Laboratory Assignment: Neural Networks for temporal data processing (Lab3-1, 2017)

Solve the following assignments, whose completion is required to access the oral examination. Send the assignments all-together (once you have completed all the labs, not only this single one) as a compressed folder including one subfolder for each laboratory (e.g. the name of the subfolder should be LAB3-1 for the first laboratory, then LAB3-2, etc..).

The subfolder for this lab should be called "`Lab3-1`" (Neural Networks for temporal data processing 1) and should include the Matlab structures and all the scripts, as requested in the assignments below (for each assignment there is specific list of structures to provide). Use different sub-folders for the assignments (e.g. "Assignment1", "BonusTrackAssigment1", etc.). You can organize the code as you wish, implementing all the helper functions that you need provided that these are included in the subfolder and are appropriately called in the scripts.

Bonus track assignments are meant to be for those who finish early, but they are not formally required for completing the Lab Assignment.

Useful documentation:
Matlab Neural Network Toolbox User's Guide http://it.mathworks.com/help/pdf_doc/nnet/index.html
Matlab documentation using the help command (e.g. `help train`)
A brief recap of the process for creating, customizing and training neural networks is in the file AdditionalMaterial1.pdf

Assignment #1 – Laser Task

The Laser task consists in a next-step prediction (autoregressive, a particular case of transduction) on a time series obtained by sampling the intensity of a far-infrared laser in a chaotic regime. Import the dataset from Matlab (`load laser_dataset`), rescale values to [-1,1], properly separate input and target data, then split the available data in training (first 5000 time steps), and test set (remaining time steps). Note that for model selection you will use the data in the training set, with a further split in training (first 4000 samples) and validation (last 1000 samples). Try to plot the time series data using the command `plot`.

- Create and train Time Delay Neural Networks (`timedelaynet`) and Recurrent Neural Networks (`layrecnet`), using different values of the hyper-parameters (size of the hidden layer, length of the input delay for timedelaynet, training function, learning parameters, number of training epochs, etc.). As regards the hyper-parameters, you can either (manually) choose a set of values to consider, either (systematically) use a grid.
- Select the best network hyper-parameters <u>on the validation set</u>, the hyper-parameterization with the smallest Mean Squared Error (MSE), e.g. by using the command `immse`.
- Train the selected network on all the training data and evaluate the MSE of such network on the training set and on the test set.

Notes:

- You can use cell2mat for manipulating input and target data, and num2cell for opposite data structure transformation.
  E.g.
  load laser_dataset; %load the laser dataset
  allData = cell2mat(laserTargets); %converts the cell structured data into a row matrix.

- For next -step prediction tasks input and target data can be separated by an appropriate shift of indexing, e.g. `inputData = allData(1:end-1); targetData = allData(2:end);`
- To give a practical insight into the next-step prediction type of tasks, suppose that the input sequence is as follows:

time step	*1*	*2*	*3*	*4*	*5*	…
input value	3	5	9	12	8	…

  then: the desired output (i.e. the target) at step 1 is 5, at step 2 is 9, at step 3 is 12, and so on…

The output of the assignment should then consist in the following data, pertaining to the Time Delay Neural Network and to the Recurrent Neural Network, <u>only to the selected hyper parametrization</u>:

- The script .m file(s)
- The net structure
- The TR (training record) structure
- Training, validation and test Mean Squared Error (e.g. by using `immse` command)
- Target vs output plot, both for training and test data (.fig and .png)
- A plot of the learning curve (.fig and .png)

Bonus Track Assignment #1– Mackey-Glass Task

The Mackey–Glass (MG) time series is a standard benchmark for chaotic time series prediction models. The task of interest for this assignment is a next-step prediction task (autoregressive, a particular case of transduction) on the MG time series.
Import the dataset from the attached file MGtimeseries.mat, properly separate input and target data, then split the available data in training (first 5000 time steps), and test set (remaining time steps). Note that for model selection you will use the data in the training set, with a further split in training (first 4000 samples) and validation (last 1000 samples). Try to plot the time series data using the command `plot`.

- Create and train Time Delay Neural Networks (`timedelaynet`) and Recurrent Neural Networks (`layrecnet`), using different values of the hyper-parameters (size of the hidden layer, length of the input delay for timedelaynet, training function, learning parameters, number of training epochs, etc.). As regards the hyper-parameters, you can either (manually) choose a set of values to consider, either (systematically) use a grid.
- Select the best network hyper-parameters <u>on the validation set</u>, the hyper-parametrization with the smallest Mean Squared Error (MSE), e.g. by using command `immse`.
- Train the selected network on all the training data and evaluate the MSE of such network on the training set and on the test set.

Notes:

- You can use cell2mat for manipulating input and target data, and num2cell for opposite data structure transformation.
- For next -step prediction tasks input and target data can be separated by an appropriate shift of indexing, e.g. `inputData = allData(1:end-1); targetData = allData(2:end);`
- To give a practical insight into the next-step prediction type of tasks, suppose that the input sequence is as follows:

time step	*1*	*2*	*3*	*4*	*5*	…
input value	3	5	9	12	8	…

then: the desired output (i.e. the target) at step 1 is 5, at step 2 is 9, at step 3 is 12, and so on…

The output of the assignment should then consist in the following data, pertaining <u>only to the selected hyper parametrization</u>:

- The script .m file(s)
- The net structure
- The TR (training record) structure
- Training, validation and test Mean Squared Error (`immse` command)
- Target vs output plot, both for training and test data (.fig and .png)
- A plot of the learning curve (.fig and .png)

Bonus Track Assignment #2– NARX networks

Apply NARX networks with different values of the hyper-parameters (size of the hidden layer, length of the input delay, length of the output delay, training function, learning parameters, number of training epochs, etc.) to the previous tasks in Assignments #1 and Bonus Track Assignment #1.