ReservoirPy: Efficient Training of Recurrent Neural Networks for Timeseries Processing
Xavier Hinaut, Nathan Trouvain

To cite this version:
Xavier Hinaut, Nathan Trouvain. ReservoirPy: Efficient Training of Recurrent Neural Networks for Timeseries Processing. EuroSciPy 2022 - 14th European Conference on Python in Science, Aug 2022, Basel, Switzerland. hal-03780006

HAL Id: hal-03780006
https://inria.hal.science/hal-03780006
Submitted on 18 Sep 2022

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L’archive ouverte pluridisciplinaire HAL, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d’enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.
Reservoir Computing paradigm

Reservoir Computing is a simple user-friendly library based on Python scientific modules. It provides a flexible interface to implement efficient Reservoir Computing (RC) [2] architectures with a particular focus on Echo State Networks (ESN) [1]. Advanced features of Reservoir allow to improve computation time efficiency on a simple laptop compared to basic Python implementation. Some of its features are offline and online training, parallel implementation, sparse matrix computation, fast spectral initialization, advanced learning rules (e.g. Intrinsic Plasticity) etc. It also makes possible to easily create complex architectures with multiple reservoirs (e.g. deep reservoirs), readouts, and complex feedback loops. Moreover, graphical tools are included to easily explore hyperparameters with the help of the hyperopt library. It includes several tutorials exploring exotic architectures and examples of scientific papers reproduction.

Reservoir Computing state update
\[ y(t) = \left( 1 - \beta \right) y(t) + \beta (W u(t) + W y(t-1)) \]

Output (read-out) update
\[ y(t) = W^T y(t) \]

- \( u(t) \): Inputs
- \( W, W' \): Input, recurrent, output matrices
- \( W' \) and \( W \) matrices are kept random
- Only \( W' \) is trained (e.g. ridge regression)
- \( \beta \) : time constant of reservoir units
- read-out (if \( \beta \to 0 \) then read-out is approximated)
- \( f \) : activation function (usually tanh)

Evaluation
\[ \text{predictions} = \text{run}(\text{esn}, \text{X}) \]

Node & model creation
A node is an independent operator that applies a function on some data, potentially in a recurrent way. A model consists nodes together to compose operators.

ESNs (Echo State Network) is a specific instance of Reservoir Computing with firing rate neurons

Why repository? [3]
- Free dedicated, modular, reusable tools.
- Other open source tools: PyDE, EchoTools, matplotlib scripts...
- Possibility of using existing machine learning frameworks and libraries.

Node
Base object used to apply functions defined as: \( f(\text{in}) \to \text{out} \)

Model
Composition of nodes

Building blocks
Node
Base object used to apply functions defined as: \( f(\text{in}) \to \text{out} \)

Model
Composition of nodes

Interactive Hyperparameters
Random search on Lorenz attractor
Graphical tools to explore hyperparameters are included (using Hyperopt [4]).
With 125 random sets of HPs you can already have a good idea of the landscape.
SM: spatial noise
LR: look rule
ridge: regularization parameter.
See [4] for more details on a general method to optimize HPs for reservoirs.
See [7] to understand why random search is better than grid search.

Sound classification example [5]

Dataset [8]
- 495 canary songs
- 40 labels
- MFC preproc.

Canary using spectrogram with labels

CPU training time [ssec]: LSTM vs. Reservoir
Reservoir learn with less data than LSTM (with equal number of trainable parameters).

REFERENCES