Department of Computer Science, Electrical and Space Engineering
Luleå University of Technology

D7041E "Applied artificial intelligence"

(Please report any found inconsistencies to Evgeny/Denis as soon as you find them)

LAB 6: Time-series and Reservoir Computing

1. Introduction

In this lab we will work with data-driven models for predicting behaviour of dynamic systems and classification of time-series. In the case of modelling dynamic systems the task is to predict the future states of the system using historic data. In the case of time-series classification there is a need to assign a class to a given signal. In this lab we will use a particular approach suitable for both tasks. It is called Reservoir Computing. In particular, you will implement Echo State Networks (ESN), which is one of the most known architectures for Reservoir Computing.

In this lab you are not provided with the code. There are only textual guidelines and references to the literature. The challenge is to implement the whole approach and the experiments from the scratch.

2. Echo State Networks

Your task here is to implement ESN in Python. Do not be afraid it is a very simple architecture. Please start by reading the paper "A Practical Guide to Applying Echo State Networks" by M. Lukoševičius. It has all the details you need for the implementation.

Some guiding questions:

1. What does teacher forcing mean?
2. What is ridge regression? Why is it useful?
3. How to achieve echo state property?
4. What is Root-Mean-Square Error (RMSE)?

Useful tips:

1. For this lab you do not need to implement leaky-integrated discrete-time continuous-value neurons (i.e., α=1).
2. You do not have to worry about the sparsity of the recurrent weight matrix **W** but you have to worry about the echo state property of this matrix.
3. Do not implement the function for solving regression yourself. Check what NumPy or SciPy could offer you.

3. Modeling of dynamic systems with ESNs

In order to check the correctness of your ESN implementation we will first let ESNs learn the behavior of two dynamic systems.

The first system is a sinusoidal signal, which is an example of a simple dynamic system with the constant cyclic behavior. The ground truth signal has the following form:

$$y(n) = 0.5\sin(n/4),$$

where n is the current time step.

In this task the output also acts as an input and it is necessary to predict the value of a function at the next time step.

In order to do the experiment:

1. Generate a sequence for 4,000 time steps; n is in the range between 1 and 4,000.
2. Take the first 3,000 steps for the training and leave the last 1,000 steps for the testing.
3. Fix the reservoir size to N_x = 1,000 neurons.
4. Generate input weight matrix $\mathbf{W}^{in}$ from uniform distribution between -1 and 1. Scale $\mathbf{W}^{in}$ by 0.2.
5. Do not forget to add a bias input neuron, which constantly feeds 1 to the reservoir.
6. Set the spectral radius (feedback strength) for the reservoir connection matrix to ρ = 0.8.

7. Feed-in ground truth-values of the signal to the ESN during the training.
8. For time step n the teaching forcing signal is the value of the signal at time step n-1, i.e., $y(n\text{-}1)$.
9. Using the state of reservoirs for the training sequence and the corresponding values of the teaching forcing signal estimate the readout matrix.
10. When estimating the readout matrix discard the first 1,000 steps in the training sequence from the calculation.
11. In the operating phase, the network acts as the generator of the signal (thanks to trained readout matrix) feeding its previous prediction (at time $n-1$) back to the reservoir.
12. At the beginning of the operating phase, initialize the reservoir with the last state for the training sequence.
13. Make predictions for all 1,000 testing time steps.
14. Repeat the experiment for 10 different random initializations of ESNs.
15. Plot the ground truth for the testing sequence versus the predictions obtained for ESN. Summarize 10 independent runs using mean values of predictions.

Note. It is expected that your ESN would be able to predict the sinusoidal signal extremely well even for a long-scale prediction horizon.

The second system is a Mackey-Glass series, which is generated by the nonlinear time delay differential equation. The following equation is used to obtain the Mackey-Glass series:

$$\frac{dy(n)}{dn} = \beta \frac{y(n-\tau)}{1+y^a(n-\tau)} - \gamma y(n).$$

where n is time; $y(n)$ is the output of the time-series at time n. β, a, γ, τ are the parameters of the equation.

Similar to the sinusoidal signal, in the operating phase you have to predict the value of a function at the next time step. The prediction of the ESN at time step n acts as an input to the ESN at time step $n+1$.

In order to do the experiment:

1. Generate a Mackey-Glass series for 4,000 time steps; n is in the range between 1 and 4,000. You have the Python function in "mackey_glass_gen.py", which generates the time-series. Keep the default parameters for β, a, γ, τ.
2. Take the first 3,000 steps for the training and leave the last 1,000 steps for the testing.

3. Fix the reservoir size to N_x = 1,000 neurons.
4. Generate input weight matrix $\mathbf{W}^{in}$ from uniform distribution between -1 and 1. Scale $\mathbf{W}^{in}$ by 0.2.
5. Do not forget to add a bias input neuron, which constantly feeds 1 to the reservoir.
6. Set the spectral radius (feedback strength) for the reservoir connection matrix to $\rho = 0.8$.
7. Feed-in ground truth-values of the signal to the ESN during the training.
8. For time step n the teaching forcing signal is the value of the signal at time step n-1, i.e., $y(n\text{-}1)$.
9. Using the state of reservoirs for the training sequence and the corresponding values of the teaching forcing signal estimate the readout matrix.
10. When estimating the readout matrix discard the first 1,000 steps in the training sequence from the calculation.
11. In the operating phase, the network acts as the generator of the signal (thanks to trained readout matrix) feeding its previous prediction (at time $n-1$) back to the reservoir.
12. At the beginning of the operating phase, initialize the reservoir with the last state for the training sequence.
13. Make predictions for all 1,000 testing time steps.
14. Repeat the experiment for 10 different random initializations of ESNs using the same time-series.
15. Plot the ground truth for the testing sequence versus the predictions obtained for ESN. Summarize 10 independent runs using mean values of predictions.

Note. It is expected that your ESN would be able to predict the Mackey-Glass series extremely well for the first several hundred of time steps. However, as time progresses the predictions will start to deviate from the ground truth.

4. Time-series classification with ESNs

In this task we will perform classification of time-series with ESNs. For the sake of simplicity we will only work with one univariate dataset called "Swedish Leaf".

In order to do the experiment:

1. Download the dataset at https://www.cs.ucr.edu/%7Eeamonn/time_series_data_2018/
2. You will have separate training and testing datasets.
3. Fix the reservoir size to $N_x = 800$ neurons.
4. Generate input weight matrix $\mathbf{W}^{in}$ from uniform distribution between -1 and 1. Scale $\mathbf{W}^{in}$ by 0.25.
5. Do not forget to add a bias input neuron, which constantly feeds 1 to the reservoir.
6. Set the spectral radius (feedback strength) for the reservoir connection matrix to $\rho = 0.99$.
7. For each signal in the training dataset: initialize the reservoir to zero and feed-in the signal to the ESN. Save the last state of the reservoir.
8. For each signal in the training dataset the output signal, which you will use for the regression, is a one-hot encoding representing the correct class for that signal.
9. Using the saved states of reservoirs for the training dataset and the corresponding one-hot encodings, estimate the readout matrix.
10. In the operating phase, for each signal in the testing dataset: initialize the reservoir to zero and feed-in the signal to the ESN.
11. Use the last state of the reservoir to make predictions from the readout matrix.
12. Use the highest value among the predictions as the predicted class.
13. Calculate accuracy on the testing dataset using the ground truth.
14. Repeat the experiment for 10 different random initializations of ESNs.
15. Report the average accuracy of the predictions obtained from ESN.

Note. It is expected that the ESN's accuracy would be about 0.75.

Congrats, you have just become familiar with the state-of-the-art approach to data-driven modelling of dynamic systems! Well done!