RSCAD Neural Network Library

April 2024

Chapter 1

Library Components

The components in the library can be divided into two, Full Neural Networks and Single Layers. Full Neural Networks are components that include the computation of all layers involved while single layer blocks are used to design Networks one layer at a time.

1.1 Layered Components

There are four different layer components:

- The input Layer (InputLayer.def).
- Hidden Layer (MidLayer.def).
- The Output Layer (OutputLayer.def).
- LSTM Unit (lstmUnit.def).

For the ANN layers, each component has 3 parameters to be selected before compiling, these are numInputs, numOutputs, and activation.

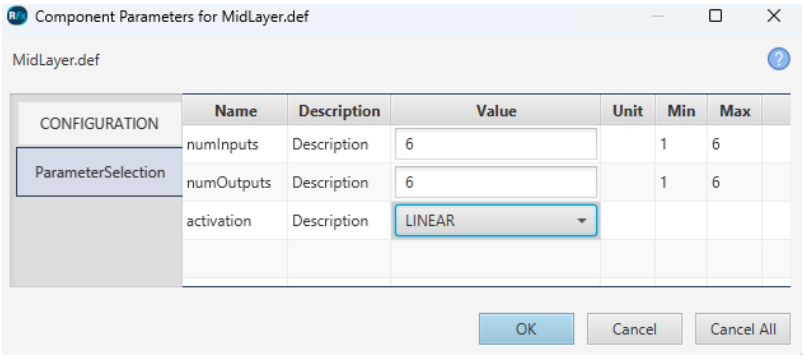

Figure 1.1

Output Layer

The output layer takes in the following inputs:

- Input1 - Input6: The inputs into the layer, used as the outputs of the previous layers
- loss y1/y2: The derivative of the loss in terms of prediction y1 and y2. This was the user can design the loss function outside of the component and feed it as input into the output layer.
- Learning_rate: The rate at which the Neural Network Learns
- Lambda: Regularization Term to drop out weights

the outputs of the layer are:

- y1/y2: The predictions of the network, this output is what should be used for the loss function design.
- Li1-Li6: These are the losses propagated backwards to the previous layer, as in, these outputs are fed as inputs into the previous layer and represent the loss at the node connecting both layers.

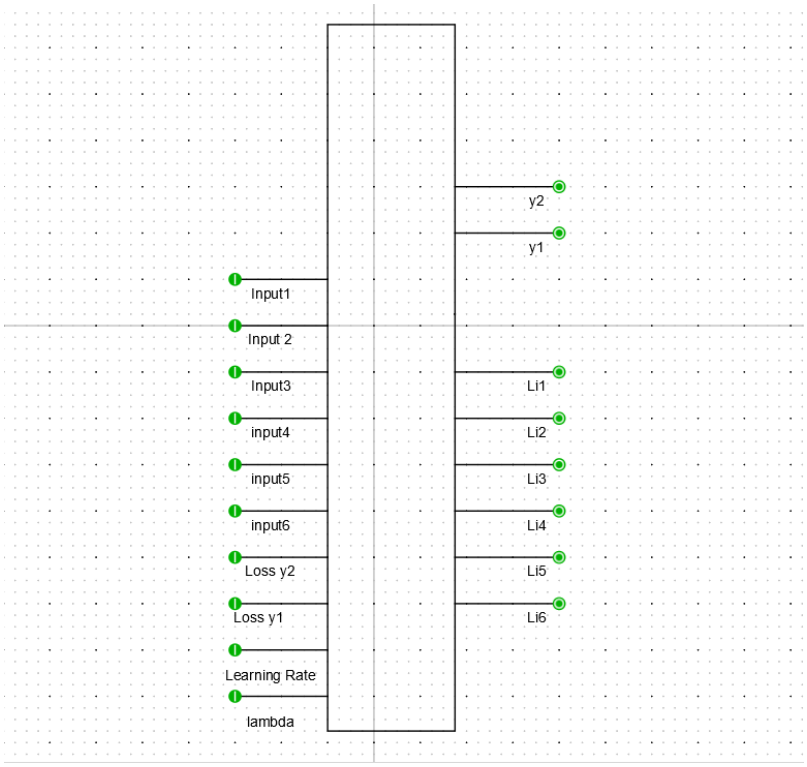

Figure 1.2

Hidden Layer

The hidden layer takes in the following inputs:

- Input1 - Input6: The inputs into the layer, used as the outputs of the previous layers
- L1-L6: The previous loss of the output of this layer, fed back into it from the next layer.
- Learning_rate: The rate at which the Neural Network Learns
- Lambda: Regularization Term to drop out weights

the outputs of the layer are:

- y1-y6: The output nodes of the hidden layer

- Li1-Li6: Losses propagated backwards to the previous layer, these outputs are fed as inputs into the previous layer and represent the loss at the node connecting both layers.

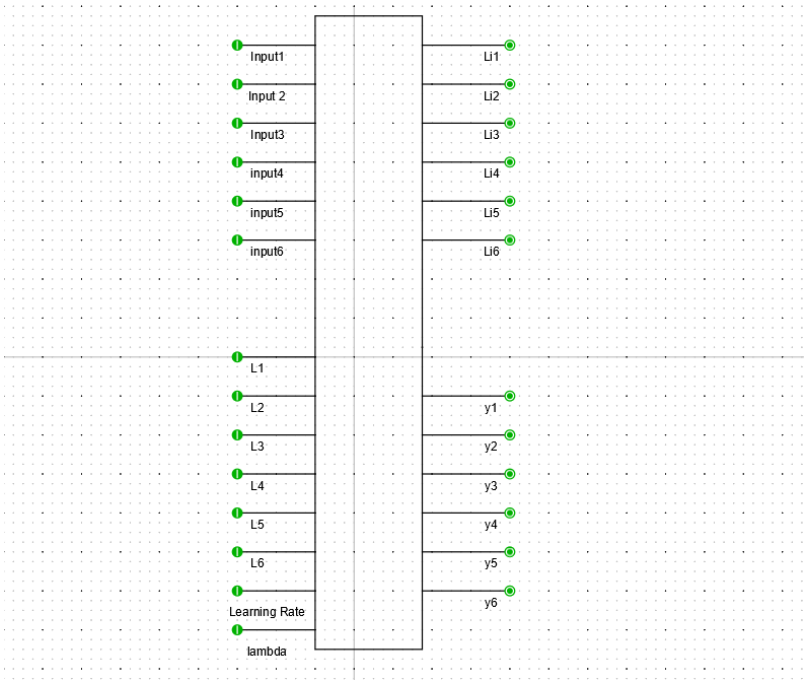

Figure 1.3

Input Layer

The hidden layer takes in the following inputs:

- Input1 - Input4: The input variables into the Neural Network.
- L1-L6: The previous loss of the output of this layer, fed back into it from the next layer.
- Learning_rate: The rate at which the Neural Network Learns
- Lambda: Regularization Term to drop out weights

the outputs of the layer are:

- y1-y6: The output nodes of the hidden layer

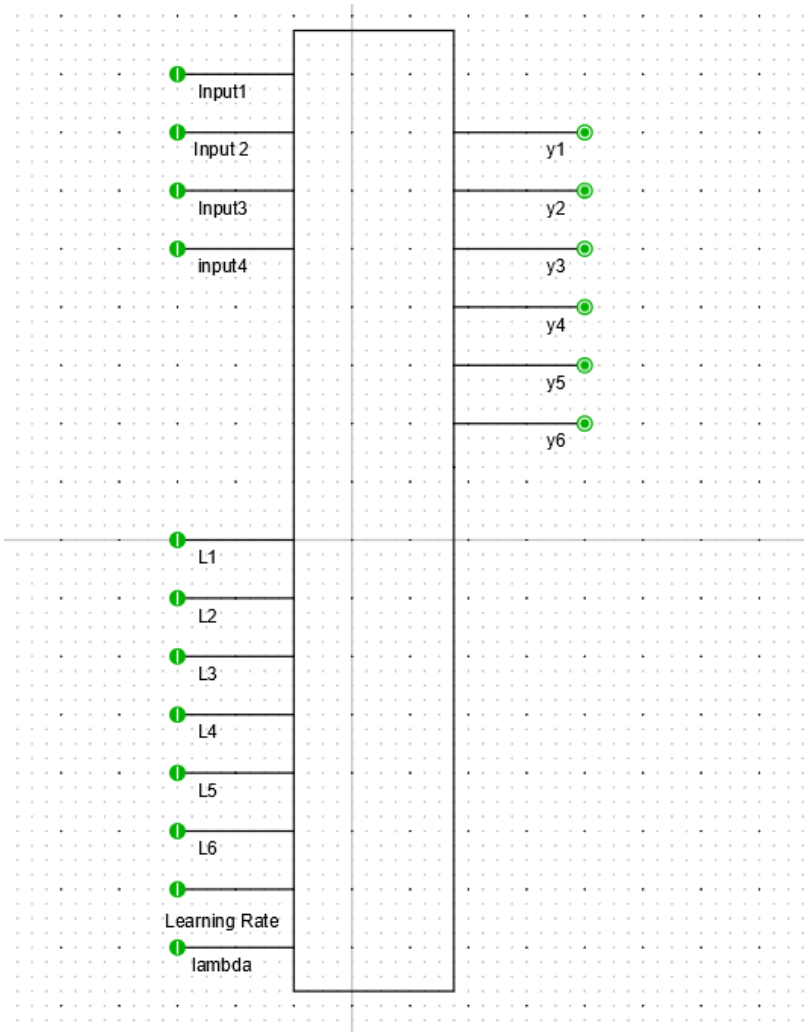

Figure 1.4

Using these components the user can construct their own Neural Network, to give an example, I will construct a 2 and 1 hidden layer Neural Network with 2 inputs and 1 output.

3 Layer Network

First we define the required variables:

- Loss
- Learning Rate
- Regularization Term

We can take a simple example of MSE loss function, the derivative of the loss function with respect to the prediction is simply the difference between the prediction and the target (neglecting the 2 multiplication term). The learning rate and regularization lambda will be sourced from sliders as shown below:

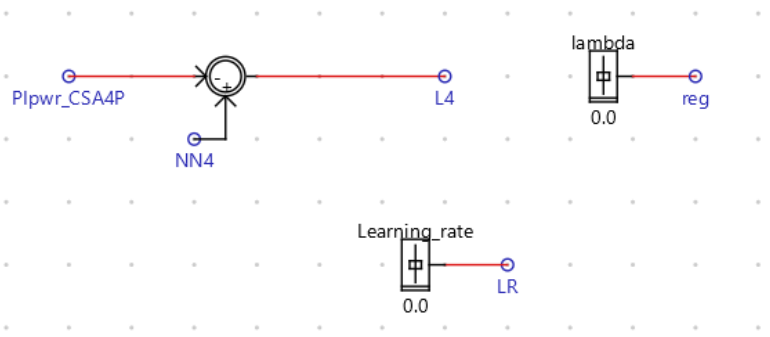

Figure 1.5

To construct a Neural Network with one hidden layer, we can connect the input layer to the output layer as shown below:

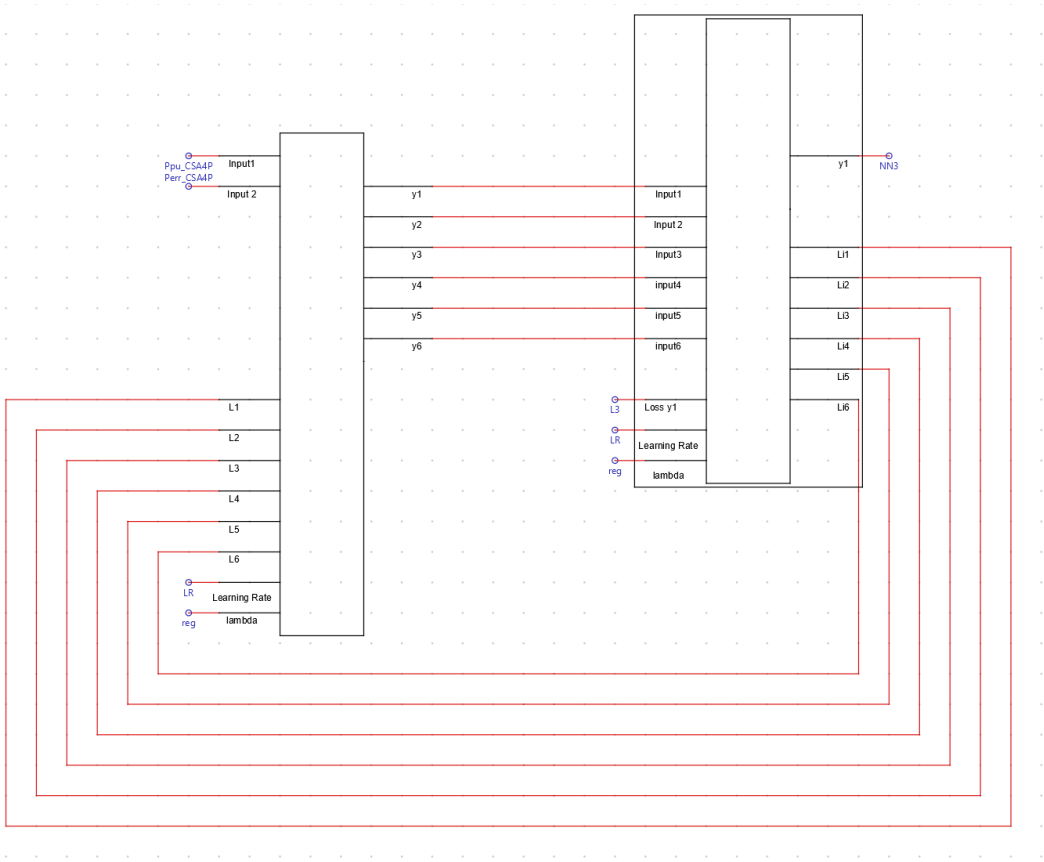

Figure 1.6

4 Layer Network

Using the same logic and parameters, we can construct a 2 hidden layer network by adding a MidLayer.def component as shown below.

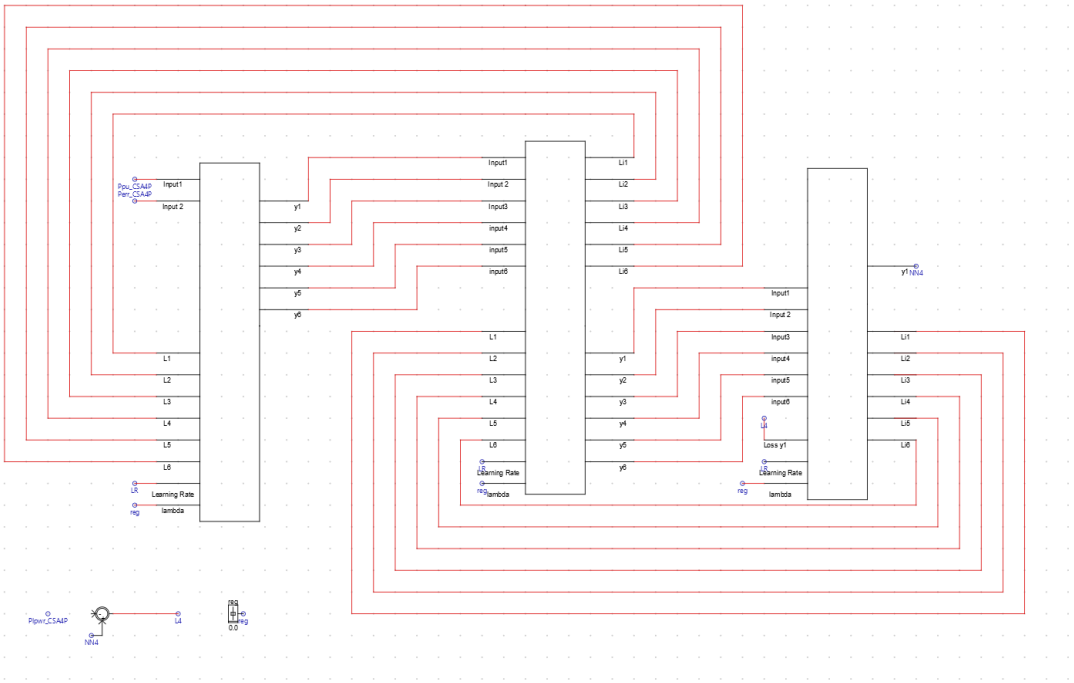

Figure 1.7

Training and Regularization

Since there is no learning rate optimizer, the rate should be adjusted according to the behaviour visible to the user. For example, during training, if the direction of prediction and the direction of the target are not similar during a sudden change in input parameters, then the weights do not have the correct signs. In this case the learning rate will be set high (0.01-0.0001) until the Neural Network output and the Target variable move in the same direction during changes in input parameters. Once that is achieved, the learning rate should be then reduced to somewhere 1e-4 - 1e-9, the user should adjust based on apparent results. Make sure not to keep the same input parameter range for extended duration's of time. If, during training, the shape of the prediction during transient is very odd, you can stop the training during transient and resume after, training is stopped by setting the learning rate to zero.

For the Regularization term, a value of 0.0 can be taken to avoid complications. However, if it is required, then start with a low number (1e-8) and adjust accordingly. Setting lambda larger ($\approx$ 1e-4) can lead to most of the weights dropping out, an appropriate value of lambda most likely is found in the range 1e-8 - 1e-4.

1.2 Full Models

Pre-Training Model