ESANN 2012 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium), 25-27 April 2012, i6doc.com publ., ISBN 978-2-87419-049-0.
Available from http://www.i6doc.com/en/livre/?GCOI=28001100967420.

Constructive Reservoir Computation with Output Feedbacks for Structured Domains

Claudio Gallicchio, Alessio Micheli and Giulio Visco *

Department of Computer Science - University of Pisa
Largo B. Pontecorvo, 3 - 56127 Pisa, Italy.

Abstract. We introduce a novel constructive algorithm which progressively builds the architecture of GraphESN, which generalizes Reservoir Computing to learning in graph domains. Exploiting output feedback signals in a forward fashion in such construction, allows us to introduce supervision in the reservoir encoding process. The potentiality of the proposed approach is experimentally assessed on real-world tasks from Toxicology.

1 Introduction

Reservoir Computing (RC) [1] is an extremely efficient input driven paradigm for modeling recursive systems. RC comprises different models, among which the Echo State Network (ESN) [2] is undoubtedly the most popular and investigated one. An ESN is composed of a large, sparsely connected, non-linear recurrent hidden untrained *reservoir* and of a linear feed-forward trained *readout*. Recently, RC has been generalized from sequence domains to structured domains (SDs), i.e. variable size structures which include either sequences, trees and graphs. TreeESN [3] and GraphESN [4] implement recursive systems, for trees and graphs, respectively, exploiting the extremely efficient RC paradigm.

The automatic determination of the RC architecture and the limitations due to the fixed reservoir dynamics (independent on the target) remain as interesting open issues (e.g. [1, 5, 6]). In this paper, we cope with both such issues by building on the GraphESN model a new constructive approach which allows us to introduce stable output feedbacks on SD processing. In particular, inspired by Cascade Correlation [7, 8], we introduce an algorithm to incrementally construct the architecture of a GraphESN by progressively adding new sub-networks which are trained to simulate the residual error of the model. The proposed constructive approach is the ground to introduce a novel general method to manage output feedbacks in RC models, hence suitable for learning in SDs. This results in a reservoir encoding process gradually modified during the network construction on the basis of the target information.

2 Constructive Output Feedbacks in GraphESN

A graph $\mathbf{g}$ is a couple $(V(\mathbf{g}), E(\mathbf{g}))$, where $V(\mathbf{g})$ and $E(\mathbf{g})$ are the set of vertices and the set of edges of $\mathbf{g}$, respectively, with $E(\mathbf{g}) = \{(u, v)|u, v \in V(\mathbf{g})\}$. The number of vertices in $\mathbf{g}$ is denoted by $|V(\mathbf{g})|$. The neighborhood of v is the set of

*This work is partially supported by the EU FP7 RUBICON project (contract n. 269914).

ESANN 2012 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium), 25-27 April 2012, i6doc.com publ., ISBN 978-2-87419-049-0.
Available from http://www.i6doc.com/en/livre/?GCOI=28001100967420.

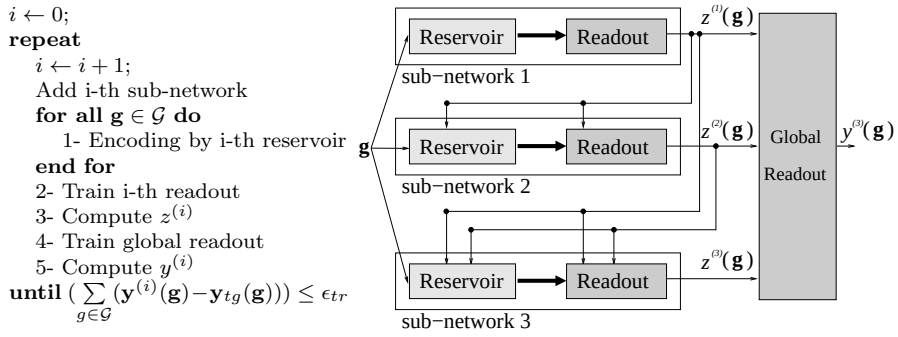

Fig. 1: *Left*: Training algorithm. *Right*: Incremental top level architecture (for $i = 3$). Symbols $z^{(i)}$ denote the outputs of sub-networks, $y^{(i)}$ is the global output of the model (see text), and ϵ_{tr} is a threshold on the global training error.

adjacent vertices to v, i.e. $\mathcal{N}(v) = \{u \in V(\mathbf{g}) | (u, v) \in E(\mathbf{g})\}$ for an undirected graph, where $|\mathcal{N}(v)|$ is the degree of v. The maximum degree over the set of considered graphs is denoted as k. The label associated to vertex v is denoted by $\mathbf{u}(v) \in \mathbb{R}^{N_U}$, where $\mathbb{R}^{N_U}$ is the vectorial label input space.

GraphESNs [4] generalize RC to SD (graph) processing. A generalized reservoir architecture is used to encode each (variable size and topology) input graph into a structured state representation isomorphic (i.e. with the same skeleton) to the input. In analogy to ESNs, the reservoir of GraphESNs is left untrained after *contractive* initialization, and training is required only for the parameters of the readout. Considering graph-to-element tasks [4], i.e. tasks in which (variable size and topology) input graphs are required to be mapped into vectorial outputs, a training set is represented as $\mathfrak{T} = \{(\mathbf{g}, \mathbf{y}_{tg}(\mathbf{g})) |\ \mathbf{g} \in \mathcal{G},\ \mathbf{y}_{tg}(\mathbf{g}) \in \mathbb{R}^{N_Y}\}$, where $\mathcal{G}$ denotes a set of graphs, $\mathbf{y}_{tg}(\mathbf{g})$ is the target output associated to $\mathbf{g}$, and $\mathbb{R}^{N_Y}$ is the vectorial output space. For the sake of simplicity, in the following we assume $N_Y = 1$, e.g. $y_{tg}(\mathbf{g}) \in \{-1, +1\}$ for classification tasks on graphs.

The constructive approach presented in this paper builds a cascade of GraphESNs, progressively added and each trained to approximate the error obtained by the already frozen sub-networks. Each GraphESN contains an input layer with N_U units, a generalized recurrent reservoir with N_R units and a feed-forward readout with $N_Y = 1$ unit. Fig. 1 shows the training algorithm for incremental network construction and the resulting multi-layer network architecture, including the functional dependencies among the network components.

Observe that, in the proposed incremental approach, the outputs of already frozen sub-networks, trained to previous global error of the model and denoted as $z^{(i)}$, are used to feed each newly added sub-network both in the readout part, as in standard cascade models, and in the reservoir part, introducing output feedback in the encoding process. The proposed algorithm, based on *forward output feedbacks*, is called GraphESN-FOF. The process of incremental construction of the GraphESN-FOF architecture continues with the addition of subsequent sub-GraphESNs until a stop condition is met (e.g. on the global training error,

ESANN 2012 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium), 25-27 April 2012, i6doc.com publ., ISBN 978-2-87419-049-0.
Available from http://www.i6doc.com/en/livre/?GCOI=28001100967420.

see algorithm in Fig. 1). Note that the constructive scheme described here for general SD can be also applied to standard RC for sequence processing tasks, as whenever the input reduces to a set of sequences, GraphESN dynamics reduces to standard RC one.

Let us start considering the i-th GraphESN added in the incremental architecture. The reservoir is used to encode (pass 1 of the algorithm in Fig. 1) each input graph $\mathbf{g}$ into a structured state $\mathbf{x}^{(i)}(\mathbf{g})$, which has the same topology of $\mathbf{g}$. This is realized by applying the same reservoir architecture to each vertex $v \in V(\mathbf{g})$, in order to compute a state label $\mathbf{x}^{(i)}(v) \in \mathbb{R}^{N_R}$ according to the equation:

$$\mathbf{x}^{(i)}(v) = tanh(\mathbf{W}_{in}\mathbf{u}(v) + \hat{\mathbf{W}} \sum_{v' \in \mathcal{N}(v)} \mathbf{x}^{(i)}(v') + \mathbf{w}_{fof} \sum_{j=1}^{i-1} z^{(j)}(\mathbf{g})) \qquad (1)$$

where $\mathbf{u}(v) \in \mathbb{R}^{N_U}$ is the label attached to v, $\mathbf{W}_{in} \in \mathbb{R}^{N_R \times N_U}$ is the input-to-reservoir weight matrix, $z^{(j)}(\mathbf{g}) \in \mathbb{R}$ for $j = 1, \ldots, i-1$ are the outputs of the previous sub-GraphESNs in the cascade (see Fig. 1) and $\mathbf{w}_{fof} \in \mathbb{R}^{N_R}$ is the forward output feedback weight vector, which is a new component introduced here with respect to the correspondent equation in GraphESN [4]. The reservoir is initialized to implement *contractive* dynamics and is left untrained. Assuming *tanh* as reservoir activation function and Euclidean distance as metric in the reservoir space, a sufficient condition for contractivity of reservoir dynamics is given by $\sigma = k \, \|\hat{\mathbf{W}}\|_2^2 < 1$, where k is the maximum degree over $\mathcal{G}$. Weights in $\mathbf{W}_{in}$ are drawn randomly from a uniform distribution over $[-scale_{in}, scale_{in}]$. Elements in $\mathbf{w}_{fof}$ are all equal to w_{fof}. The condition of contractivity, inherited from ESNs and TreeESNs, has a relevant role for GraphESNs. According to the Banach Theorem, contractivity of eq. 1 ensures stability of the encoding also in case of cyclic and undirected input structures, which imply mutual dependencies among reservoir state variables. Moreover, contractivity bounds the reservoir dynamics within a region characterized by Markovian properties [9, 4]. The encoding process is therefore practically implemented by resorting to an iterated version of eq. 1, which is applied until convergence to a stable solution (i.e. the fixed point of eq. 1). After the convergence of the encoding process, a *mean state mapping* function is applied to get a fixed-size state representation for the whole graph, i.e. $\chi(\mathbf{x}^{(i)}(\mathbf{g})) = (1/|V(\mathbf{g})|) \sum_{v \in V(\mathbf{g})} \mathbf{x}^{(i)}(v)$.
The output of the i-th GraphESN is computed by the i-th readout (see Fig. 1):

$$z^{(i)}(\mathbf{g}) = f_{out}^{(i)}(\mathbf{w}_{out}^{(i)}[\chi(\mathbf{x}^{(i)}(\mathbf{g}))^T, z^{(1)}(\mathbf{g}), \ldots, z^{(i-1)}(\mathbf{g})]^T) \qquad (2)$$

where $\mathbf{w}_{out}^{(i)} \in \mathbb{R}^{1 \times (N_R + i - 1)}$ is the i-th readout weight matrix and $f_{out}^{(i)}$ is the activation function for the units in the i-th readout (in this paper we use $f_{out}^{(i)} \equiv tanh$ for every i). The global output of the network, when i sub-GraphESNs have been added to the architecture, is finally computed by a global readout

$$y^{(i)}(\mathbf{g}) = f_{out}(\mathbf{w}_{out}[z^{(1)}(\mathbf{g}), z^{(2)}(\mathbf{g}), \ldots, z^{(i)}(\mathbf{g})]^T) \qquad (3)$$

ESANN 2012 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium), 25-27 April 2012, i6doc.com publ., ISBN 978-2-87419-049-0.
Available from http://www.i6doc.com/en/livre/?GCOI=28001100967420.

where $y^{(i)}(\mathbf{g}) \in \mathbb{R}$ is the global output of the network after the addition of i sub-GraphESNs, $\mathbf{w}_{out} \in \mathbb{R}^{1\times i}$ is the readout weight matrix and f_{out} is the readout activation function (in this paper we use $f_{out} \equiv id$). As in standard RC, only readout parameters (i.e. $\mathbf{w}_{out}^{(i)}$ and $\mathbf{w}_{out}$) are trained. In particular, the i-th GraphESN is trained to simulate the residual error of the network at the previous pass of the constructive algorithm, i.e. for each $\mathbf{g} \in \mathcal{G}$ the target for the i-th readout is $z_{tg}^{(i)}(\mathbf{g}) = y^{(i-1)}(\mathbf{g}) - y_{tg}(\mathbf{g})$, where we assume $y^{(0)}(\mathbf{g}) = 0$. Hence, $\mathbf{w}_{out}^{(i)}$ is selected (passes $2-3$ of the algorithm in Fig. 1) to solve the least squares regression problem $\|\mathbf{w}_{out}^{(i)}\mathbf{X}^{(i)} - (f_{out}^{(i)})^{-1}(\mathbf{z}_{tg}^{(i)})\|_2^2$, where the columns of $\mathbf{X}^{(i)}$ contain the vectors $[\chi(\mathbf{x}^{(i)}(\mathbf{g}))^T, z^{(1)}(\mathbf{g}), \ldots, z^{(i-1)}(\mathbf{g})]^T, \forall \mathbf{g} \in \mathcal{G}$, and $\mathbf{z}_{tg}^{(i)}$ is a row vector containing the elements $z_{tg}^{(i)}(\mathbf{g}) \;\; \forall \mathbf{g} \in \mathcal{G}$, respectively representing input and target information for the readout of the i-th GraphESN. After the training process for the i-th GraphESN, the readout weights in $\mathbf{w}_{out}^{(i)}$ are frozen and the global readout is (re-)trained (passes $4-5$ of the algorithm in Fig. 1) to solve $\|\mathbf{w}_{out}\mathbf{Z}^{(i)} - (f_{out})^{-1}(\mathbf{y}_{tg})\|_2^2$, where, for every $\mathbf{g} \in \mathcal{G}$, the columns of $\mathbf{Z}^{(i)}$ contain the vectors $[z^{(1)}(\mathbf{g}), z^{(2)}(\mathbf{g}), \ldots, z^{(i)}(\mathbf{g})]^T$, and the row vector $\mathbf{y}_{tg}$ contains the elements $y_{tg}(\mathbf{g})$. The readouts in sub-networks and the global readout can be trained as in standard RC, using direct methods, e.g. ridge regression, or resorting to iterative LMS learning algorithm. Finally note that the reservoir of each sub-GraphESN added in the architecture is fed by the estimations of the global residual errors computed by the readouts of the previous sub-GraphESNs (eq. 1). This introduces a form of supervision in the reservoir encoding process, resulting in the progressive construction of reservoir state representations influenced by the target.

3 Experiments

The potentiality of the GraphESN-FOF approach has been experimentally assessed on the four tasks from the Predictive Toxicology Challenge (PTC) dataset [10], reporting the carcinogenicity of 417 chemicals in correspondence of four classes of rodents: female rats (FR), female mice (FM), male rats (MR), male mice (MM). Each molecule is represented as an undirected graph, where vertices stand for atoms and edges stand for bonds. Each vertex label contains a 1-of-m encoding of the atom element and the partial charge. The label dimension is 24 and the maximum degree is $k = 4$. Four classification tasks are defined, one for each class of rodents [11], where a target $+1$ is associated to active molecules, and a target -1 is associated to inactive ones.

For model selection, we considered GraphESN-FOF with $scale_{in} \in \{1, 0.1\}$, $w_{fof} \in \{1, 2\}$ and $\sigma = 1$. Sub-GraphESNs readouts were trained by ridge regression with regularization parameter $\lambda_r \in \{0.1, 0.2, 0.01\}$. Global readouts were trained using LMS with weight decay parameter $\lambda_{wd} \in \{0, 0.01\}$. Reservoir dimensions $N_R = 30, 50$ were considered. For comparison, analogous experiments on GraphESNs were conducted, using the same parametrizations for $scale_{in}$, λ_r and σ, with $N_R \in \{50, 100, 200, 500\}$. It is worth to note that in our ex-

ESANN 2012 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium), 25-27 April 2012, i6doc.com publ., ISBN 978-2-87419-049-0.
Available from http://www.i6doc.com/en/livre/?GCOI=28001100967420.

periments, the number of trainable parameters for GraphESN-FOF ($\mathbf{w}_{out}$ and $\mathbf{w}_{out}^{(i)}$) and GraphESN are comparable (roughly proportional to the total number of reservoir units). We considered full connected reservoirs. The constructive algorithm in GraphESN-FOF was stopped whenever the fitting on the training set was analogous to the one achieved by GraphESN and the absolute difference between the errors in two consecutive steps was below 1%. Accordingly, the maximum number of sub-networks added varied between 6 and 20 on the four tasks. Performance accuracy was evaluated by a 5-fold stratified cross validation, with 5 independent reservoir guesses for every reservoir parametrization. Readout regularization and reservoir parametrization were chosen on a validation set by an extra level of stratified 5-fold cross validation.

Table 1 reports the mean test accuracies on the PTC tasks for GraphESN and GraphESN-FOF. The results show a small but consistent advantage (in 3 over the 4 cases) of the constructive adaptive approach of reservoir computation in GraphESN-FOF with respect to standard GraphESNs. The performances of GraphESN-FOF are also comparable to those achieved on the same tasks by state-of-the-art kernels for SDs, e.g. Marginalized, Optimal Assignment (OA) and OA with reduced graph representation, reported in [11].

Model	*FR*	*FM*	*MR*	*MM*
GraphESN	67.7(±0.1)	60.7(±0.4)	56.7(±0.9)	67.1(±0.1)
GraphESN-FOF ($N_R = 30$)	67.9(±1.5)	62.8(±1.6)	57.4(±1.7)	65.4(±1.6)
GraphESN-FOF ($N_R = 50$)	68.3(±1.1)	62.5(±1.2)	57.2(±1.3)	66.6(±1.7)

Table 1: Mean test accuracies (%) on PTC for GraphESN and GraphESN-FOF.

The effectiveness of the introduction of output feedbacks in producing a reservoir state space organization progressively more suitable for the task at hand is investigated using Principal Component Analysis (PCA). Fig. 2 shows the first two PCs of the reservoir space of a GraphESN-FOF, in correspondence of subsequent passes (i.e. sub-networks) in the iterative constructive algorithm for the FR task. Points in the plots are the projections in PCs space of the encoded training input graphs. White and black points respectively denote patterns with positive and negative targets. Note that the separation among white and black points progressively increases with the addition of sub-networks.

4 Conclusions

We introduced a new adaptive input driven model for learning in SDs, based on the generalization of RC to graphs and named GraphESN-FOF. The contributions of this paper are twofold: a novel constructive approach for multi-layer RC and a scheme for stable output feedbacks in RC for SDs. The methodology has the advantage of automatically construct the RC architecture during training. Moreover, the output feedback signals enable to introduce supervision in the RC encoding process, with the aim of tailoring reservoir dynamics to the task

ESANN 2012 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium), 25-27 April 2012, i6doc.com publ., ISBN 978-2-87419-049-0.
Available from http://www.i6doc.com/en/livre/?GCOI=28001100967420.

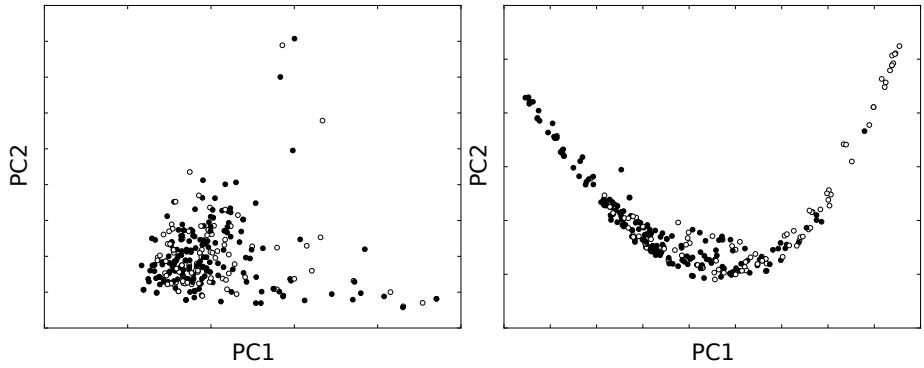

Fig. 2: PCA of sub-networks reservoirs for passes 1 (*left*) and 9 (*right*) of the GraphESN-FOF construction with $N_R = 50$ for the FR task.

at hand. The effectiveness of such contribution has been shown through the analysis of the empirical results and of the PCA plots of reservoir spaces.

References

[1] M. Lukoševičius and H. Jaeger. Reservoir computing approaches to recurrent neural network training. *Computer Science Review*, 3(3):127 – 149, 2009.

[2] H. Jaeger and H. Haas. Harnessing nonlinearity: Predicting chaotic systems and saving energy in wireless communication. *Science*, 304(5667):78–80, 2004.

[3] C. Gallicchio and A. Micheli. TreeESN: a preliminary experimental analysis. In *Proceedings of the ESANN 2010*, pages 333–338. d-side, 2010.

[4] C. Gallicchio and A. Micheli. Graph echo state networks. In *Proceedings of the IJCNN 2010*, pages 2159–2166. IEEE, 2010.

[5] R. F. Reinhart and J. J. Steil. Reservoir regularization stabilizes learning of echo state networks with output feedback. In *Proceedings of the ESANN 2011*, pages 59–64. Ciaco - i6doc.com, 2011.

[6] F. Wyffels, B. Schrauwen, and D. Stroobandt. Stable output feedback in reservoir computing using ridge regression. In *Artificial Neural Networks - ICANN 2008*, volume 5163, pages 808–817. Springer, 2008.

[7] S. E. Fahlman and C. Lebiere. The cascade-correlation learning architecture. In *Advances in Neural Information Processing Systems (NIPS)*, pages 524–532, 1989.

[8] E. Littmann and H. Ritter. Learning and generalization in cascade network architectures. *Neural Computation*, 8:1521–1539, 1996.

[9] P. Tiňo, B. Hammer, and M. Bodén. Markovian bias of neural-based architectures with feedback connections. In *Perspectives of Neural-Symbolic Integration*, pages 95–133. Springer-Verlag, 2007.

[10] C. Helma, R. King, and S. Kramer. The predictive toxicology challenge 2000-2001. In *Bioinformatics*, number 17, pages 107–108, 2001.

[11] H. Fröhlich, J.K. Wegner, F. Sieker, and A. Zell. Optimal assignment kernels for attributed molecular graphs. In *Proceedings of the ICML 2005*, pages 225–232. ACM, 2005.