Echo State Networks

Jyotirmaya Shivottam

S-Lab

September 12, 2023

Problem Overview

- LTSF - Long-Term Sequence Forecasting
  - Predicting the future evolution of a time-series, given its history.
- Approaches
  - Linear models - Auto-Regressive Integrated Moving Average (ARIMA), etc.
  - Recurrent Neural Networks (RNNs)
    - Echo State Network (ESNs)
    - Long Short-Term Memory (LSTM)
  - Self-attention based methods - Transformers

Echo State Networks (ESNs)

- A type of RNNs. Also a type of Reservoir Computing. Similar to Liquid State Machines (LSMs).
- Designed to capture transient + long-term dependencies.
- **Only the readout layer is trained**; the internal weights are randomly initialized and kept fixed.
- Training process is very fast and efficient, enabling online-learning.
- Two key ideas:
  - Random initialization of internal weights
  - Sparse connectivity

ESN Architecture

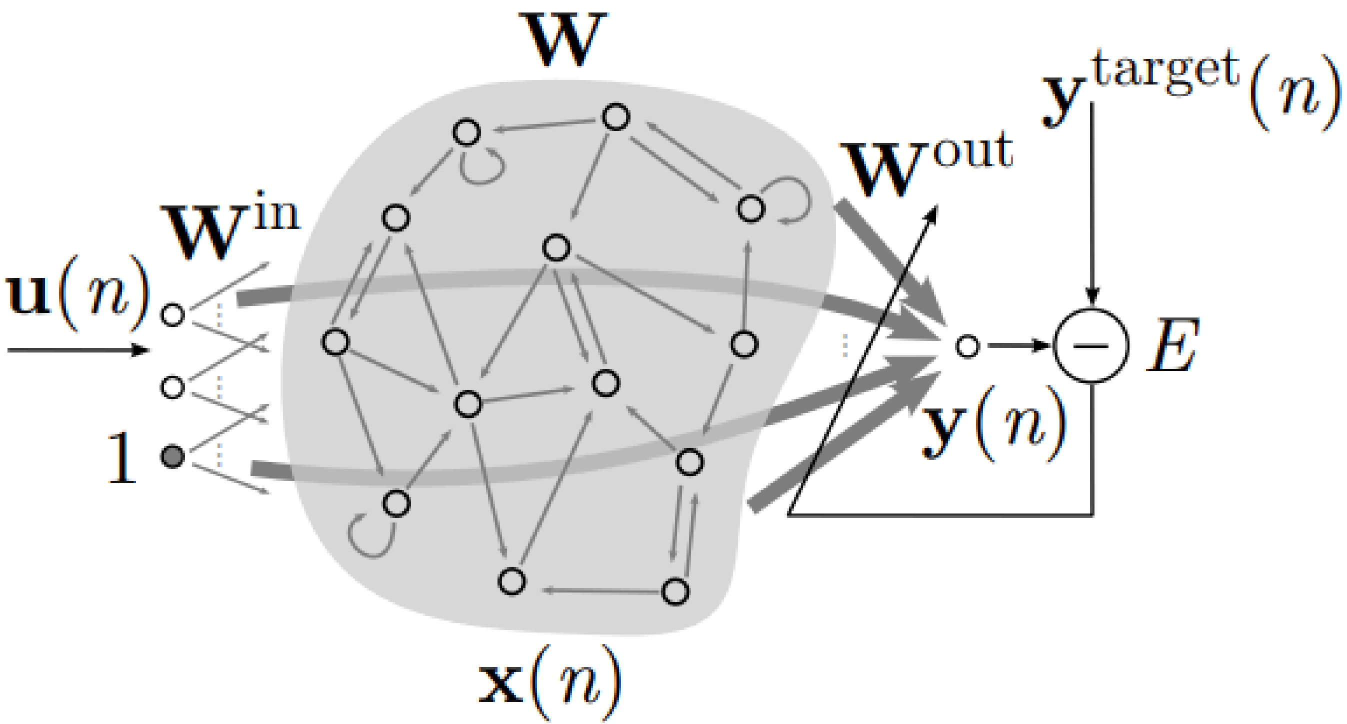
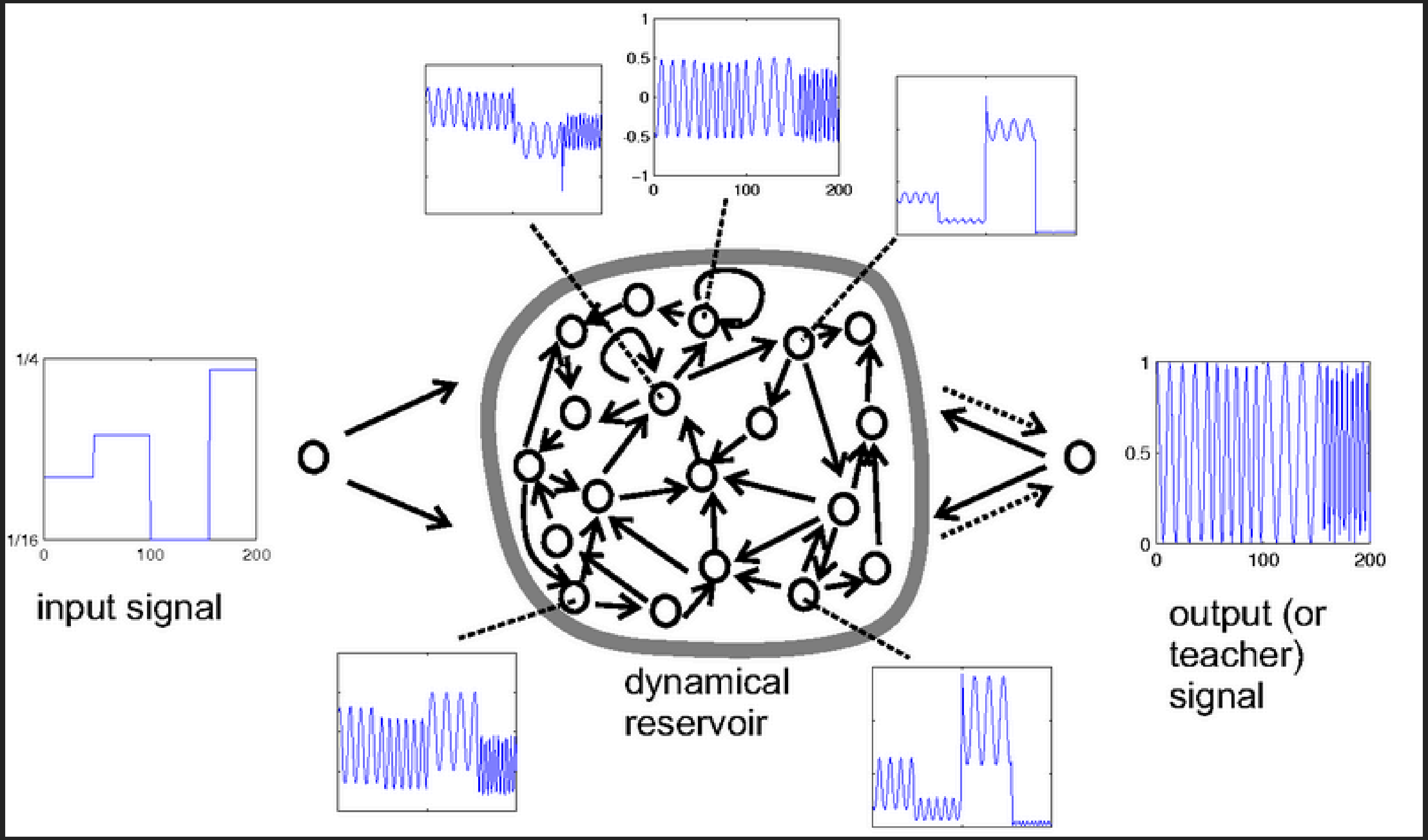


Fig. 1: An echo state network.

ESN Architecture



Echo State Property (ESP)

- The random initialization of the internal weights is done in such a way that the network has the `Echo State Property` (ESP).
- Mathematically, an ESN has ESP if $\rho(W) < 1$ where W is the internal weight matrix for a 0-input signal, and spectral radius, $\rho(W) = \max_i |\lambda_i|$.
- Empirically, W is rescaled to have $\rho(W) < 1$ during initialization.

Echo State Property (ESP)

- The idea is that the network should not be too sensitive to the initial conditions.
  - Sparse W and ESP ensure that for a given signal, the network (ideally) takes the same (similar) pathways each time to converge to the same state, irrespective of the initial state $\rightarrow$ *Short-term memory*.
  - If the network generalizes well, it should be able to find the right pathways for a new signal, by decomposing it into a non-linear combination of the pathways it has already seen (ideal case).
  - In other words, *the dynamics of the internal state of the reservoir should echo, i.e., "reflect" / "mimic" the dynamics of the input signal*.

Some formulae

- $x(t) = (1-\alpha)x(t-1) + \alpha f(W_{in}u(t) + Wx(t-1))$
- $y(t) = W_{out}x(t)$
- $W_{in} \in \mathbb{R}^{N \times K}$, $W \in \mathbb{R}^{N \times N}$, $W_{out} \in \mathbb{R}^{L \times N}$
- $\alpha \in [0,1]$, $x(t) \in \mathbb{R}^N$, $u(t) \in \mathbb{R}^K$, $y(t) \in \mathbb{R}^L$.
- f is a non-linear activation function, e.g., $\tanh$, σ, etc.
- α is the leaking rate.
- Usually, ridge regression (Tikhonov Regularization) is used to train W_{out}.
- SVM, Linear layers etc. can also be used.

Main hyperparameters

- Reservoir size, N - $\uparrow \rightarrow \uparrow$ memory / "capacity".
- Spectral radius, $\rho(W)$
  - $\rho(W) < 1 \rightarrow$ ESP
  - $\uparrow \rightarrow \uparrow$ chaotic dynamics
- Leaking rate, $\alpha \in [0, 1]$
  - $\alpha = 1 \rightarrow$ no memory
  - $\alpha = 0 \rightarrow$ no learning
- Input scaling, β - Regulates non-linearity.

Some other considerations

- Different types of ESN
  - $\rightarrow$ *MultiReservoirESN*
    - GroupedESN
    - DeepESN
    - HierarchicalESN
  - GraphESN
  - DynamicGraphESN
- Usage areas
  - Physical systems, mainly for chaotic systems
  - In silico
  - $\rightarrow$ Neuromorphic computing

Advantages / Disadvantages

- Advantages
  - Very fast training
  - Online learning is possible
  - Only effective method for chaotic systems
  - Can be used for neuromorphic / in silico computing
- Disadvantages
  - Hyperparameter tuning is difficult $\rightarrow$ research ongoing.
    - Random / Grid Search
    - Evolutionary Algorithms
    - Genetic Algorithms
  - Reservoir size does not scale well for problems such as in NLP, though MultiReservoirESN can help.

Results

- Refer to code.

- On ETTh1 (Real-world dataset)
  - Task: To predict transformer "Oil Temperature" - 336 steps ahead

Model	MSE	R^2	Time (in s)
ESN	0.0059	-0.87	3.7
LSTM	0.0038	-3.10	37
Linear	0.0909	--	31
DLinear	0.0910	--	28

- On Mackey-Glass (Synthetic dataset)
  - Task: To predict evolution of a chaotic system - 336 steps ahead

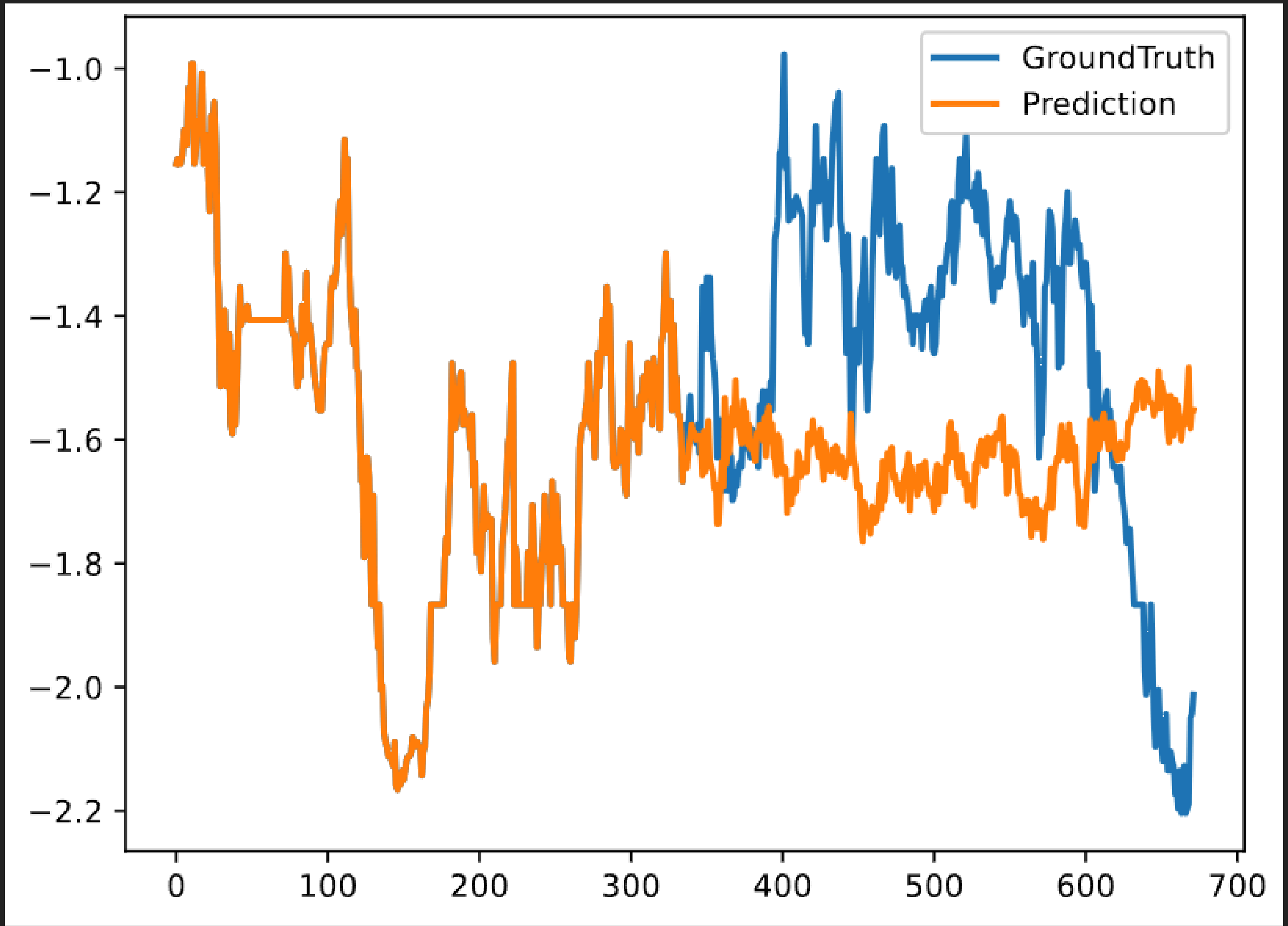
Model	MSE	R^2	Time (in s)
ESN	0.021	0.62	3.7
LSTM	0.0331	-0.66	45

- Result caveats
  - No hyperparameter optimization done for ESN & LSTM here.
  - They are not GPU-accelerated.
  - Linear models are using GPU.

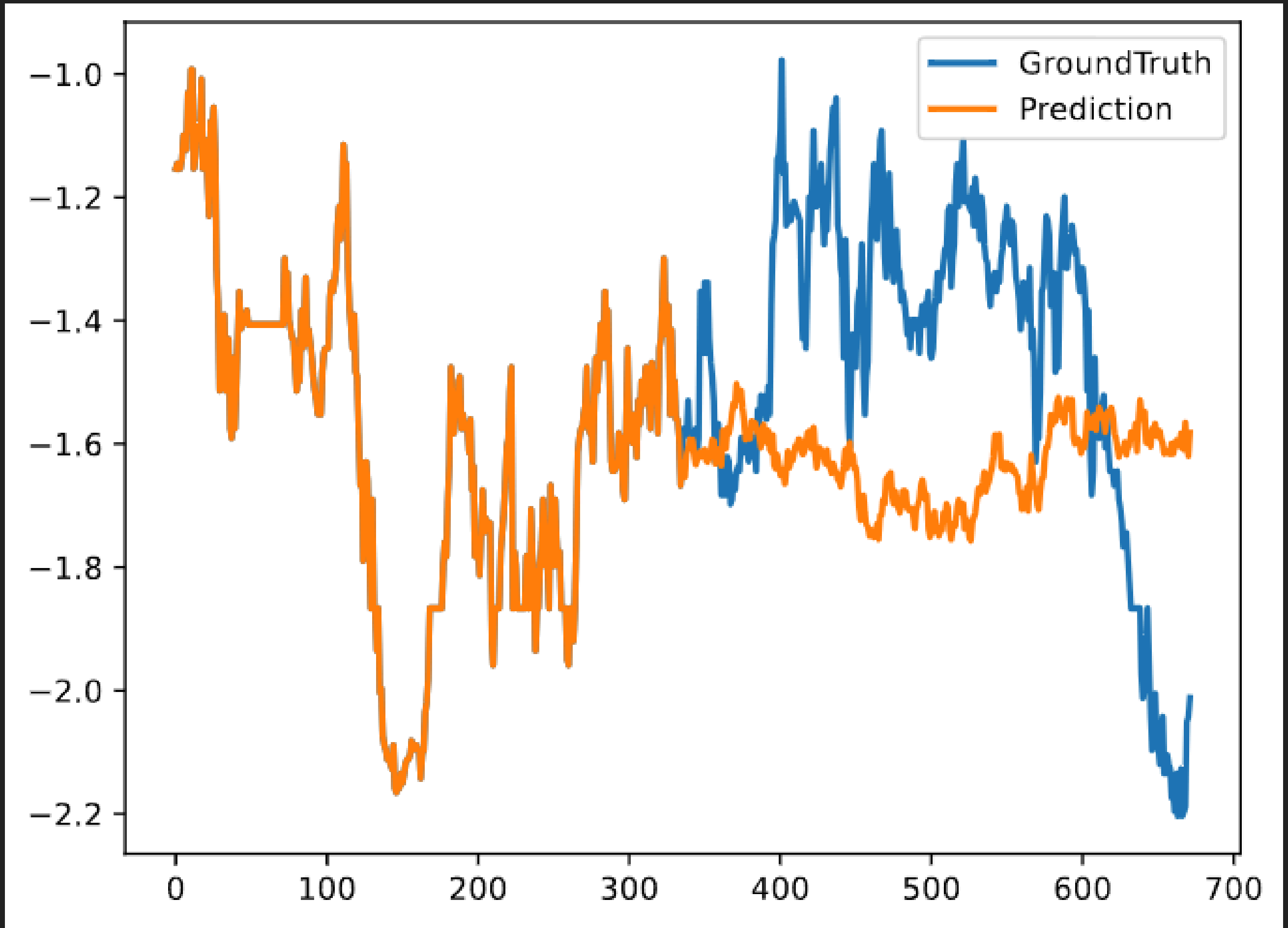
Comparison with Linear & Transformers

- "Linear" as in Linear NN layer, not Linear Regression or ARIMA.
- Refer to *Are Transformers Effective for Time Series Forecasting?*.

Results - Linear on ETTh1



Results - DLinear on ETTh1



Final comments

- Based on own results & literature survey:

$$\text{ESN} > \text{GRU} \sim \text{LSTM} > \text{Linear} \sim \text{Transformer for LTSF}$$

- However, there is a literature gap here:
  - None of the Transformer papers compares with ESN / RC (Sample size = 6+).
  - They rarely compare against LSTM; mostly other transformer variants or ARIMA.
  - Transformer papers ~~usually~~ *sometimes* don't plot predictions. When they do, they don't match very well with GT.
  - Transformer papers use MSE / MAE as the metric (demonstrably not very useful w/o additional context).

M-Y	Model	vs LSTM	Plots output
Mar 2021	Informer	N.A.	✅
Jan 2022	Autoformer	✅	✅ (#1)
Feb 2022	TS2Vec	✅	✅
Jun 2022	FEDFormer	❌	❌
June 2022	CATN	✅	❌
Aug 2022	(N/D)Linear⭐	❌	✅
Feb 2023	CrossFormer	✅	✅ (#1)
Mar 2023	PatchTST	❌	✅ (#1)
May 2023	CARD	❌	✅

- (#1) - In Appendix.
- ⭐ - Personally tested.
- CARD - DLinear results are anomalous for some datapoints.

Additional comments

- No paper compares with ESN / RC.
- Only PatchTST seems to outperform (N/D)Linear. Ignoring CARD's anomalous data, the difference with (N/D)Linear is minimal.
- Informer seems to perform the worst in all papers.
- TS2Vec is not compared in any paper, except PatchTST. Informer fares better here, compared to even its own paper.

References

- Echo State Network - Scholarpedia
- ReservoirPy - Hyperopt
- A Practical Guide to Applying Echo State Networks - Mantas Lukosevicius - 2012
- Are Transformers Effective for Time Series Forecasting?
- *Too many to put here. I'll compile & share on iLibrary*

Thank you! Questions?