Lecture 18: Introduction to Reservoir Computing (Continuation of Lecture 17)

Basics and implementation of a standard ESN Learning

1. Training:
   - To train an RC model from a given initial condition $x(0)$, we first integrate the real dynamics in Eq.(1) to obtain N_{train} additional states $\{x(t)\}_{t=1,\ldots,N_{\text{train}}}$.

Basics and implementation of a standard ESN Learning

1. Training:
    - To train an RC model from a given initial condition $x(0)$, we first integrate the real dynamics in Eq.(1) to obtain N_{train} additional states $\{x(t)\}_{t=1,\ldots,N_{\text{train}}}$.
    - We then iterate the reservoir dynamics in Eq.(4) for N_{train} times from $r(0) = 0$, using the training data as inputs (i.e., $u(t) = x(t)$). This produces a corresponding sequence of reservoir states, $\{r(t)\}_{t=1,\ldots,N_{\text{train}}}$.

Basics and implementation of a standard ESN Learning

1. Training:
   - To train an RC model from a given initial condition $x(0)$, we first integrate the real dynamics in Eq.(1) to obtain N_{train} additional states $\{x(t)\}_{t=1,\dots,N_{\text{train}}}$.
   - We then iterate the reservoir dynamics in Eq.(4) for N_{train} times from $r(0) = 0$, using the training data as inputs (i.e., $u(t) = x(t)$). This produces a corresponding sequence of reservoir states, $\{r(t)\}_{t=1,\dots,N_{\text{train}}}$.
   - Finally, we solve for the output weights W_{out} that render Eq.(3) the best fit to the training data using Ridge Regression with Tikhonov regularization:

$$W_{out} = XR^T(RR^T + \lambda\mathbb{I})^{-1} \tag{5}$$

     where, X and R are matrices whose columns are the $x(t)$ and $r(t)$, respectively, for $t = 1, \dots, N_{\text{train}}$. $\mathbb{I}$ is the $N_{res} \times N_{res}$ identity matrix, and $\lambda > 0$ is a regularization hyperparameter that prevents ill-conditioning of the weights, which can be symptomatic of overfitting the data.

Basics and implementation of a standard ESN Learning

1. Training:
   - To train an RC model from a given initial condition $x(0)$, we first integrate the real dynamics in Eq.(1) to obtain N_{train} additional states $\{x(t)\}_{t=1,\dots,N_{\text{train}}}$.
   - We then iterate the reservoir dynamics in Eq.(4) for N_{train} times from $r(0) = 0$, using the training data as inputs (i.e., $u(t) = x(t)$). This produces a corresponding sequence of reservoir states, $\{r(t)\}_{t=1,\dots,N_{\text{train}}}$.
   - Finally, we solve for the output weights W_{out} that render Eq.(3) the best fit to the training data using Ridge Regression with Tikhonov regularization:

   $$W_{out} = XR^T(RR^T + \lambda\mathbb{I})^{-1} \tag{5}$$

   where, X and R are matrices whose columns are the $x(t)$ and $r(t)$, respectively, for $t = 1, \dots, N_{\text{train}}$. $\mathbb{I}$ is the $N_{res} \times N_{res}$ identity matrix, and $\lambda > 0$ is a regularization hyperparameter that prevents ill-conditioning of the weights, which can be symptomatic of overfitting the data.

2. Training Error: A training error E_{train} can be computed from

$$E_{\text{train}} = \frac{\left\|x_{out}(t) - x(t)\right\|}{\left\|x(t)\right\|} = \frac{\left\|W_{out} \cdot r(t) - x(t)\right\|}{\left\|x(t)\right\|} \tag{6}$$

where $x_{out}(t)$ is machine extracted data given by Eq.(3) and $x(t)$ target data obtained from the integrating the real system in Eq.(1), for $t = 1, \dots, N_{\text{train}}$, and where $\|\cdot\|$ denotes standard deviation.

Basics and implementation of a standard ESN Learning

3. Prediction:
   - Before prediction, it is good practice first to test the trained RC. To test a trained RC model from a given initial condition $x(0)$, we first integrate the true dynamics Eq.(1) forward in time to obtain a total of $N_{\text{warmup}} \geq 0$ states $\{x(t)\}_{t=1,\dots,N_{\text{warmup}}}$.

Basics and implementation of a standard ESN Learning

3. Prediction:
   - Before prediction, it is good practice first to test the trained RC. To test a trained RC model from a given initial condition $x(0)$, we first integrate the true dynamics Eq.(1) forward in time to obtain a total of $N_{\text{warmup}} \geq 0$ states $\{x(t)\}_{t=1,\dots,N_{\text{warmup}}}$.
   - During the first N_{warmup} iterations of the RC dynamics of Eq.(4), the input term $u(t)$ comes from the real trajectory (given by $x(t)$ of Eq.(2)), i.e., $u(t) = x(t)$.

Basics and implementation of a standard ESN Learning

3. Prediction:
   - Before prediction, it is good practice first to test the trained RC. To test a trained RC model from a given initial condition $x(0)$, we first integrate the true dynamics Eq.(1) forward in time to obtain a total of $N_{\text{warmup}} \geq 0$ states $\{x(t)\}_{t=1,\ldots,N_{\text{warmup}}}$.
   - During the first N_{warmup} iterations of the RC dynamics of Eq.(4), the input term $u(t)$ comes from the real trajectory (given by $x(t)$ of Eq.(2)), i.e., $u(t) = x(t)$.
   - Thereafter, prediction starts when the reservoir system in Eq.(4) runs autonomously. This is achieved by replacing the input term $u(t)$ in Eq.(4) with the model's own output at the previous iteration (i.e., $u(t) = x_{out}(t) = W_{out} \cdot r(t)$). This creates a closed-loop system given by:

     $$r(t+1) = (1-\alpha)r(t) + \alpha \tanh\Big(W_{res} \cdot r(t) + W_{in} \cdot W_{out} \cdot r(t) + b\Big) \tag{7}$$

     which we iterate without further input $x(t)$ from the real system of Eq.(1).

Basics and implementation of a standard ESN Learning

3. Prediction:
   - Before prediction, it is good practice first to test the trained RC. To test a trained RC model from a given initial condition $x(0)$, we first integrate the true dynamics Eq.(1) forward in time to obtain a total of $N_{\text{warmup}} \geq 0$ states $\{x(t)\}_{t=1,\ldots,N_{\text{warmup}}}$.
   - During the first N_{warmup} iterations of the RC dynamics of Eq.(4), the input term $u(t)$ comes from the real trajectory (given by $x(t)$ of Eq.(2)), i.e., $u(t) = x(t)$.
   - Thereafter, prediction starts when the reservoir system in Eq.(4) runs autonomously. This is achieved by replacing the input term $u(t)$ in Eq.(4) with the model's own output at the previous iteration (i.e., $u(t) = x_{out}(t) = W_{out} \cdot r(t)$). This creates a closed-loop system given by:

$$r(t+1) = (1-\alpha)r(t) + \alpha \tanh\Big(W_{res} \cdot r(t) + W_{in} \cdot W_{out} \cdot r(t) + b\Big) \tag{7}$$

     which we iterate without further input $x(t)$ from the real system of Eq.(1).
   - The autonomous reservoir states $r(t)$ from Eq.(7) is used to calculate the predicted output $x^{p}_{out}(t)$ of the input data $x(t)$ for $t > N_{\text{warmup}}$ as:

$$x^{p}_{out}(t) = W_{out} \cdot r(t) \tag{8}$$

Basics and implementation of a standard ESN Learning

4. Prediction Error: A prediction error E_{pred} can be computed similarly to the training error as:

$$E_{\text{pred}} = \frac{\left\| x_{out}^{p}(t) - x(t) \right\|}{\left\| x(t) \right\|} = \frac{\left\| W_{out} \cdot r(t) - x(t) \right\|}{\left\| x(t) \right\|} \qquad (9)$$

where $x_{out}^{p}(t)$ is machine extracted data given by Eq.(8) and $x(t)$ target data obtained from the integrating the real system in Eq.(1), for $t > N_{\text{watmup}}$, and where $\|\cdot\|$ denotes standard deviation.

Basics and implementation of a standard ESN Learning

NOTE: The (normalized) root-mean-square error can also be used to measure the training and prediction errors given in Eq.(6) and Eq.(9).

- Given an RC predicted trajectory $\overline{x}(t)$ and a corresponding trajectory of the real system $x(t)$ – each of length N – we calculate the root-mean-square error (RMSE) as:

$$\text{RMSE} = \sqrt{\frac{1}{N}\sum_{t=1}^{N}\left\|x(t)-\overline{x}(t)\right\|^2} \tag{10}$$

  where, this time, $\|\cdot\|$ denotes the Euclidean norm.

- To obtain a normalized version of this (NRMSE) – which is frequently used as the objective function to optimize standard RC hyperparameters – we divide the RMSE by the range of the data in the real system i.e.,

$$NRMSE = \frac{RMSE}{x_{i,max}-x_{i,min}} \tag{11}$$

  where the maximum ($x_{i,max}$) and minimum ($x_{i,min}$) for dimension $i = 1, \dots, n$ of the state space are calculated over the corresponding training data.

Basics and implementation of a standard ESN Learning

- NRMSE values range between 0 and 1, with 0 indicating a perfect fit and 1 representing the worst fit. This normalized scale allows for easier model performance interpretation and comparison across different contexts.
- When comparing multiple models or algorithms, the NRMSE can provide a more reliable basis for model selection. By normalizing the error, the NRMSE allows for a fair comparison of model performance, regardless of the specific scale or variability of the data.