Sutskever, R. Salakhutdinov, Improving neural networks by preventing co-adaptation of feature detectors, CoRR. URL http://arxiv.org/abs/1207.0580

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

[9] M. Arjovsky, A. Shah, Y. Bengio, Unitary evolution recurrent neural networks, CoRR abs/1511.06464. URL http://arxiv.org/abs/1511.06464

[10] N. Kalchbrenner, I. Danihelka, A. Graves, Grid long short-term memory, CoRR abs/1507.01526. URL http://arxiv.org/abs/1507.01526

[11] N. Srivastava, Improving neural networks with dropout, Master's thesis, U. Toronto (2013).

[12] Y. Tang, C. Eliasmith, Deep networks for robust visual recognition, in: Proc. ICML, 2010, pp. 1055–1062.

[13] R. Salakhutdinov, G. E. Hinton, Deep boltzmann machines, in: AISTATS, 2009, pp. 448–455.

[14] J. Tang, C. Deng, G.-B. Huang, Extreme learning machine for multilayer perceptron, IEEE Trans. Neural Networks and Learning Systems (2015) 1–1doi:10.1109/TNNLS.2015.2424995.

[15] D. Yu, L. Deng, Deep convex net: A scalable architecture for speech pattern classification., in: Proc. Interspeech, 2011, pp. 2285–2288.

[16] D. C. Ciresan, U. Meier, J. Masci, L. M. Gambardella, J. Schmidhuber, Flexible, high performance convolutional neural networks for image classification, in: Proc. IJCAI, 2011, pp. 1237–1242.

[17] M. Ranzato, C. S. Poultney, S. Chopra, Y. LeCun, Efficient learning of sparse representations with an energy-based model., in: Proc. NIPS, MIT Press, 2006, pp. 1137–1144.

[18] D. C. Ciresan, U. Meier, J. Schmidhuber, Multi-column deep neural networks for image classification, in: Proc. CVPR, 2012, pp. 3642–3649.

[19] L. Zhang, D. Zhang, Visual understanding via multi-feature shared learning with global consistency, IEEE Trans. Multimedia 18 (2) (2016) 247–259. doi:10.1109/TMM.2015.2510509.

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

[21] F. Agostinelli, M. Anderson, H. Lee, Adaptive multi-column deep neural networks with application to robust image denoising, Proc. NIPS 26 (2013) 1–9.

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

[23] H. Jaeger, The ''echo state'' approach to analysing and training recurrent neural networks - with an erratum note, Tech. rep., GMD Report 148, German National Research Center for Information Technology (2001). URL http://goo.gl/gJDEZJ

[24] A. Jalalvand, K. Demuynck, 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.

[25] A. Jalalvand, F. Triefenbach, K. Demuynck, J.-P. Martens, Robust continuous digit recognition using reservoir computing, Computer Speech and Language 30 (1) (2015) 135 – 158.

[26] G.-B. Huang, Q.-Y. Zhu, C.-K. Siew, Extreme learning machine: theory and applications, Neurocomputing 70 (2006) 489–501.

[27] G. Huang, S. Song, J. N. D. Gupta, C. Wu, Semi-supervised

and unsupervised extreme learning machines, IEEE Trans. Cybernetics 44 (12) (2014) 2405–2417. doi:10.1109/TCYB.2014.2307349.

[28] L. Zhang, W. Zuo, D. Zhang, LSDT: latent sparse domain transfer learning for visual adaptation, IEEE Trans. Image Processing 25 (3) (2016) 1177–1191. doi:10.1109/TIP.2016.2516952.
URL http://dx.doi.org/10.1109/TIP.2016.2516952

[29] L. Zhang, D. Zhang, Domain adaptation extreme learning machines for drift compensation in e-nose systems, IEEE Trans. Instrumentation and Measurement 64 (2015) 1790–1801.

[30] A. Jalalvand, W. De Neve, J.-P. Martens, R. Van de Walle, Towards using reservoir computing networks for noise-robust image recognition, in: Proc. IJCNN, 2016, pp. 1666–1672.

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

[32] G.-B. Huang, X. Ding, H. Zhou, Optimization method based extreme learning machine for classification, Neural Computation 74 (13) (2010) 155–163.

[33] A. Krizhevsky, Learning multiple layers of features from tiny images, Tech. rep., University of Toronto (2009).

[34] http://rodrigob.github.io/are_we_there_yet/build/, accessed: 2016-09-15.

[35] A. Coates, H. Lee, A. Y. Ng, An analysis of single-layer networks in unsupervised feature learning, in: Proc. Int. Conf. on Artificial Intelligence and Statistics (AISTATS), 2011, pp. 215–223.

[36] F. Visin, K. Kastner, K. Cho, M. Matteucci, A. Courville, Y. Bengio, ReNet: A recurrent neural network based alternative to convolutional networks, arXiv-CS.CV.
URL http://arxiv.org/abs/1505.00393

Azarakhsh Jalalvand obtained the M.Eng. degree in Artificial Intelligence and Robotics from Iran University of Science and Technology (2009), and the PhD degree in Computer Engineering from Ghent University, Ghent, Belgium (2015). The focus of his PhD research was on the noise-corrupted speech and image signal processing using artificial neural networks. He is currently a post-doctoral researcher in the Data Science Lab, Ghent University. His principal research interests are noise-robust signal processing, machine learning and time-series data analysis.

Kris Demuynck received the master's degree in Electrical Engineering and the PhD degree in Applied Sciences from the Katholieke Universiteit Leuven, in 1993 and 2001, respectively. Between 2001 and 2012 he worked as a Postdoctoral Researcher at the Katholieke Universiteit Leuven on various national and international projects. In 2012 he joined the Data Science Lab at Ghent University, where he works as part-time professor and senior researcher. His principal research interests are large vocabulary continuous speech recognition (LVCSR), machine learning and search algorithms.

Wesley De Neve received the M.Sc. degree in Computer Science and the Ph.D. degree in Computer Science Engineering from Ghent University, Ghent, Belgium, in 2002 and 2007, respectively. He is currently working as a senior researcher for both the Data Science Lab at Ghent University - iMinds in Belgium and the Image and Video Systems Lab at the Korea Advanced Institute of Science and Technology (KAIST) in South Korea. His current research interests are natural language understanding, visual content analysis and machine learning.

Jean-Pierre Martens graduated in 1974 as a M.Sc. in Electrical Engineering and obtained his PhD in 1982. Since 1974, he has been working at University Gent, first as a researcher, then as an associate professor, and since 1996 as a full professor. He has carried out research in psycho-acoustics, auditory model-based speech analysis, speech recognition and music signal analysis. He is a member of ASA, IEEE, INNS and ISCA.