Pattern classification with CNNs as reservoirs

D. Verstraeten[†*], S. Xavier-de-Souza[‡], B. Schrauwen[†], J. Suykens[‡], D. Stroobandt[†], J. Vandewalle[‡]

†Dept. of Electronics and Information Systems, Ghent University
Sint Pietersnieuwstraat 41, 9000 Ghent, Belgium
‡Dept. of Electrical Engineering - ESAT-SCD-SISTA
Kasteelpark Arenberg 10, B-3001 Leuven (Heverlee) Belgium
[*] Corresponding author: david.verstraeten@ugent.be

Abstract—Reservoir Computing is a novel method in the field of neural networks and machine learning, which combines the computational power of a nonlinear dynamic system with the ease of training of a linear classifier. The basic setup is as follows: a sufficiently complex network of nonlinear nodes (called the reservoir) is excited by an input signal, and the instantaneous dynamic response of the system is then used to train a simple linear readout function.

In this contribution, we present a proof-of-concept system that demonstrates the possibility of using the nonlinear spatiotemporal dynamics of a Cellular Neural/Nonlinear Network (CNN) to play the role of a reservoir. We discuss the advantages and limitations of this approach and illustrate the idea by using the system to solve both a simple academic task and a real world speech recognition problem. We use a global optimization method called Coupled Simulated Annealing (CSA) to optimize CNN templates that give suitable reservoir properties to the CNN. Finally, we validate the simulation results using an ACE16k CNN chip.

1. Introduction

In the field of machine learning, a large class of interesting real-world problems shares the same two characteristics: a strong nonlinear input-output mapping, and the fact that a large amount of the information is contained in the temporal sequence of the input data. Examples are, amongst others, problems from systems control, robotics, timeseries forecasting, language processing, etc. In this field, recurrent neural networks are often seen as a method with a large potential for capturing the nonlinear and dynamic properties of the problem, but are also notoriously difficult to train.

A possible solution to this training problem for RNNs has been proposed under the name *Echo State Networks [2],* but has since been extended to a broader class of learning systems under the name *Reservoir Computing (RC)[4].*

Reservoir Computing uses a recurrently connected network of discrete nonlinear nodes (originally sigmoid neurons) - the *reservoir* - which is constructed randomly at the beginning of the experiment (i.e. the weights are drawn from a random distribution), globally scaled for stability and left untrained afterwards. The input timeseries of the training set are used to excite the reservoir and the dynamic response of the reservoir is then used to train a linear output layer based on the desired output vectors at every timestep. Because the only layer that is actually trained is linear, this can be done using any of the available methods for estimating linear models, e.g. linear least-squares techniques.

Cellular Neural/Nonlinear Networks (CNNs) [1] are a well-established computational paradigm with a sound mathematical foundation and a whole range of applications, mainly in image and video processing. The behaviour of these CNNs is governed by the weight templates, which are either trained using a learning rule but are also often tuned manually. In this paper, we present a novel use of the CNN model that does not require hand-tuning of the template and which broadens the area of application for CNNs substantially.

This paper is constructed as follows: in section 2 we formalize the experimental setting. In section 3 we will present the results of the experiments and discuss some of the implications. In section 4 finally, we draw conclusions from our work and suggest some possibilities for future work.

2. Experimental setup

We use the following notations: k is a discrete index indicating the timestep, $\mathbf{u}[k]$ is the input vector at time k to the system, $\mathbf{x}[k]$ is a vector of internal states of the reservoir, $\mathbf{y}[k]$ and $\hat{\mathbf{y}}[k]$ are vectors representing the desired and estimated output respectively, and $\mathbf{W}_{\mathbf{B}}^{\mathbf{A}}$ is a weight matrix containing the connections from $\mathbf{A}$ to $\mathbf{B}$. The reservoir is a recurrently connected, discrete time neural network, and is governed by the following state equation:

$$\mathbf{x}[k+1] = f(\mathbf{W}_{\mathbf{res}}^{\mathbf{res}}\mathbf{x}[k] + \mathbf{W}_{\mathbf{res}}^{\mathbf{in}}\mathbf{u}[k]) \qquad (1)$$

with $f(\cdot)$ a nonlinear function - usually a tanh(). In this paper, however, we use a piecewise linear function as saturation characteristic to approximate the CNN model more accurately.

The output of the system is then computed as follows:

$$\hat{\mathbf{y}}[k+1] = \mathbf{W}_{\mathbf{out}}^{\mathbf{res}}\mathbf{x}[k+1] + \mathbf{W}_{\mathbf{out}}^{\mathbf{in}}\mathbf{u}[k] \qquad (2)$$

As mentioned before, in traditional RC systems the $\mathbf{W}_{\mathbf{res}}^{\mathbf{in}}$ and $\mathbf{W}_{\mathbf{res}}^{\mathbf{res}}$ matrices are constructed randomly at the beginning of any experiment and left untrained. The $\mathbf{W}_{\mathbf{out}}^{\mathbf{in}}$ and $\mathbf{W}_{\mathbf{out}}^{\mathbf{res}}$ matrices are trained by minimizing the mean square error between the desired and estimated outputs, with an additional constraint on the norm of the weight vector to perform regularization:

$$\mathbf{W_{out}} = \arg\min_{W} \left(||\hat{\mathbf{y}} - \mathbf{y}||^2 + \lambda\, ||W||^2 \right) \tag{3}$$

where $\hat{\mathbf{y}} = \mathbf{Wx}$ and $\mathbf{y}$ are concatenated across all timesteps, and λ is a regularization parameter that controls the amount in which large weights are penalized to avoid overfitting. Determining the optimal weights for the readout layer is a well known problem and can be solved using ridge regression, also known as Tikhonov regularization [5], where the optimal regularization parameter λ is determined using in this case threefold cross-validation on a distinct validation set.

2.1. CNNs as reservoirs

There exist many CNN models, but here we use a space-invariant model where every cell is connected to its eight neighbours using the same weight template. The cells are further characterized by a piecewise linear output function, and the network operates in a discrete time simulation mode. With these assumptions, the differences between the CNN and the traditional reservoir setup is twofold:

- Instead of a randomly connected network, the cells are connected in a regular 2D lattice, with a space-invariant weight template.
- The output nonlinearity is a piecewise linear function instead of the traditional smooth tanh function.

These two restrictions can easily be incorporated into a simulation model, and we can thus simulate a CNN as reservoir using only adjustments to the network topology and nonlinearity. The input signal is connected to the CNN cells using a random input connection matrix - as with traditional reservoirs - where here the connection weights are randomly set to .1 or -.1.

We also validated the simulation results on actual hardware. For this, we used an ACE16k chip [3] with 128x128 cells and a weight precision of 8 bits. However, for computational reasons we only use a center 8x8 grid of nodes leaving the other cells inactive. When making the transition from the software simulation to actual hardware, additional differences need to be noted:

- The limited precision of the template weights : the weights are represented internally as 8 bit digital values.
- The chip is built on analog VLSI technology, which introduces additional noise and other effects such as thermal variation.

2.2. Studied tasks

We have evaluated the performance of the system on two machine learning tasks: a simple academic task and a more demanding real-world problem.

Signal classification The objective is to classify a onedimensional input signal. The signal can be either a sawtooth or square signal with the same period so the reservoir cannot use the period to discern between the instances of the signal, and the desired output is a onedimensional value indicating the correct signal type at every timestep coded as +1 and -1. The transitions between both signal types occurs at random moments, so the main difficulty of this task occurs on the transition moments between the signals, where the reservoir has to detect a change in the signal characteristics very quickly. The error is computed using a zero-one loss function at every timestep.

Isolated digits recognition This real world problem consists of the classification of isolated digits zero to nine spoken by two different speakers, where each digit was uttered ten times resulting in a total dataset of 200 speech samples. The speech was preprocessed using a biological model of the human ear resulting in a 86-dimensional input vector for every $16ms$ of speech. The desired output is a 10-dimensional vector, one for every digit. The performance of the system is expressed as word error rate (WER), which is the fraction of misclassified digits.

3. Experimental results and discussion

3.1. Sweeping the parameter space

In order to reduce the parameter space we opted to use a symmetric template with only three distinct weight values: diagonal, lateral (horizontal and vertical) and self-recurrent. Thus, we were able to do a full sweep of the interesting part of the threedimensional parameter space.

Fig. 1 on the left hand side shows simulation results for the signal classification task as the diagonal, lateral and self-recurrent weights are varied. To the left, the error is shown, and to the right the variance of the error due to the variation of the remaining weight. The two top figures show that positive diagonal weights are beneficial, but that only the magnitude and not the sign of the lateral weights matters. Additionally, the middle row shows that positive diagonal weights require small negative self-recurrent weights and vice versa.

The same figure on the right hand side shows the same plots for the speech recognition task. The aspect of the figures is less noisy, indicating a smoother error surface. Also, in all three plots a symmetry is apparent which means that only the absolute value instead of the sign of the weights is important. The top plot shows that the value of the lateral weights have a bigger impact on performance than the diagonal weights. Interestingly, the middle plot shows that the values of the self-recurrent and diagonal weights are related performance-wise, as indicated by the diagonal ridge

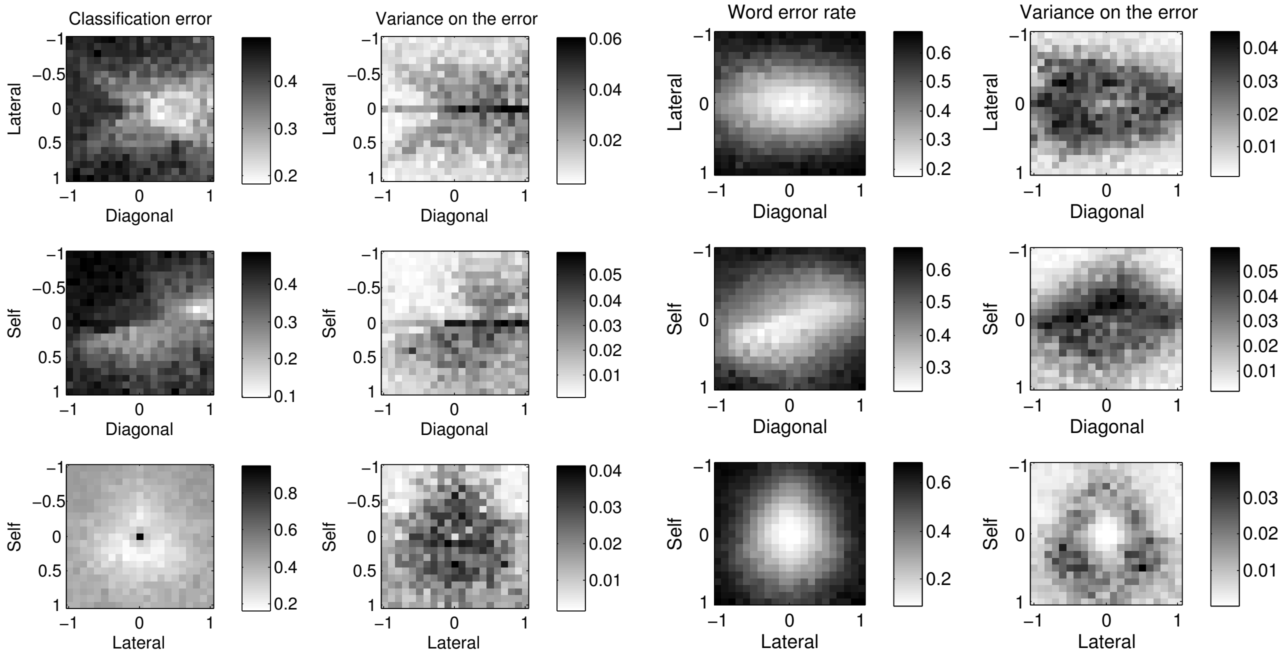

Figure 1: Simulation results for the signal classification task (left) and speech recognition task (right) as a function of the template parameters. The template is symmetric - with the lateral (horizontal and vertical), diagonal and self-recurrent weights being three separate parameters. In the left column the mean error on the testset is shown, in the right column the variance on the error due to changes in the remaining weight.

of optimal performance. Finally, the variance of the performance is almost everywhere inversely related to the error, except in the middle area of the bottom plot which is thus the optimal region with regards to performance and robustness.

3.2. Template optimization with Coupled Simulated Annealing

Next, we used a global optimization technique called Coupled Simulated Annealing (CSA) [6] for the optimization of the CNN template, both in simulation and on the actual chip.

Traditionally, reservoirs are randomly constructed recurrent networks of nonlinear nodes, where the interconnection weights are drawn from a certain distribution - usually Gaussian. In the case of CNNs however, the interconnection structure is quite specific and the parameter space is far less-dimensional than for general reservoirs. This allows the use of search-based optimization techniques. Here, we use an extension of the well known technique of Simulated Annealing (SA), which uses several coupled SA processes running in parallel. CSA couples these parallel classifiers by their acceptance probabilities, which results in a better overall performance and less sensitivity to the initial conditions. For the experiments here, the initial temperature was set to 1, the number of parallel probes to 5 and the maximal number of iterations to 1000. Figure 2 shows a schematic view of the different components involved in the experiments. In case the experiments are done in software, the whole simulation, evaluation and optimization process is done using Matlab. For experiments on chip, the input timeseries are transformed into avi files with a single frame per timestep, and the resulting dynamic response from the chip is also saved as an avi file, which is then transformed back to the internal representation necessary for training and simulation. In both cases, the result of the evaluation step - i.e. the error on the test set - is used by the Coupled Simulated Annealing module to adjust the template.

Since CSA is a probabilistic algorithm, the parameter space is sampled in a non-uniform manner. Additionally, our experiments show that for random templates, the error surface is quite complex. This makes it difficult to visualize the results for the optimization process. We therefore only mention the optimal performances here.

On the speech recognition task, running the CSA optimization on our simulation model yielded a minimal error of 3.6%, and the same algorithm on the chip achieved a minimal error of 6%. We did notice a larger occurrence of badly performing templates on the chip (i.e. with large error rates), which is probably caused by the fact that those templates are less robust to the noise on chip, which renders the task more difficult. As a means of comparison: the average performance of a standard reservoir of the same size is 2%, and the direct application of a linear readout layer to the feature vectors - i.e. without a reservoir in between - gives a performance of 11%.

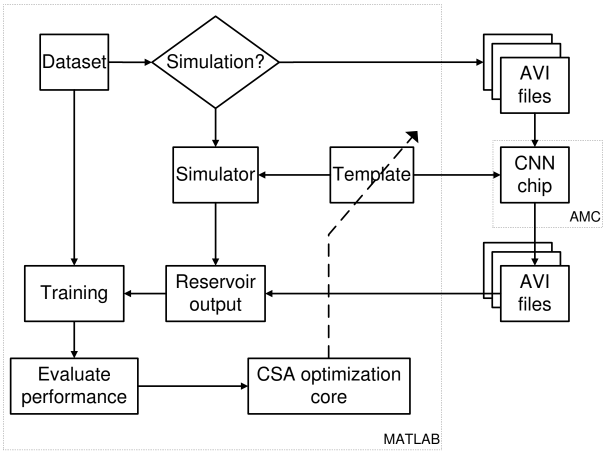

Figure 2: Overview of the experimental setup.

On the signal classification task, the optimal template found by the CSA optimization core attained a performance of 1% in simulation, and an even better performance of 0.1% on chip. Here, standard reservoirs give a similar performance to the CNN simulation model of 1%, and the direct application of the linear readout to the input is not able to solve the task at all since it requires at least some short-term memory. The fact that the speech task performs worse on chip as opposed to the signal classification task, is possibly due to the fact that the on-chip noise has a negative effect on the speech recognition task whereas for the simple signal classification task it actually helps discerning between the signals, but further work is needed to make this conclusive.

4. Conclusions and future work

We have presented a novel, proof-of-concept use of Cellular Neural Networks in the general framework of Reservoir Computing. This framework extends the possible application areas for CNNs substantially. Reservoir Computing offers a simple method for using dynamic nonlinear media such as CNNs for machine learning tasks and as such it enables the use of the impressive computational power and speed of current analog VLSI implementations for a broad class of machine learning tasks such as speech recognition. We have shown a proof of concept system and applied it to both an academic and a real world task, demonstrating consistent and competitive results compared to our simulation model.

A possible future research direction is e.g. the implementation of the linear readout layer on chip. Currently the computation of the readout layer is done offline in Matlab, which requires a lot of communication between the chip and the host computer. However, by using a multiplicative mask on the cell outputs and then running the chip in diffusion mode - which lets all cells converge to the average activation level of the network - this can be done fast and on-chip. Finally, it is useful to look into the possibility of using the large body of theoretical work on the dynamic behaviour of CNNs for guiding the search for an optimal template, since the dynamic regime in which the reservoir operates is quite important for its performance.

Acknowledgments

DV acknowledges research supported by the Institute for the Promotion of Innovation through Science and Technology in Flanders (IWT Vlaanderen). BS acknowledges research supported by FWO Flanders project G.0317.05 and the Photonics@be Interuniversity Attraction Poles program (IAP 6/10), initiated by the Belgian State, Prime Minister's Services, Science Policy Office. SXS, JS and JVDW acknowledge research supported by Research Council K.U. Leuven: GOA AMBioRICS, CoE EF/05/006, OT/03/12, PhD/postdoc & fellow grants; Flemish Government: FWO PhD/postdoc grants, FWO projects G.0499.04, G.0211.05, G.0226.06, G.0302.07; Research communities (ICCoS, ANMMM, MLDM); AWI: BIL/05/43, IWT: PhD Grants; Belgian Federal Science Policy Office: IUAP DYSCO.

References

[1] L. O. Chua and L. Yang. Cellular neural networks: Theory. *IEEE Transactions on Circuits and Systems*, 35(10):1273–1290, 1988.

[2] H. Jaeger and H. Haas. Harnessing nonlinearity: predicting chaotic systems and saving energy in wireless telecommunication. *Science*, 308:78–80, April 2 2004.

[3] A. Rodriguez-Vazquez, G. Linan-Cembrano, L. Carranza, E. Roca-Moreno, R. Carmona-Galan, F. Jimenez-Garrido, R. Dominguez-Castro, and S.E. Meana. ACE16k: the third generation of mixed-signal SIMD-CNN ACE chips toward VSoCs. *IEEE Transactions on Circuits and Systems I*, 51(5):851–863, 2004.

[4] B. Schrauwen, D. Verstraeten, and J. Van Campenhout. An overview of reservoir computing: theory, applications and implementations. In *Proceedings of the European Symposium on Artifical Neural Networks (ESANN)*, 2007.

[5] A.N. Tikhonov and V. Y. Arsenin. *Solutions of Ill-Posed Problems*. Winston and Sons, 1977.

[6] S. Xavier de Souza, J. Suykens, J. Vandewalle, and D. Bolle. Cooperative behavior in coupled simulated annealing processes with variance control. In *Proc. of the International Symposium on Nonlinear Theory and its Applications (NOLTA)*, 2006.