HNet on Tensorflow ver. 0.9

A Connectionist Modeling Tool based on Back-Propagation Algorithm

Manual

Latest Update: 2017-03-09

The newest version is on https://github.com/CODEJIN/HNet_on_Tensorflow

Before start the HNet on Tensorflow

'H-Net on Tensorflow' was tested at Python 3.5 and Tensorflow 0.12.1. and 1.0.1 This program is not guaranteed to work in python 2.x.

***Currently, there are some warning messages on the Tensorflow 1.0.1 version, but anyway it can run. It may be related with Tensorflow module. I think that it will be solved sometime later.**

1 Files

HNet on Tensorflow is constructed by the below files.

HNet_Core.py
HNet_Enum.py
HNet_GUI.py
HNet_UI/__init__.py
HNet_UI/About.py
HNet_UI/Image_Resources_rc.py
HNet_UI/Learning.py
HNet_UI/LearningSetup.py
HNet_UI/Macro.py
HNet_UI/Main.py
HNet_UI/PatternSetup.py
HNet_UI/ProcessSetup.py
HNet_UI/StructureSetup.py

To run, all files is in the folder, except the manual file.
In addition, the save files of every example are in the subfolder 'Examples'.

2 The modules which should be installed

To use H-Net on Tensorflow, user should to install two modules on Python 3.x.

2.1 Tensorflow

There are two types of Tensorflow Python version: GPU and CPU only.

2.1.1 GPU support version

pip install tensorflow-gpu

2.1.2 CPU only version

pip install tensorflow

2.2 Matplotlib

pip install matplotlib

3 Run H-Net

This manual is based on the GUI version. In terminal or console, type the below command:

python HNet_GUI.py

Tutorial

1 Back-propagation Tutorial

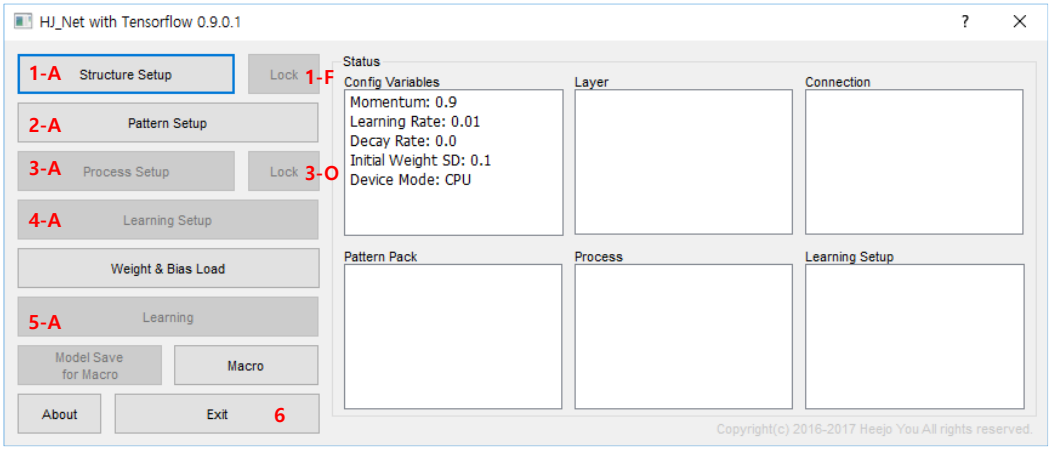

Figure 1. The main window in BP algorithm using

1. Structure Setup
   A. Click ‘Structure Setup’ button
   B. Assign the config variables and click the ‘Submit’ button.
   C. Click the ‘Back-propagation’ tab and insert each layer’s unit size. The ‘XOR5’ example used 10, 200, 5 units about the input, hidden, and output layers, respectively.
   D. Click ‘Make’ button
   E. Click ‘Exit’ button
   F. Click ‘Lock’ button which is located at the right of structure setup button

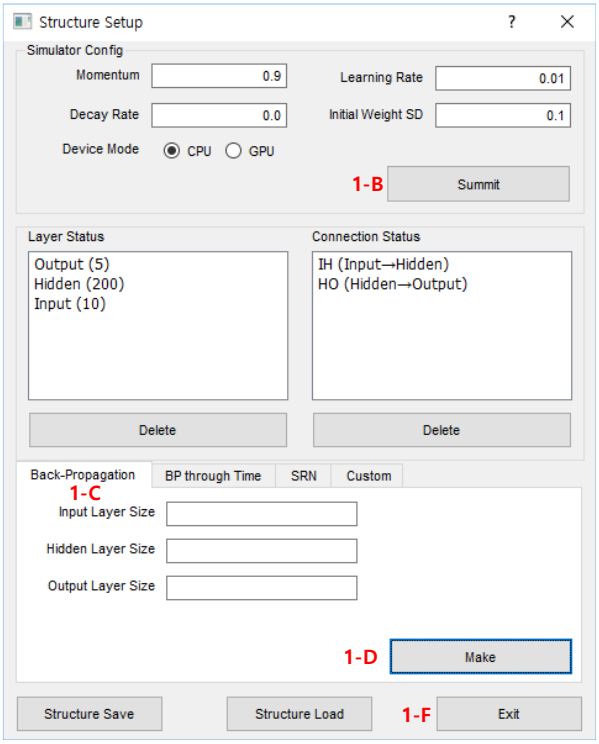

Figure 2. The structure setup in BP algorithm using

2. Pattern Setup
   A. Click ‘Pattern Setup’ button
   B. Click ‘Broswer…’ button and select pattern data. The ‘XOR5’ example used ‘XOR5.txt’. If you

want to use a different pattern, refer the chapter ‘2. Pattern Setup’ of Detail function.

C. Assign pack name. The ‘XOR5’ example used ‘XOR5’ as the pack name.
D. Click ‘Insert’ button
E. Click ‘Exit’ button

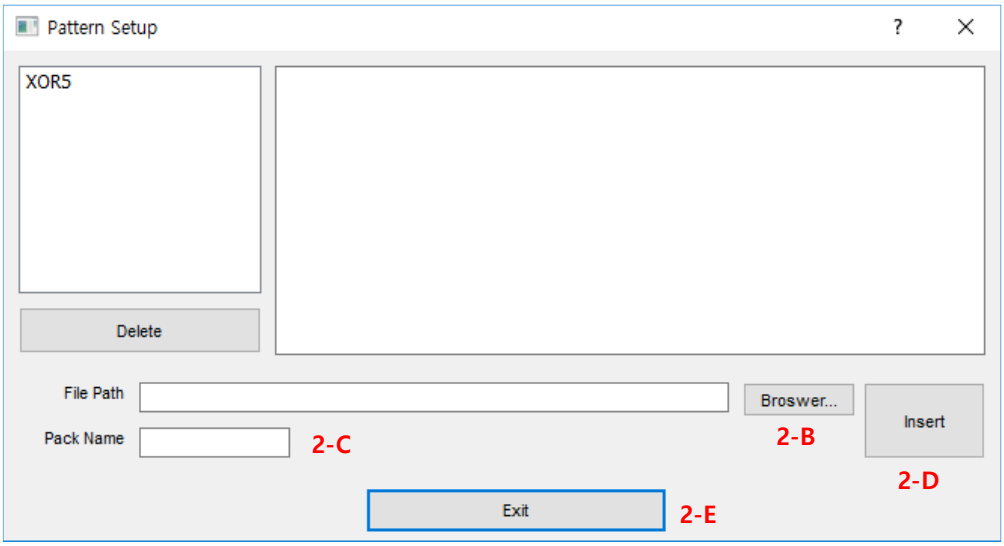

Figure 3. The pattern setup in BP algorithm using

3. Process Setup
   A. Click ‘Process Setup’ button
   B. Assign a training process name. The ‘XOR5’ example used ‘Training’.
   C. Click ‘Making’ button
   D. Click the ‘Back-propagation order’ tab and assign what layers are input, hidden, and output, respectively.
   E. Select the output layer’s activation calculation function. The ‘XOR5’ example used ‘Sigmoid’.
   F. Click ‘Training apply’ button
   G. Click ‘End’ button
   H. Assign a test process name. The ‘XOR5’ example used ‘Test’.
   I. Click ‘Making’ button
   J. Click the ‘Back-propagation order’ tab and assign what layers are input, hidden, and output, respectively.
   K. Select the output layer’s activation calculation function. The ‘XOR5’ example used ‘Sigmoid’.
   L. Click ‘Test apply’ button
   M. Click ‘End’ button
   N. Click ‘Exit’ button
   O. Click ‘Lock’ button which is located at the right of process setup button

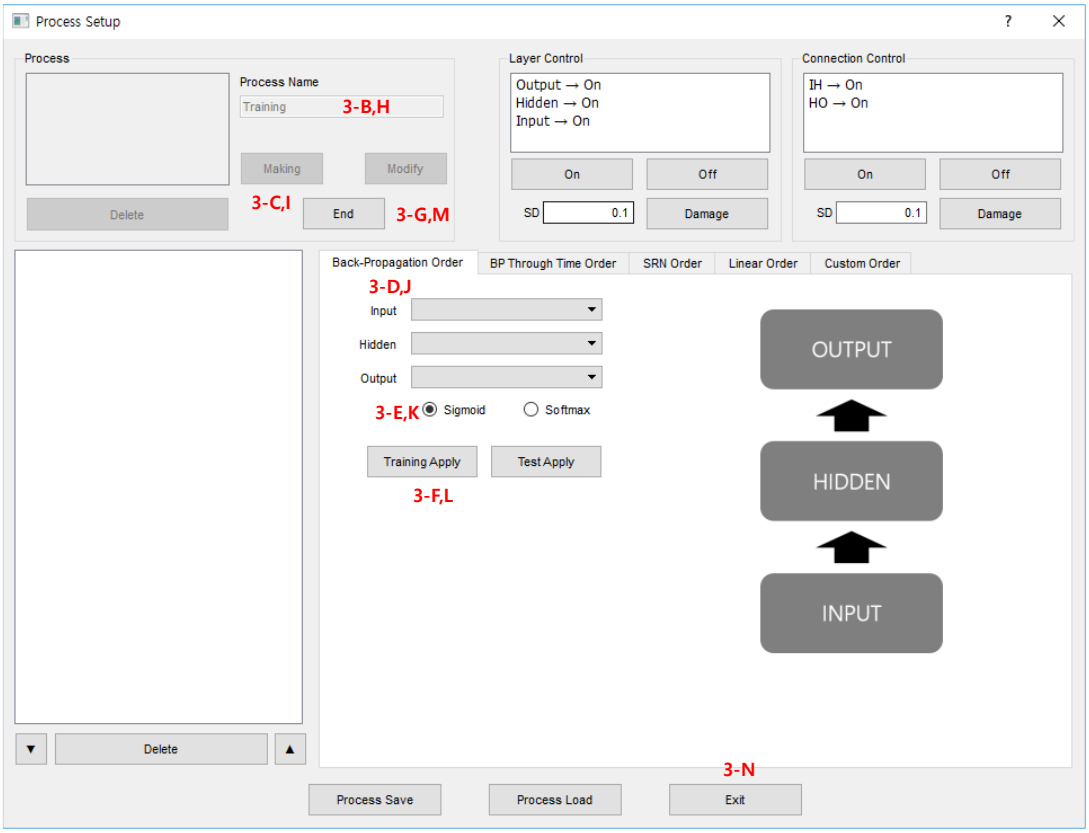

Figure 4. The process setup in BP algorithm using

4. Learning Setup
   A. Click ‘Learning Setup’ button.
   B. Assign a learning setup name. The ‘XOR5’ example used ‘XOR5’.
   C. Click ‘Making’ button of Learning setup panel.
   D. Assign training epoch, test timing, mini-batch size, and shuffling method. The ‘XOR5’ example used 10,000, 1,000, 1,000, and ‘random all’, respectively.
   E. Click ‘Making’ button of Training pattern matching panel.
   F. Select pattern pack and process you assigned.
   G. Click ‘Auto assign’ button. If there is no problem in pattern data, everything is assigned automatically. If there is any order at the ‘Order’ combo-box, you should to assign manually. Refer the chapter ‘4.2 Training pattern matching’.
   H. Click ‘End’ button of Training pattern matching panel.
   I. Click ‘Making’ button of Test pattern matching panel.
   J. Select pattern pack and process you assigned.
   K. Click ‘Auto assign’ button. If there is no problem in pattern data, everything is assigned automatically. If there is any order at the ‘Order’ combo-box, you should to assign manually. Refer the chapter ‘4.3 Test pattern matching’.
   L. Assign the extract data you want to get. Select extraction data type and click ‘Assign’ button
      i. Raw activation and semantic stress does not need to assign pattern.
      ii. Mean squared error and cross entropy need to assign pattern for the comparison. The ‘XOR5’ example used ‘Out’.
   M. Click ‘End’ button of Test pattern matching panel.
   N. Click ‘End’ button of Learning setup panel.
   O. Click ‘Exit’ button

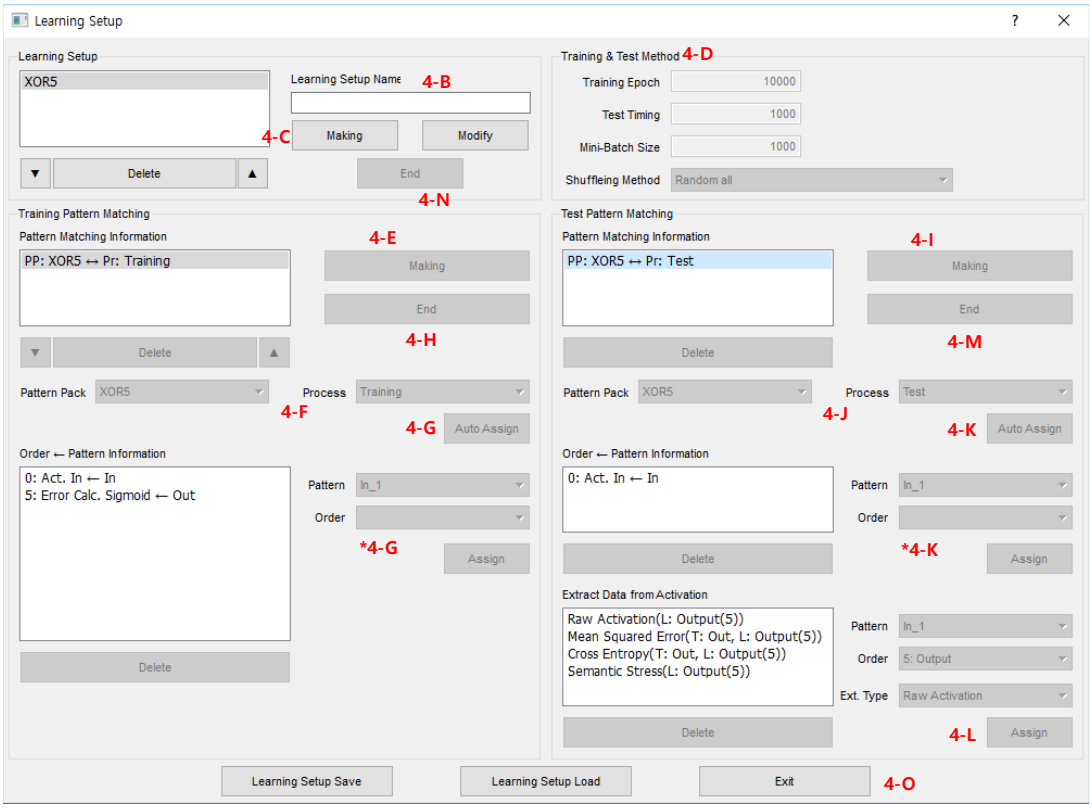

Figure 5. The learning setup in BP algorithm using

5. Learning
   A. Click ‘Learning’ button.
   B. Click ‘Start’ button.
   C. When training finished, the result will be exported at the H-Net folder automatically.
   D. Click ‘Exit’ button

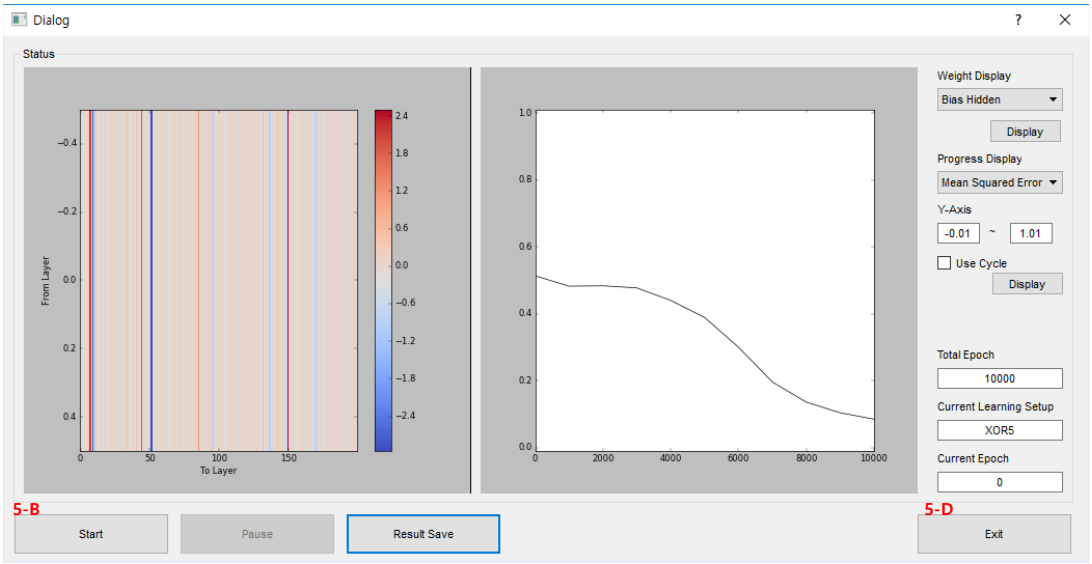

Figure 6. The learning window in BP algorithm using

6. Click ‘Exit’ button of main window to finish H-Net

2 Back-Propagation through Time Tutorial

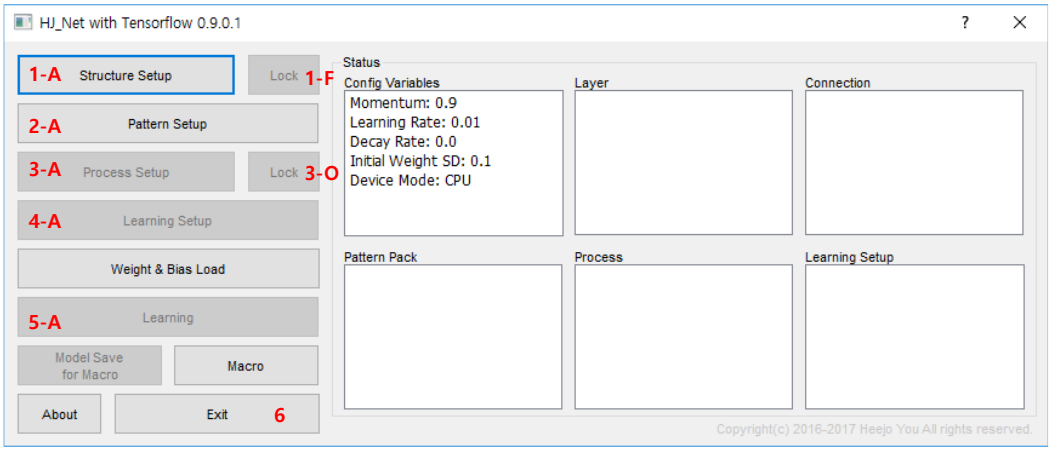

Figure 7. The main window in BPTT algorithm using

1. Structure Setup
   A. Click 'Structure Setup' button
   B. Assign the config variables and click the 'Submit' button.
   C. Click the 'BP through Time' tab and insert each layer's unit size and tick count. The 'BPTT' example used 5, 200, 5 units about the input, hidden, and output layers, respectively, and tick was assigned 5.
   D. Click 'Make' button
   E. Click 'Exit' button
   F. Click 'Lock' button which is located at the right of structure setup button

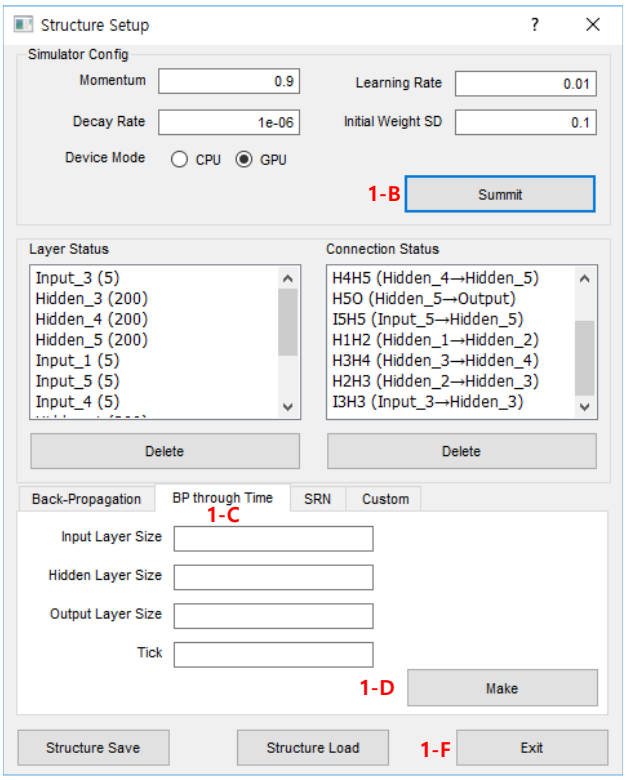

Figure 8. The structure setup in BPTT algorithm using

2. Pattern Setup
   A. Click 'Pattern Setup' button
   B. Click 'Broswer…' button and select pattern data. The 'BPTT' example used 'MoreLess.txt'. If you

want to use a different pattern, refer the chapter ‘2. Pattern Setup’ of Detail function.

C. Assign pack name. The ‘BPTT’ example used ‘MoreLess’ as the pack name.
D. Click ‘Insert’ button
E. Click ‘Exit’ button

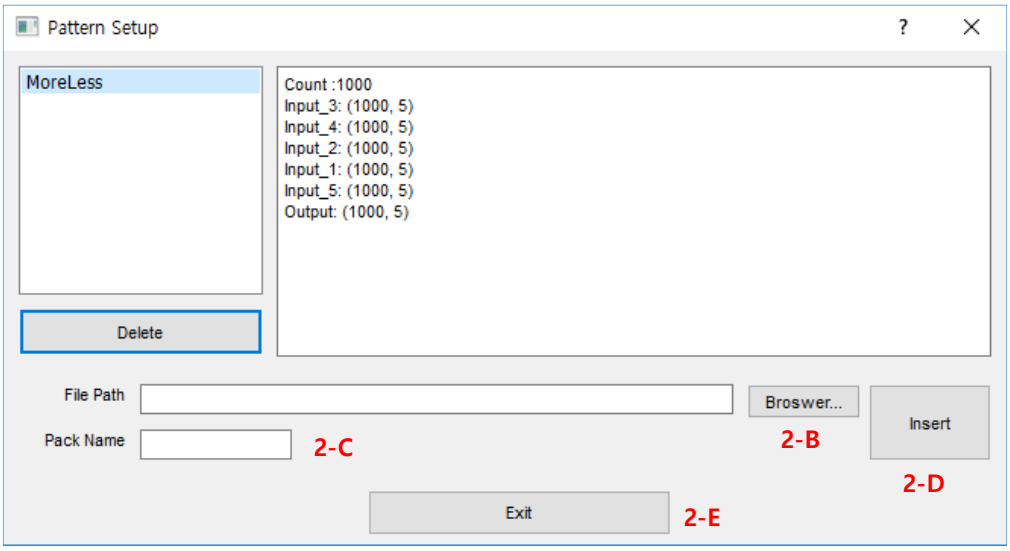

Figure 9. The pattern setup in BPTT algorithm using

3. Process Setup
   A. Click ‘Process Setup’ button
   B. Assign a training process name. The ‘BPTT’ example used ‘Training’.
   C. Click ‘Making’ button
   D. Click the ‘BP through Time Order’ tab
      i. Insert “Input_” and “Hidden_” to input and hidden text-box, respectively.
      ii. Insert the number of ticks to tick text-box. The ‘BPTT’ example used 5.
      iii. Assign what is the output layer.
   E. Select the output layer’s activation calculation function. The ‘BPTT’ example used ‘Sigmoid’.
   F. Click ‘Training apply’ button
   G. Click ‘End’ button
   H. Assign a test process name. The ‘BPTT’ example used ‘Test’.
   I. Click ‘Making’ button
   J. Click the ‘BP through Time Order’ tab
      i. Insert “Input_” and “Hidden_” to input and hidden text-box, respectively.
      ii. Insert the number of ticks to tick text-box. The ‘BPTT’ example used 5.
      iii. Assign what is the output layer.
   K. Select the output layer’s activation calculation function. The ‘BPTT’ example used ‘Sigmoid’.
   L. Click ‘Test apply’ button
   M. Click ‘End’ button
   N. Click ‘Exit’ button
   O. Click ‘Lock’ button which is located at the right of process setup button

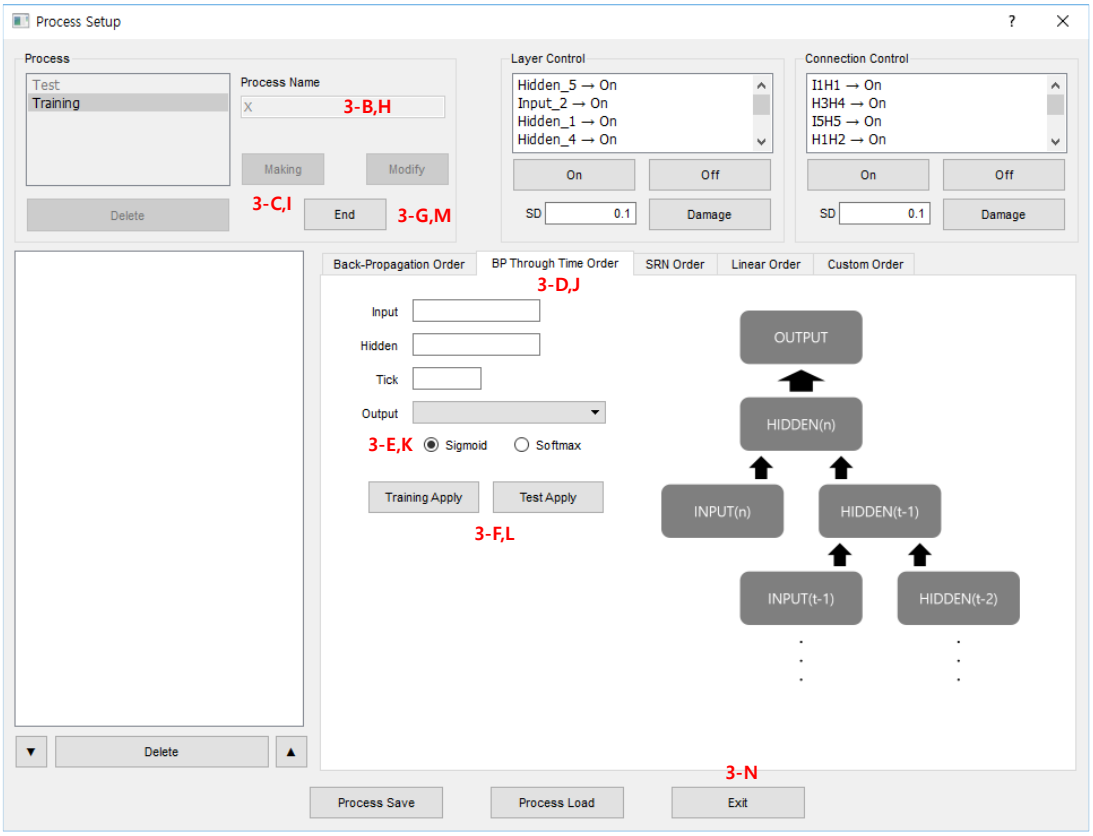

Figure 10. The process setup in BPTT algorithm using

4. Learning Setup
   - A. Click ‘Learning Setup’ button.
   - B. Assign a learning setup name. The ‘BPTT’ example used ‘BPTT’.
   - C. Click ‘Making’ button of Learning setup panel.
   - D. Assign training epoch, test timing, mini-batch size, and shuffling method. The ‘BPTT’ example used 5,000, 500, 200, and ‘random all’, respectively.
   - E. Click ‘Making’ button of Training pattern matching panel.
   - F. Select pattern pack and process you assigned.
   - G. Click ‘Auto assign’ button. If there is no problem in pattern data, everything is assigned automatically. If there is any order at the ‘Order’ combo-box, you should to assign manually. Refer the chapter ‘4.2 Training pattern matching’.
   - H. Click ‘End’ button of Training pattern matching panel.
   - I. Click ‘Making’ button of Test pattern matching panel.
   - J. Select pattern pack and process you assigned.
   - K. Click ‘Auto assign’ button. If there is no problem in pattern data, everything is assigned automatically. If there is any order at the ‘Order’ combo-box, you should to assign manually. Refer the chapter ‘4.3 Test pattern matching’.
   - L. Assign the extract data you want to get. Select extraction data type and click ‘Assign’ button
     - i. Raw activation and semantic stress does not need to assign pattern.
     - ii. Mean squared error and cross entropy need to assign pattern for the comparison. The ‘BPTT’ example used ‘Output’.
   - M. Click ‘End’ button of Test pattern matching panel.
   - N. Click ‘End’ button of Learning setup panel.
   - O. Click ‘Exit’ button

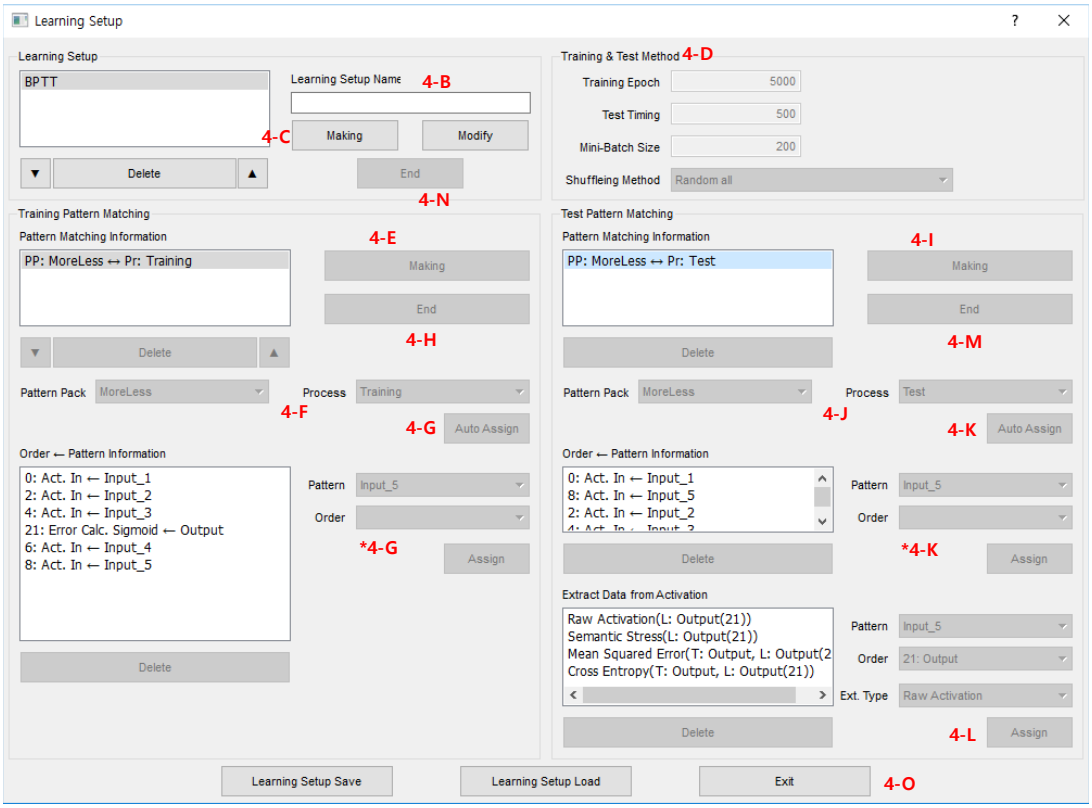

Figure 11. The learning setup in BPTT algorithm using

5. Learning
   A. Click ‘Learning’ button.
   B. Click ‘Start’ button.
   C. When training finished, the result will be exported at the H-Net folder automatically.
   D. Click ‘Exit’ button

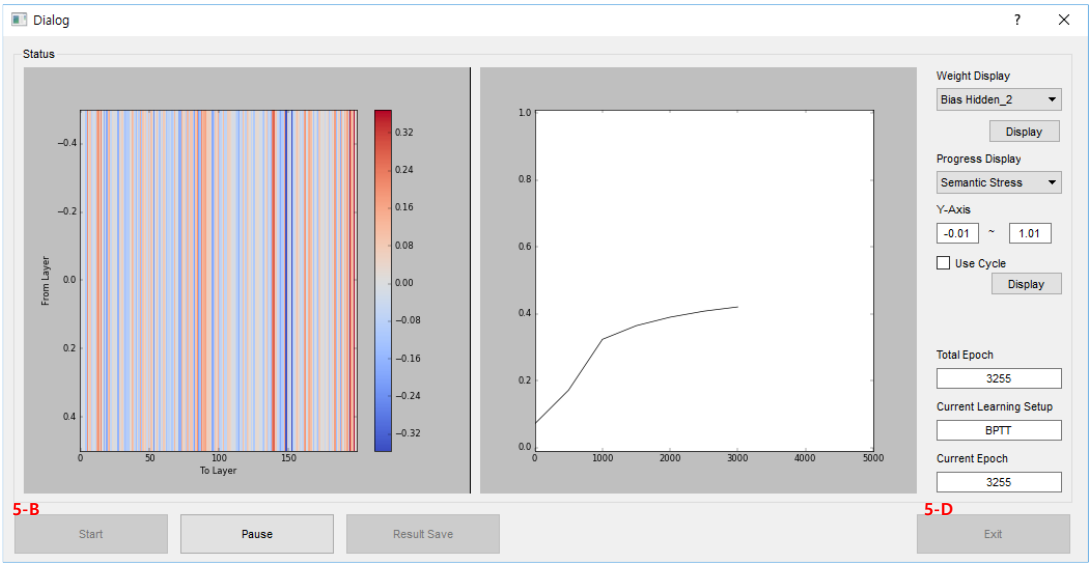

Figure 12. The learning window in BPTT algorithm using

6. Click ‘Exit’ button of main window to finish H-Net

3 Simple Recurrent Network Tutorial

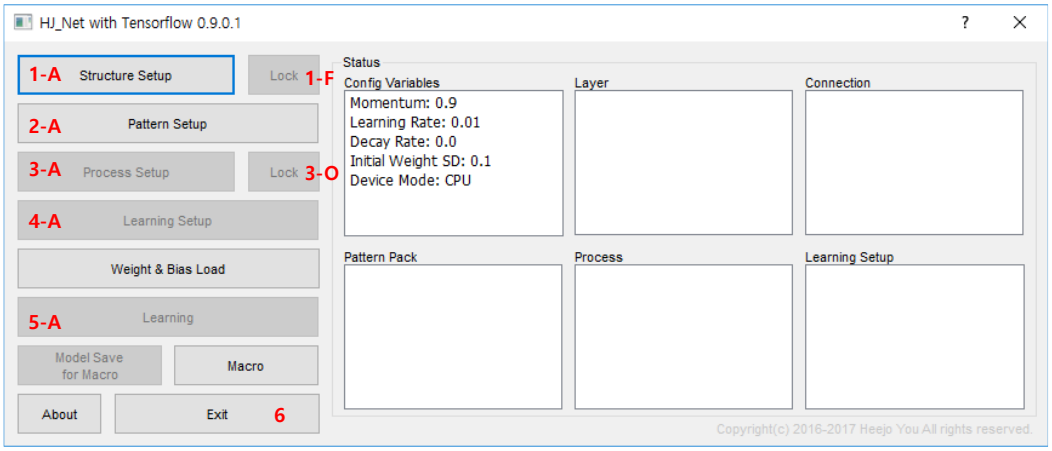

Figure 13. The main window in SRN algorithm using

1. Structure Setup
   A. Click ‘Structure Setup’ button
   B. Assign the config variables and click the ‘Submit’ button.
   C. Click the ‘SRN’ tab and insert each layer’s unit size. The ‘SRN’ example used 5, 200, 5 units about the input, hidden, and output layers, respectively.
   D. Click ‘Make’ button
   E. Click ‘Exit’ button
   F. Click ‘Lock’ button which is located at the right of structure setup button

Figure 14. The structure setup in SRN algorithm using

2. Pattern Setup
   A. Click ‘Pattern Setup’ button
   B. Click ‘Broswer…’ button and select pattern data. The ‘SRN’ example used ‘MoreLess.txt’. If you want to use a different pattern, refer the chapter ‘2. Pattern Setup’ of Detail function.
   C. Assign pack name. The ‘SRN’ example used ‘MoreLess’ as the pack name.

D. Click ‘Insert’ button
E. Click ‘Exit’ button

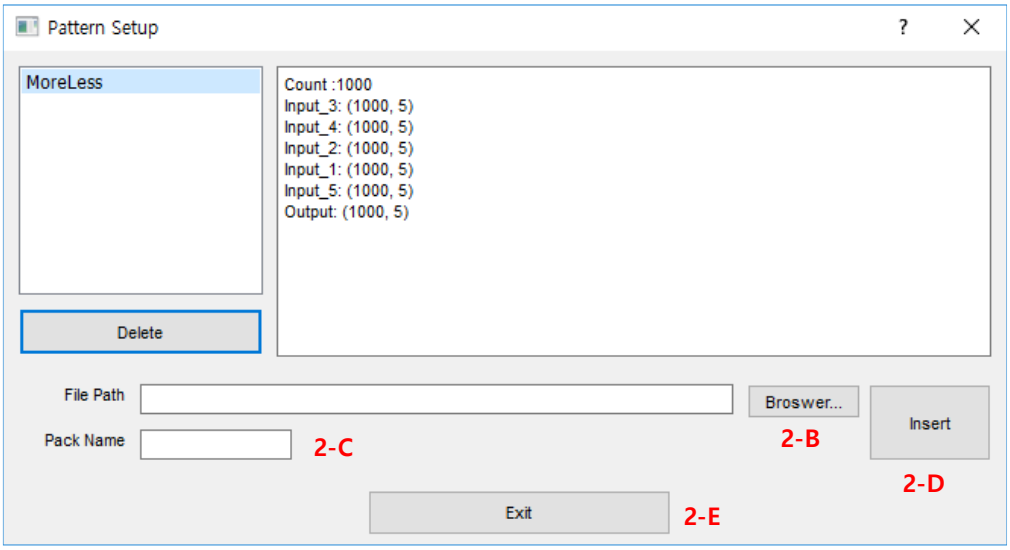

Figure 15. The pattern setup in SRN algorithm using

3. Process Setup
   A. Click ‘Process Setup’ button
   B. Assign a training process name. The ‘SRN’ example used ‘Training’.
   C. Click ‘Making’ button
   D. Click the ‘SRN Order’ tab
      i. Assign what layers are input, context, hidden, and output, respectively.
      ii. Insert max cycle value. The ‘SRN’ example used 5.
   E. Select the output layer’s activation calculation function. The ‘SRN’ example used ‘Sigmoid’.
   F. Click ‘Training apply’ button
   G. Click ‘End’ button
   H. Assign a test process name. The ‘SRN’ example used ‘Test’.
   I. Click ‘Making’ button
   J. Click the ‘SRN Order’ tab
      i. Assign what layers are input, context, hidden, and output, respectively.
      ii. Insert max cycle value. The ‘SRN’ example used 5.
   K. Select the output layer’s activation calculation function. The ‘SRN’ example used ‘Sigmoid’.
   L. Click ‘Test apply’ button
   M. Click ‘End’ button
   N. Click ‘Exit’ button
   O. Click ‘Lock’ button which is located at the right of process setup button

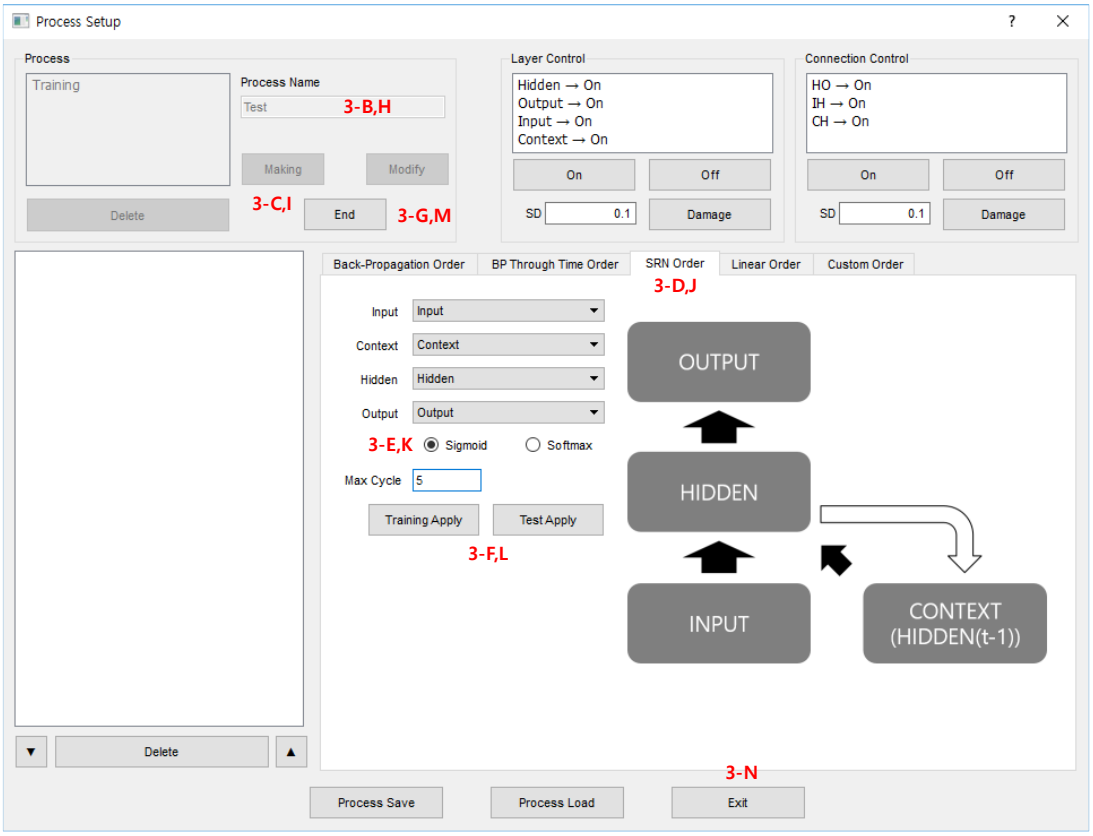

Figure 16. The process setup in SRN algorithm using

4. Learning Setup
   - A. Click ‘Learning Setup’ button.
   - B. Assign a learning setup name. The ‘SRN’ example used ‘SRN’.
   - C. Click ‘Making’ button of Learning setup panel.
   - D. Assign training epoch, test timing, mini-batch size, and shuffling method. The ‘SRN’ example used 10,000, 2000, 1000, and ‘random all’, respectively.
   - E. Click ‘Making’ button of Training pattern matching panel.
   - F. Select pattern pack and process you assigned.
   - G. Click ‘Auto assign’ button. If there is no problem in pattern data, everything is assigned automatically. If there is any order at the ‘Order’ combo-box, you should to assign manually. Refer the chapter ‘4.2 Training pattern matching’.
   - H. Click ‘End’ button of Training pattern matching panel.
   - I. Click ‘Making’ button of Test pattern matching panel.
   - J. Select pattern pack and process you assigned.
   - K. Click ‘Auto assign’ button. If there is no problem in pattern data, everything is assigned automatically. If there is any order at the ‘Order’ combo-box, you should to assign manually. Refer the chapter ‘4.3 Test pattern matching’.
   - L. Assign the extract data you want to get. Select extraction data type and click ‘Assign’ button.
     - i. Raw activation and semantic stress does not need to assign pattern.
     - ii. Mean squared error and cross entropy need to assign pattern for the comparison. The ‘SRN’ example used ‘Output’.

     Notice: If user want to see the progress of every cycle, every cycle’s extraction type which you want to see should be assigned.
   - M. Click ‘End’ button of Test pattern matching panel.
   - N. Click ‘End’ button of Learning setup panel.
   - O. Click ‘Exit’ button

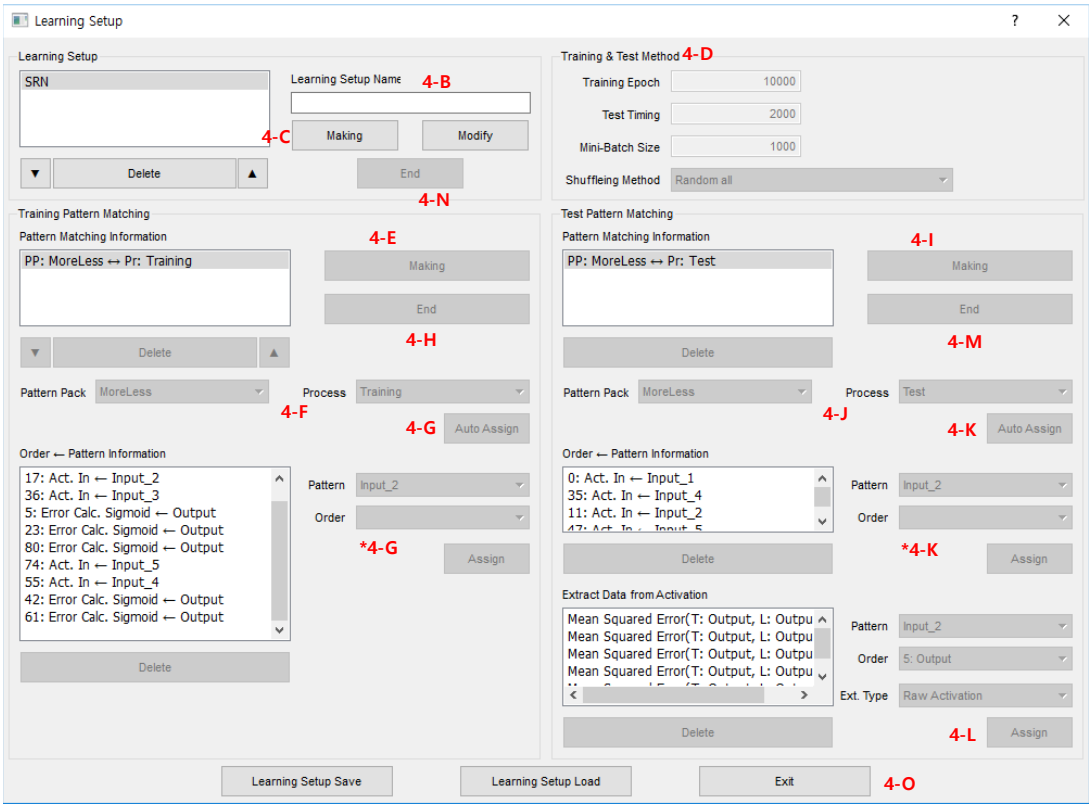

Figure 17. The learning setup in SRN algorithm using

5. Learning
   A. Click ‘Learning’ button.
   B. Click ‘Start’ button.
   C. When training finished, the result will be exported at the H-Net folder automatically.
   D. Click ‘Exit’ button

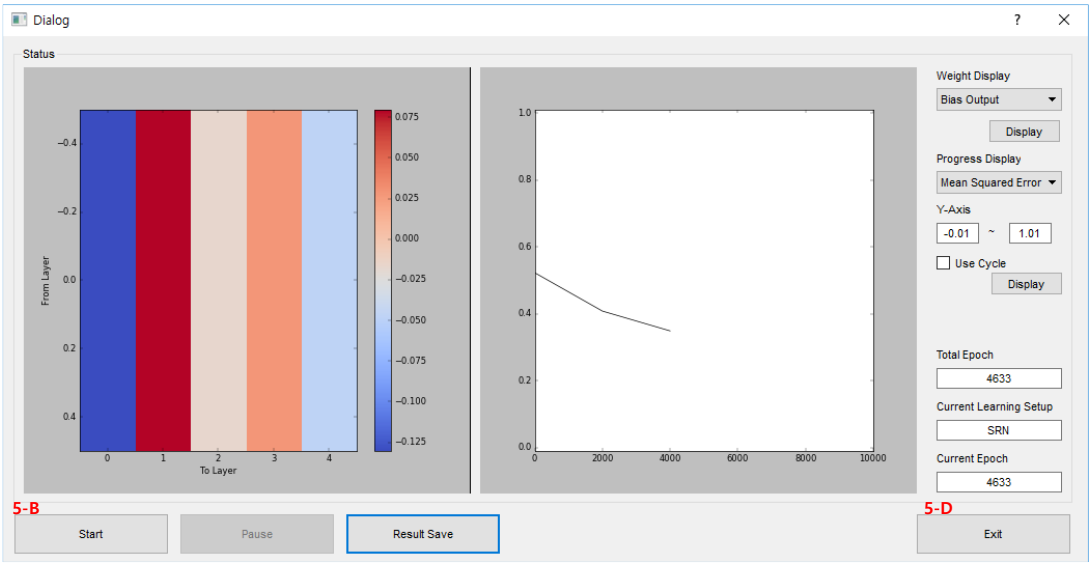

Figure 18. The learning window in SRN algorithm using

6. Click ‘Exit’ button of main window to finish H-Net

Detail Functions

1 Structure Setup

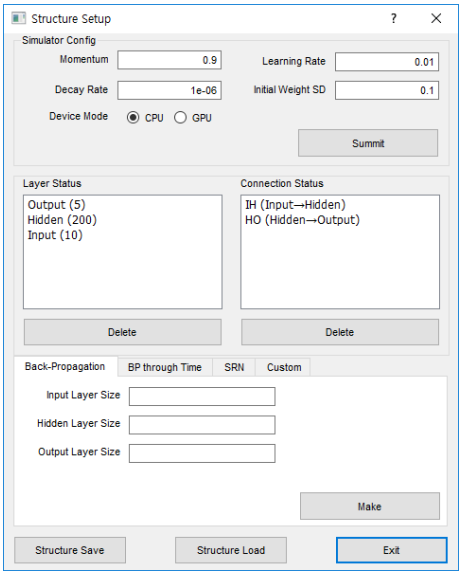

Figure 19. The architecture setup window.

Figure 19 is a window of architecture setup of program. In architecture setup, user set the initial variables and model structure. Initial variables decide the statuses of model. Therefore, when user set the architecture primarily, initial values should to be set before layer and connection setup.

1.1 Initial values

Momentum is related the sigmoid function. In sigmoid function, momentum affect the slope of graph. The larger momentum makes steeper slope of function.

Learning rate means how many weights are changed at one training. If the learning rate is too big, model's performance becomes fool. On the other hand, when the learning rate is too small, training speed becomes very slow. Users have to search a suitable learning rate.

Decay rate is used at the weight change process in training. In weight change process, weights are multiplied by $(1 - \text{Decay rate})$.

Initial weight SD decide the initial status of connections' weight. Every weight and bias is assigned some randomize values by $\text{Normal}(0, \text{SD})$. *I maybe modify the initial weight equation.

Device mode decides what processor will be used. If user's system does not support the GPU mode, this is fixed to the CPU mode. **For GPU mode, the system need a NVIDIA graphic card, CUDA, and Tensorflow GPU version.**

One more important is that user can modify the initial variable after setup. When Users click the submit button, the initial variables are modified.

At the below of structure window, there are 4 tabs. Three tabs help to make a model structure automatically, and one is custom menu.

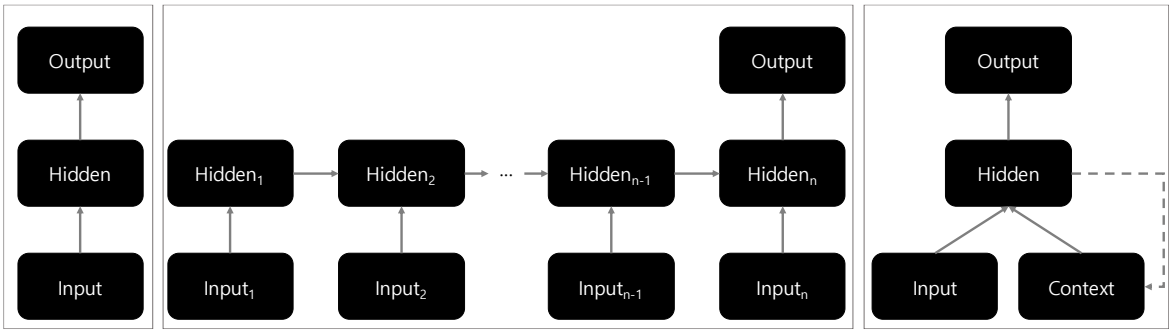

Figure 20. The basic structure which H-Net can support. Left) The Back-Propagation structure. Middle) The Back-Propagation through Time structure. Right) The Simple Recurrent Network structure

1.2 Back-Propagation

This function makes 3 layers and 2 connections. The name of layer automatically assigned Input, Hidden, and Output, and connection names become IH, HO (see the left of Figure 20). User can decide the size of the layers. When user click 'Make' button, the structure will be inserted.

1.3 Back-Propagation through Time

BPTT making requires one more variable, tick. When user makes BPTT model, the Input and Hidden layer will be made as many as the tick which user inserted (see the middle of Figure 20). The layer name become 'Name_tick' like 'Input_1' and 'Hidden_3'.

1.4 Simple Recurrent Network

SRN making function makes 4 layers and 3 connections. The basic structure is same to the Back-Propagation, but there is one more layer, Context. This layer size is same to the Hidden, and is connected to the Hidden inserted (see the right of Figure 20).

1.5 Custom

In fourth tab, user can customize the model by making the layers and connections.

1.1.1 Layer

Layer is a location which receive and send the activation. In layer section, user set how many and what layers make.

Layer name does not affect any model performance in both of training and test. This is just index for user and program. Any name is OK, but **it must not be overlapped to other layer names**.

Unit is the size of layer. Usual layer's size is related the representation of model. On the other hand, the hidden layer's size is related the computing power of model. Both of cases are user's discretion. However, **when the unit size is too big, that makes some problem about the memory**.

1.1.2 Connection

A connection links two layers. In connection section, user set what layer send activation to some layers. In this tool, connections are always one-way, not bi-direction.

Like the layer name, connection name is also just index for user and program. This does not affect any model performance, and **it must not be overlapped to other connection name**.

'From' and 'To' drop-box shows the layers which are made in layer section. The selected layer of "From" drop-box becomes the layer which activation sends. On the other hand, the selected layer of "To" drop-box becomes the layer which activation receives.

One layer can send the activation to several layers, and can also receive the activation from several layers. However, if there is already same connection which both of 'from' and 'to' is same, it is impossible to make the connection.

1.6 Save & Load

There are the functions of save and load. When user save the architecture, the save file storage the initial values, layer information, and connection information. The saved file has "HNet_Structure" extension.

When program load the 'HNet_Structure' file, current structure information will be swept. User should be careful.

1.7 Structure Lock

After making structure, **the structure should be locked in main window**. This is to initialize the weight and bias of model, and to maintain the consistency while user make processes.

2 Pattern Setup

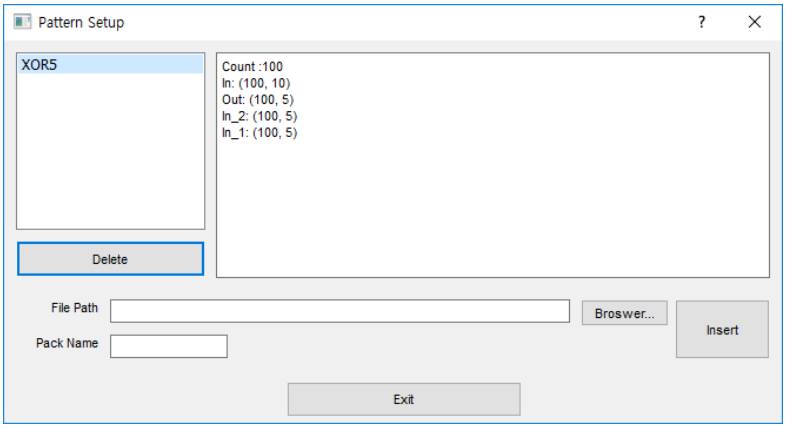

Figure 21. The pattern setup window.

In this window, user can insert the pattern pack which model use in training and test.

2.1 Pattern File

Currently, H-Net uses the extension 'txt' as pattern pack file. Although there can be multiple patterns in a file, the file content is required a strict form to prevent to confuse.

Fundamentally, H-Net demands several columns in a pattern file: Name, Probability, Cycle, and patterns. Table 1 shows an example about file structure. **In text file, these column is separated by a tap.**

In Table 1, first line is the column name. At first, the values of column 'Name' do not affect the model performance, but these become the index of exported result. Thus, I recommend avoiding to overlap.

The column 'Probability' is for **the frequency**. In training, at each epoch, model throw a dice between 0 to 1 about each pattern. And if the dice is lower than pattern's probability, the pattern will be trained at the epoch. Thus, the probability is higher, the pattern will be trained frequently. User has to assign this value between 0 to 1. When the value is lower than 0, the pattern is always not trained, and higher than 1, the pattern is always trained. **One notice is that this probability does not affect to the test.** In test, model uses all pattern because test does not affect the model's status.

The column 'Cycle' is for SRN. When user does not want to use cycle in his model, **it is fine that the all values of this column is 1 or more, not 0**. In SRN algorithm, this value will control the recurrent number.

The forth and more columns are the patterns. Although all pattern values are 0 or 1 in the example of Table 1, **it can be use all values which are between 0 and 1**. We think that the patterns are used as input or output in training and test process, so the amount of number of a pattern (separated by space, not tap) has to be same to the unit size of the layer which user wants to match. H-Net could not check the amount correctness in this stage because model cannot inference what pattern and layer have to be matched.

Table 1. An example of pattern file

Name	Probability	Cycle	Input_1	…	Output_2	…
school	0.765	10	1 1 0 1 0 1 1 …	…	1 1 1 0 0 0 0 …	…
hierarchy	0.043	21	0 0 0 1 1 0 0 …	…	0 1 1 0 1 0 0 …	…
have	0.914	33	1 0 0 0 0 0 0 …	…	1 1 0 0 1 0 0 …	…

2.2 Pack Name

Pack name does not affect any model performance in both of training and test. This is just index for user and program. Any name is OK, but it must not be overlapped to other pack names.

After pattern pack inserted, the pack name displayed at the listbox of left-upside. When user clicks the pack name, the basic information of the pattern pack will be displayed like Figure 21.

3 Process Setup

Process is a plan of the training and test flow. In process setup, user can set up some usual algorithms and customizing.

3.1 Layer & Connection Control

This function is for the layer and connection's status. Currently, there are three statuses: On, Off, and Damaged. The basic status is 'On.' If user change the status to 'Off,' the layer or connection do not work in the process. For layer, the unit activation of the layer become 0 in all calculations which use the layer activation like activation sending to another layer. On the other hand, for connection, all weights of the connection become 0.

When layer or connection status is the damage status, the Gaussian noise is added to the unit activation or weight. User can set up the standard deviation of Gaussian noise to control the degree of the lesion. But layer's final value is clipped between 0 to 1.

Importantly, this does not mean the real weights are changed. The real value is saved. These status changes only affect the process.

3.2 Back-Propagation Order

The back-propagation order menu is a simple method for using the BP algorithm. If user set up the structure which is for the BP algorithm, this function makes the process for BP algorithm.

To use this menu, model's architecture has to be constructed like right figure of program (see Figure 22). If user uses the BP structure making at the structure setup window, the structure will be compatible.

Below the output layer drop-box, **the sigmoid and softmax radio-buttons are related with the activation calculation of output layer**. This is only output layer, not hidden. **Hidden layer's activation always uses the sigmoid function in this method**.

After user select input, hidden, output layers, when user clicks 'Training Apply' or 'Test Apply' buttons, the all orders of process which is for BP algorithm will be automatically inserted. Because H-Net requires the separate processes about training and test, user should make two different processes.

And then, user can finish the process making by the clicking the 'End' button of upside of window.

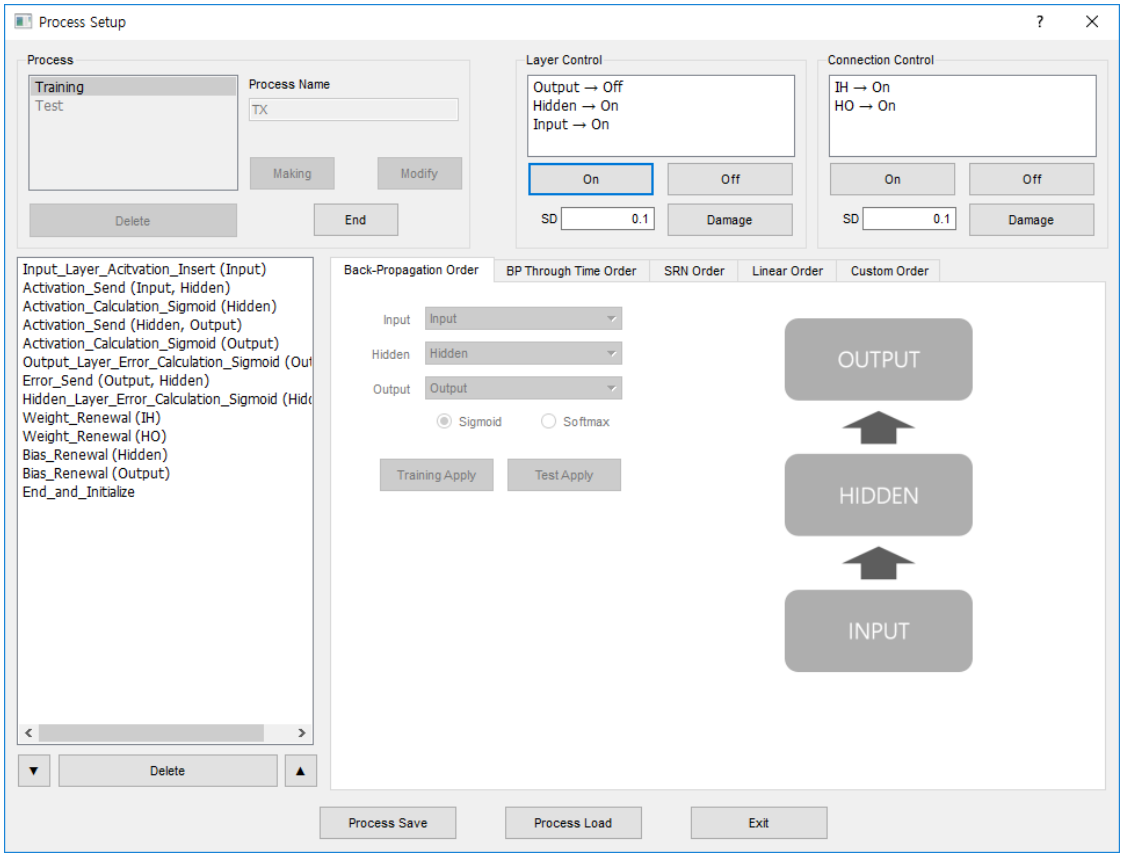

Figure 22. The BP process setup example. The order list shows the training process.

3.3 BP Through Time Order

In the BPTT order menu, user have to insert the prefix of input and hidden layers, and select the output layer. If user used the BPTT structure making function at the structure setup, the prefixes of input and hidden are 'Input_' and 'Hidden_', respectively (see Figure 23).

Like the BP Order, **the sigmoid and softmax radio-buttons are related with the activation calculation of output layer**. This is only output layer, not hidden. **Hidden layer's activation always uses the sigmoid function in this method**.

After user inserts the information about the model structure, when user clicks 'Training Apply' or 'Test Apply' buttons, the all orders of process which is for BPTT algorithm will be automatically inserted.

And then, user can finish the process making by the clicking the 'End' button of upside of window.

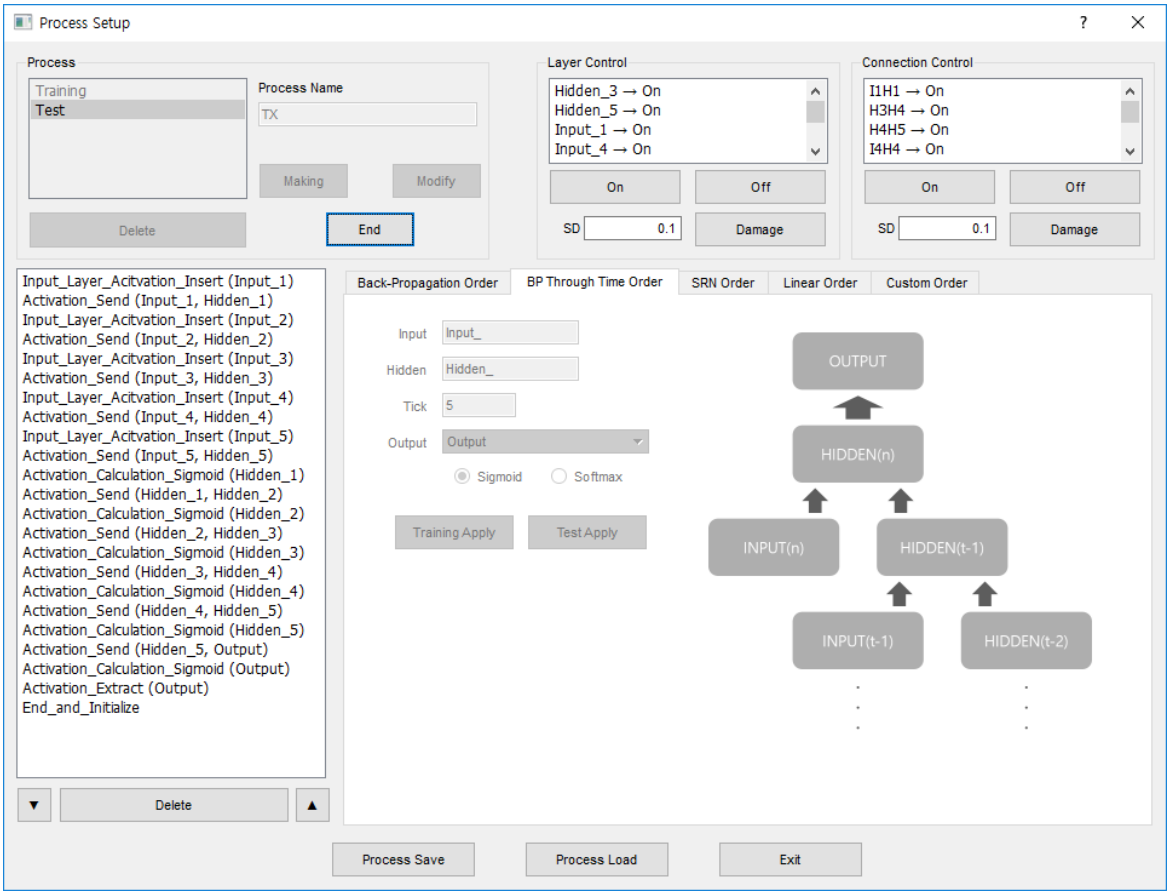

Figure 23. The BP Through Time process setup example. The order list shows the test process.

3.4 SRN Order

Like BP Order, user should to select the input, hidden, output layers. In addition, SRN Order requires two information more, context and max cycle. **The size of context layer have to be same to the hidden layer because the activation of hidden is copied to context layer at each cycle**. Max cycle decides the recurrent length of SRN algorithm.

To use this menu, model's architecture has to be constructed like right figure of program (see Figure 24). If user uses the SRN structure making at the structure setup window, the structure will be compatible.

Like the BP Order, **the sigmoid and softmax radio-buttons are related with the activation calculation of output layer**. This is only output layer, not hidden. **Hidden layer's activation always uses the sigmoid function in this method**.

After user inserts the information about the model structure, when user clicks 'Training Apply' or 'Test Apply' buttons, the all orders of process which is for BPTT algorithm will be automatically inserted.

And then, user can finish the process making by the clicking the 'End' button of upside of window.

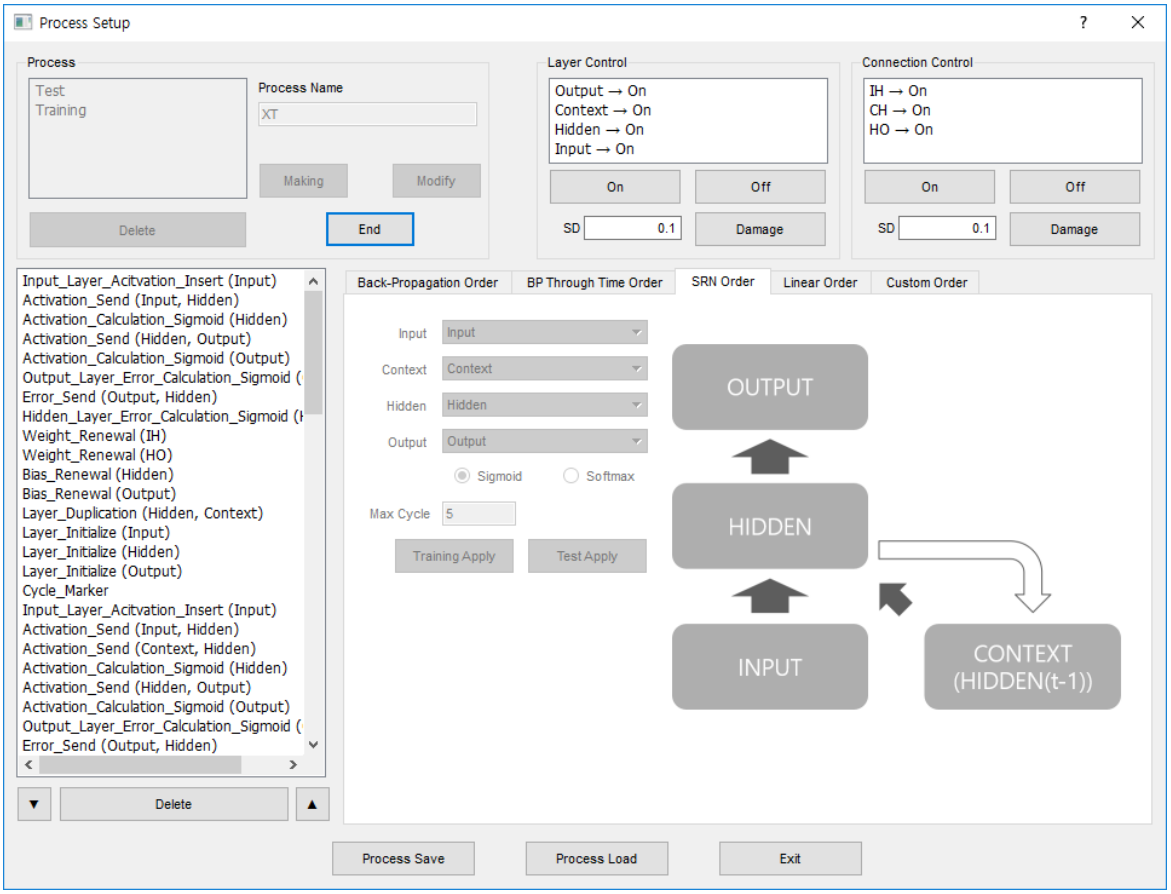

Figure 24. The SRN process setup example. The order list shows the training process.

3.5 Linear Order

In linear order menu, user can set the linear forward and backward processes. This menu is similar to the back-propagation order menu, but this can be used in more complex architectures like multiple hidden layers or separated input layers.

In forward order, each layer calculates and sends the activation. The layers, which inserted in the list-box, have to sequentially connected like back-propagation order menu.

On the other hand, in backward order, the errors of layers are calculated and weights and biases are renewed. **Importantly, the layers, which inserted in the list-box, have to reversely connected**.

The radio-buttons related with the types of first and last layer. The inserted orders are different by these types.

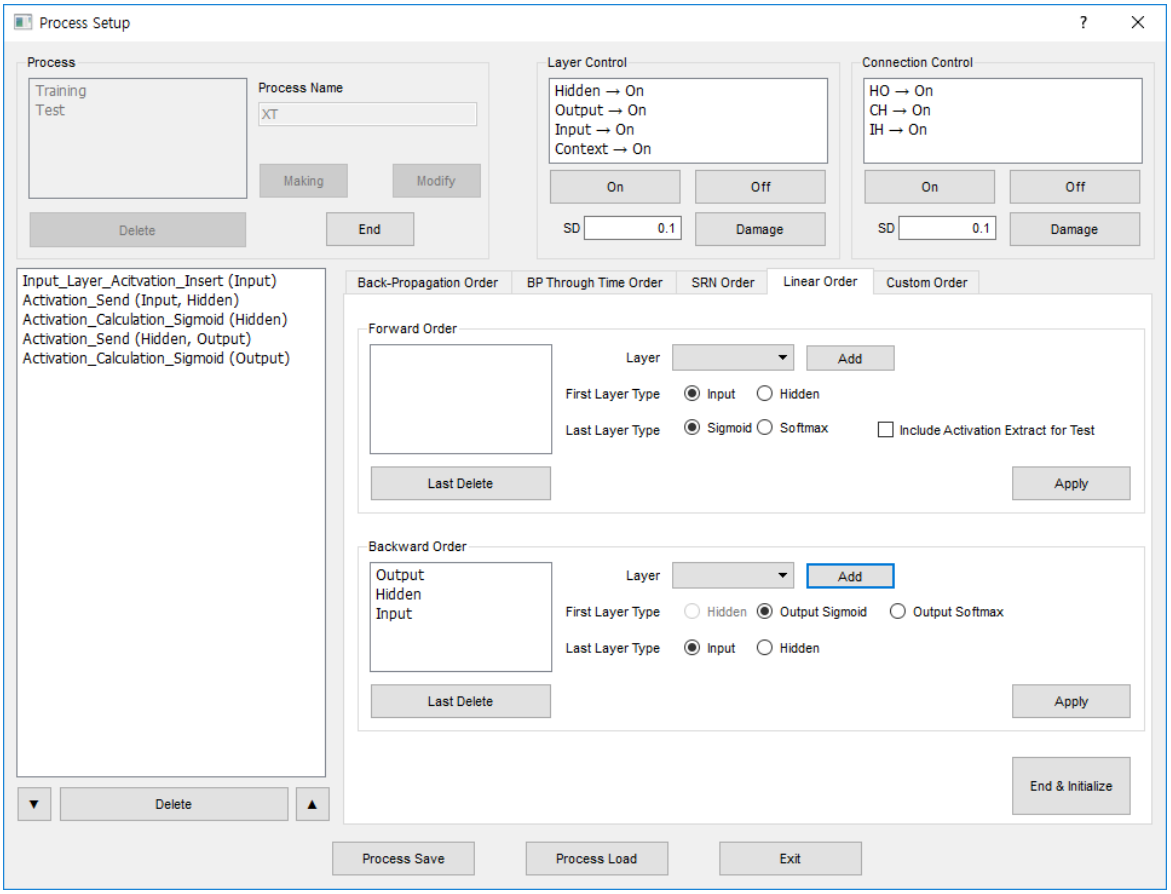

Figure 25. The linear order process setup example.

3.6 Custom Order

If user want to make a detail flow, the process can be customized. Table 2 shows how each order operates in the process flow.

A method for using this custom order menu is ‘modifying’. For example, user is designing the original BP model which has two input layers, he can modify the flow of original BP algorithm flow after using the BP Order menu. Of course, user also can create all process himself.

Table 2. order operation in the process flow

Order Name	Function
Activation Insert	Inserting external pattern to selected layer as the activation.
Activation Calculate (Sigmoid)	Calculating the inserted storage values from other layer to current layer's activation by sigmoid function.
Activation Calculate (Softmax)	Calculating the inserted storage values from other layer to current layer's activation by softmax function. However, we do not recommend to use this order in hidden layers. The using to hidden layer maybe decrease the performance of model.
Activation Calculate (ReLU)	Calculating the inserted storage values from other layer to current layer's activation by ReLU function.
Activation Send	Inserting the multiplication value of layer 1's activation and weight to layer 2's storage (**Notice: storage is not activation**).
Activation Extract	This order is a pointer for activation extraction in the test. This activation is used to make the result.
Output Layer Error Calc. (Sigmoid)	Calculate the layer's error by comparison between external pattern and layer's activation. This order is valid when activation was calculated by sigmoid function.
Output Layer Error Calc. (Softmax)	Calculate the layer's error by comparison between external pattern and layer's activation. This order is valid when activation was calculated by softmax function.
Hidden Layer Error Calc. (Sigmoid)	Calculate the layer's error by connected layer's error and weight. This order is valid when activation was calculated by sigmoid function.
Hidden Layer Error Calc. (ReLU)	Calculate the layer's error by connected layer's error and weight. This order is valid when activation was calculated by ReLU function.
Error Send	Inserting the multiplication value of layer 1's error and weight to layer 2's error storage. (**Notice: error storage is not error value. Error is calculated by Error Calc. order**).
Layer Duplicate	Copying the first layer's all information to the second layer. Two layers have to be same unit size.
Layer Initialize	Initializing selected layer. All storage, activation, and error become 0.
Cycle Marker	Adding 1 to cycle. The pattern's activation which is lower than current cycle is filtered.
Bias Renewal	Renewing selected layer's bias. The error calculation has to precede this order.
Weight Renewal	Renewing selected connection. The error calculation has to precede this order.
End & Initialize	There is no function. This is a marker to express end. However, all processes have to assign this order at the end.
Transposed Connection Duplication	Copying the first connection's transposed weights to the second connection. Two connections have to be transposed size (**the size of 'from layer' of connection 1 has to be same the size of 'to layer' of connection 2, and the size of 'to layer' of connection 1 has to be same the size of 'from layer' of connection 2**).
Connection Duplication	Copying the first connection's weights to the second connection. Two connections have to be same size (**the size of 'from layer' of connection 1 has to be same the size of 'from layer' of connection 2, and the size of 'to layer' of connection 1 has to be same the size of 'to layer' of connection 2**).
Uniform Rand. Act. Insert	Inserting randomized pattern to selected layer as the activation. The uniform random is used to make the value. The range of uniform random is 0 to the maximum value which user inserts. But final value is clipped between 0 to 1.
Normal Rand. Act. Insert	Inserting randomized pattern to selected layer as the activation. The **absolute value** of normal random is used to make the value. The mean and SD are 0.0 and the value which user inserts. But final value is clipped between 0 to 1.
Bias Equalization	All designated layer's biases are substituted to the averaged value of the biases. **The unit size of all designated layers should be same**.
Weight Equalization	All designated connection's weights are substituted to the averaged value of the weights. **The shape of all designated connections should be same.**

3.7 Save & Load

Like the structure setup, there are the functions of save and load. When user save the process, the save file storage all processes which include layer and connection status and order. The saved file has "HNet_Process" extension. When program load the 'HNet_Process' file, user should not use the other model's file. Although we make independent files about each setting, 'HNet_Process' files depend on the structure. **If the selected HNet_Process file is not compatible with current model structure, the load will be failed**.

In addition, like the structure setup, **when program load the HNet_Process file, all current process information is swept**. User has to be careful.

3.8 Structure Lock

After making processes, **the processes should be locked in main window**. This is to convert the processes to the tensor, and to maintain the consistency while user make the learning setup.

4 Learning Setup

Learning setup is the setting about learning and test. Before this stage, the structure and processes should be locked, and pattern packs must be inserted. In this stage, user can set up how model use the pattern and processes (see Figure 26).

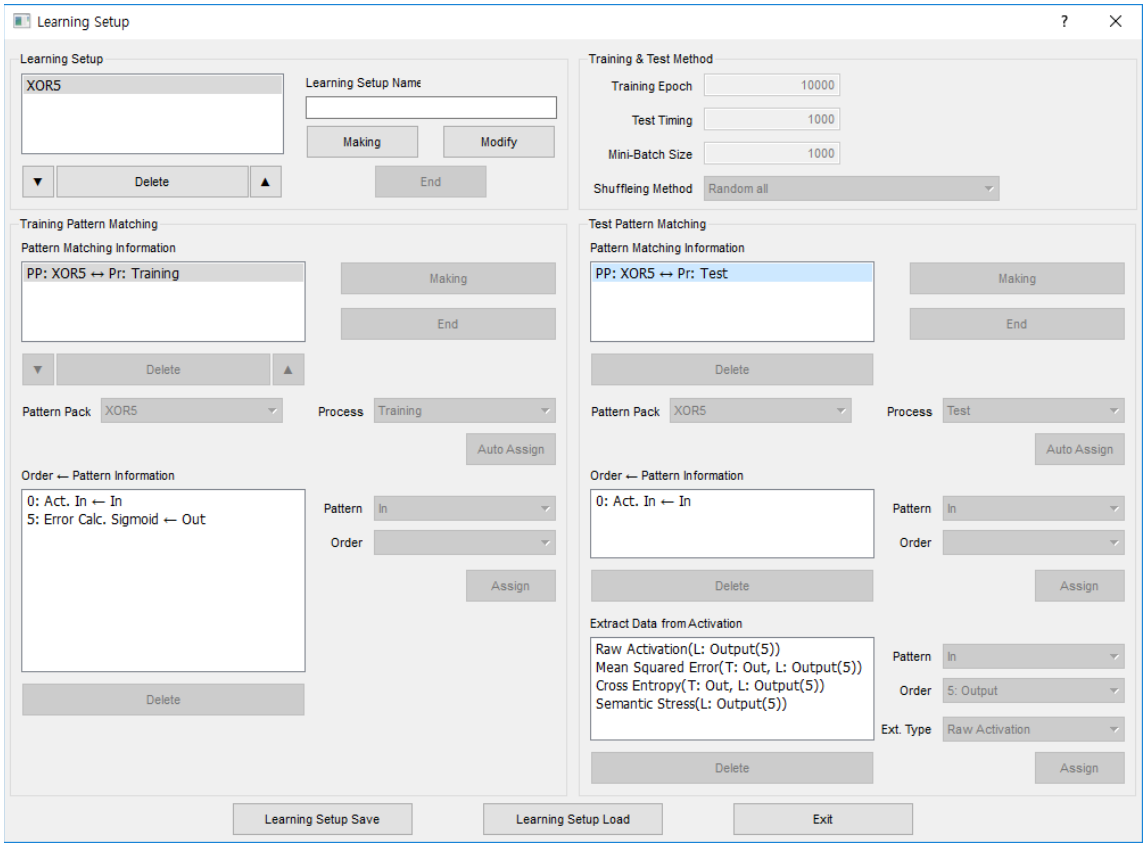

Figure 26. The learning setup examples.

4.1 Training & Test method

4.1.1 Training epoch

This decides how long the learning will progress.

4.1.2 Test timing

This decides when the test will do. If the size of test timing too small, model should spend too many resource to save the result. And I recommend the prime factor of training epoch as the test timing to know the result after all training.

4.1.3 Mini-batch size

This decides how many pattern will be trained at one time of each epoch. If the size of mini-batch is same or bigger than the amount of all patterns, it becomes full batch (all patterns are trained at one time of each epoch).

4.1.4 Shuffle method

In a learning set, a training information also can have multi pair of pattern pack and process. This means that several training flows can be used in a training. This function is to cope the situation which user premises that model uses several separate training flows in an epoch. H-Net provides some randomize methods about this function. **One notice is that random process is restricted at the mini-batch. Because it is impossible that different processes are trained at a time, pattern which is in a batch become a package** (See Table 3). If user does not want this, user must to assign 1 to the mini-batch size.

Table 3. the difference among training randomize method.

Name	Method
Random all	Both processes and patterns order will be randomized. In other words, all processes' all learning patterns will be randomized at each epoch. Ex) $Process_1$&$Pattern_{4,1,8,...}$→ $Process_3$&$Pattern_{2,10,6,...}$ → $Process_2$&$Pattern_{3,62,50,...}$ → …
Random in Pattern Pack	In all epochs, process will be always sequential, but the pattern of each process will be randomized. Ex) $Process_1$&$Pattern_{4,6,2,...}$ → $Process_1$&$Pattern_{2,42,1,...}$ → … →$Process_2$&$Pattern_{3,7,24,...}$ → $Process_2$&$Pattern_{16,5,51,...}$ →…
Sequential All	Both processes and patterns will be sequential. Ex) $Process_1$&$Pattern_{1,2,3}$ → $Process_1$&$Pattern_{4,5,6}$ → … →$Process_2$&$Pattern_{24,25,26}$ →$Process_2$&$Pattern_{27,28,29}$→…
Sequential in Pattern Pack	Only process sequence is randomized. Ex) $Process_3$&$Pattern_{1,2,3,...}$ → $Process_3$&$Pattern_{22,23,24,...}$ → … →$Process_1$&$Pattern_{45,46,47,...}$ → $Process_1$&$Pattern_{29,30,31,...}$→…

4.2 Training pattern matching

In this part, user can assign two things. One is what processes are used in the training. As Table 3 shown, H-Net can use multi process at one training epoch. For example, the model has dual route and need the separate training about each route in each epoch, two processes can be inserted at one training pattern matching.

Another thing is the matching what pattern packs are used in each process. There are three types orders which require the external patterns: 'Activation insert', 'Output layer error calculation (sigmoid)', and 'Output layer error calculation (softmax)' (see Table 2). To train the model, the external pattern should to be assigned to these orders.

The method is below.

1. Click the 'Making' button.
2. Select the pattern pack which user want at the 'Pattern pack' combo-box.
3. Select the process which user want at the 'Process' combo-box.
4. Click the 'Auto assign' button.
   A. If some pairs are inserted at 'Order ← Pattern information' list box and there is no order in 'Order' combo-box, the matching maybe correctly finished.
   B. If user see the warning message **"There is no matching pattern. Auto assign was suspended. This pair cannot use together."**, it means there is no matched pattern which is compatible at least one or more order. User should to check the model structure (especially, the **unit size of layer**) and the **pattern size** of selected pattern pack.
   C. Auto assign function is not perfect. In some cases, H-Net maybe fail to match all orders and patterns. In this cases, user should to match them manually.
5. After all order and patterns being matched, click the 'End' button.
6. If there is more training process which user want, go back to 1.

The auto assign function uses several criteria to match. At first, this function uses the comparison **the sizes of order 's layer and pattern**. If there is only one candidate, function will match them. If the candidate is two or more, function checks **the name of order's layer and pattern** among the candidates which is extracted by the first criterion. If there is only one candidate, function will match them. If the candidate is two or more by second criterion, function checks **the current cycle of order and suffix of pattern** among the candidates which is extracted by **the first criterion**. In this case, the suffix is the string which is posterior to the character '_'. If there is only one candidate, function will match them. If the candidate is two or more by second criterion, function finally checks **the suffix of order's layer and pattern** among the candidates which is extracted by **the first criterion**.

4.3 Test pattern matching

This part is very like the training pattern matching. But user should assign one more thing: 'Activation extract'. In process setup, 'Activation extract' order was told as a pointer to extract the result (see Table 2). In extract data from activation, user can assign what values are extracted. H-Net supports 4 values (see Table 4). Because mean

squared error and cross entropy need some target pattern, user must to assign the pattern. On the other hand, the pattern does not need when raw activation or semantic stress is extracted.

Table 4. The values which H-Net support to extract.

File name	Content
Raw activation	The raw activation of selected layer.
Mean squared error	The squared error is calculated by the below equation. $\sqrt{\sum \frac{(t_i - o_i)^2}{n}}$
Cross entropy	The cross entropy is calculated by the below equation. $-\frac{\sum t_i \log_e o_i + (1 - t_i) \log_e (1 - o_i)}{n}$
Semantic stress	The semantic stress is calculated by the below equation. $\frac{\sum o_i \log_2 o_i + (1 - o_i) \log_2 (1 - o_i) + 1}{n}$

4.4 Learning Setup Save & Load

Like other menus, there are the save and load functions about the learning setup. The saved file has 'HNet_Learning_Setup' extension. HNet_Learning_Setup files depend on the all other information like the structure, patterns, and processes. Also, like the other setup, when program load the HNet_Learning_Setup file, all current learning setup information will be swept. Thus, user should to be careful.

5 Learning

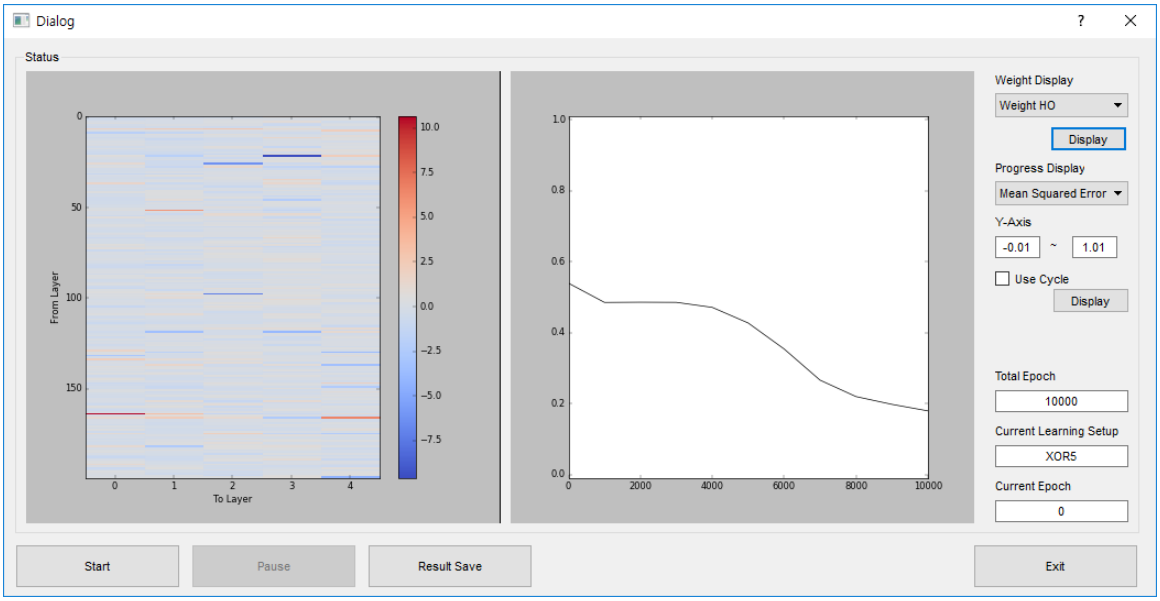

Figure 27. the learning example.

In learning, the model conduct the learning and test. In this window, when user clicks the 'Start' button, the training is started. If the 'Pause' button is clicked while model training, the training will be paused until user clicks the start button again.

5.1 Weight Display

While the model conduct training, user can see two graphs about weight and progress statuses. At the right-upside of learning window, there is a panel for control the weight status graph. User can select the weight connection or the bias of layer, and the selected weight or bias will be displayed at the left of status panel as the tile graph. If user select a weight of connection, the rows and columns of tiles mean the from and to layer of selected connection. On the other hand, If user select a bias of layer, there is no row difference because the row size of bias is always 1. The value is automatically changed at every 0.1 second, but user can change the displayed weight by clicking the 'Display' button of the panel.

5.2 Progress Display

At the below of the weight display control panel, there is another panel for control the progress status graph. User can select the displayed result type among mean squared error, cross entropy, and semantic stress. However, **to show this information, user must assign the value to extract at the learning setup**. **If there is no extraction about selected value, there is also no graph change**.

One important is the 'Use cycle' check-box. Basically, the progress graph panel shows the line graph, but this graph cannot the cycle progress in each epoch. If user checks the check box, the graph type will be changed to the tile graph, the panel shows the cycle progress (see Figure 27 and Figure 28).

On the other hand, user can control the y-axis of graph by modifying 'Y-Axis' text-boxes. If the cycle progress mode is on, the value of y-axis text-box will change the spectrum limitation of tiles.

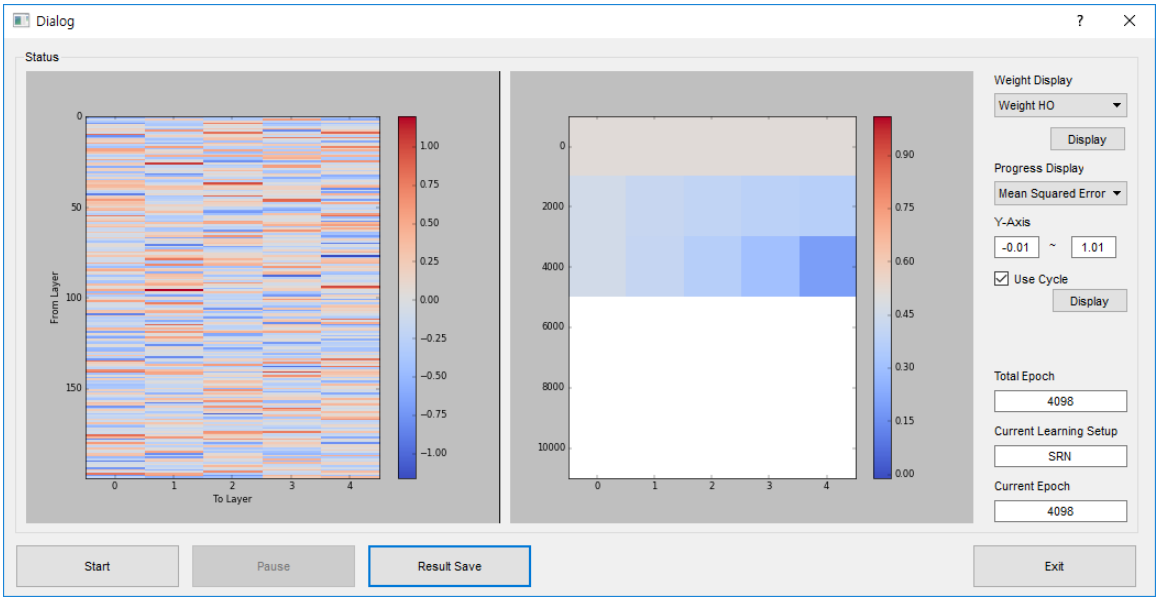

Figure 28. The example of progress graph when 'Use cycle' check-box is checked.

5.3 Test result save

Basically, when model finish to train, the test result is saved automatically. The save folder name is 'YYYYMMDD hhmmss Auto Save' (Y: year, M: month, D: day, h: hour, m: minute, s: second). But if user want to save before finishing training or to get another save data, user can click the 'Test result save' button. The result data is saved at the folder user selects. **One notice is that this button is only activated when model training status is pause or end.**

The save data includes the model information text file and the weight data as well as the result which user selected. The model information is a summary of all setup. This text file help to keep the training environment.

The weight data is about the model's all weights and biases. The saved file has 'HNet_Model' extension. In main window, user can load the previous trained weight by clicking the 'Weight & bias load' button. Importantly, **when the model structure of weight data file and current structure must be same including the name**. If not, the H-Net will show an error message about the load failure.

6 Macro

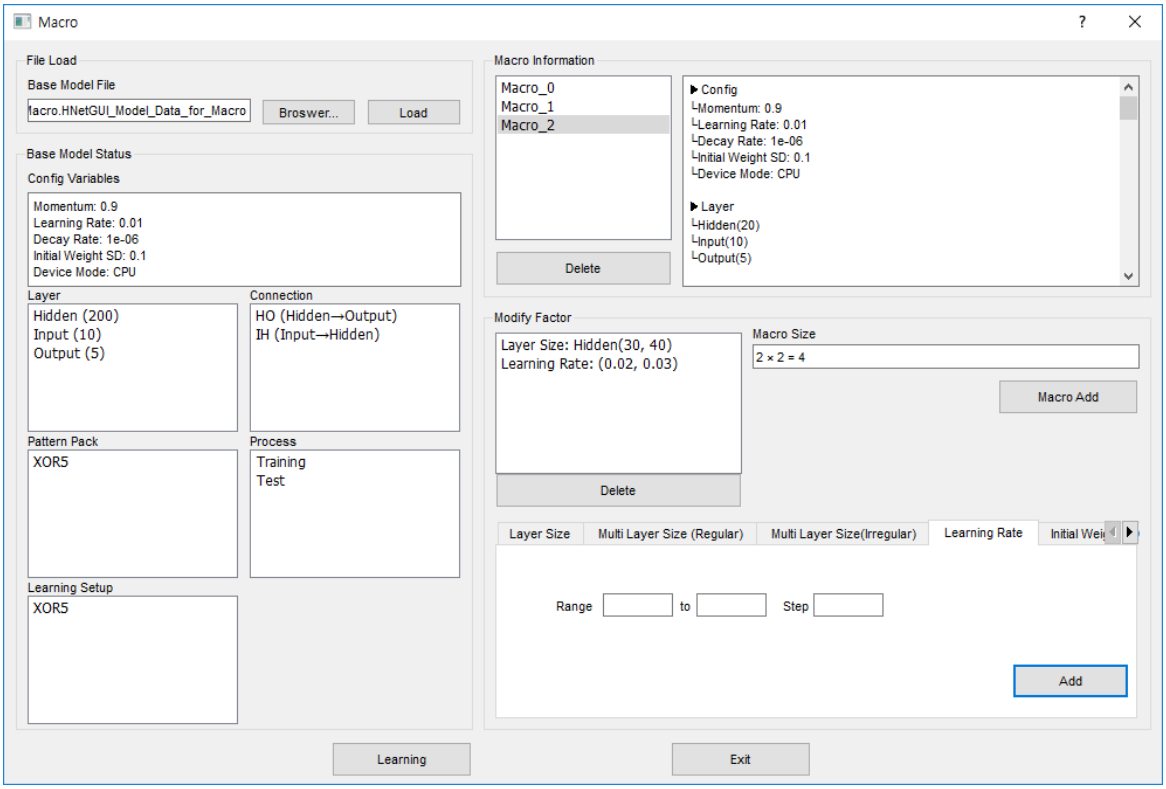

Figure 29. the batch mode windows.

Macro is for the reservation of multiple simulations. User can assign several variables to some multiple step values. This can help to check the variable's influence to the model.

To use macro, user need the file which the extension is 'HNetGUI_Model_Data_for_Macro'. **When user finish all set up, the 'Model save for macro' button of main window will be enabled, and user can save the model data for macro using this button**.

In macro window, after loading a macro file, user can change several variables to the step value by using the tabs of the modify factor panel.

6.1 Layer size (usual)

User can select which layer's size is changed and insert the range and the step of size. **One notice is that if there are some processes which use the selected layer as input or output, user cannot modify the layer**. This is because the layer is related with some patterns. The layer size change makes a compatibility problem. In short, the layer size is only able to be changed when the layer is hidden layer.

In addition, the layer which is affected by 'Bias equalization' or 'Layer duplication' order also cannot changed. Usually, the former order is used at the BPTT algorithm and latter order is used at the SRN algorithm. In this cases, user should to use the 'Multi layer size' tab.

6.2 Multi layer size (Regular)

When the names of layers which user want to change are regular like 'Hidden_1, …, 'Hidden_5', user can change the size of layers at one time. In particular, as mentioned above, when the model is based on the BPTT algorithm, the hidden layer size have to be changed by this method.

In this tab, instead of the layer combo-box, there are two text-box. When user inserts the regular part and maximum suffix number at the 'Layer prefix' and 'Max suffix', respectively, the the sizes of the layers will be changed. **Importantly, if there is a layer which is not affected despite the layer which is affected by 'Bias equalization' or 'Layer duplication' order, it will be fail by the compatibility problem**.

6.3 Multi layer size (Irregular)

When the names of layers which user want to change are irregular like ‘Hidden’ and ‘Context’, user can change the size of layers at one time. When user inserts all names of the layer which need to change the size at the ‘Layers’ text-box, the the sizes of the layers will be changed. Layer name inserted is separated by the space(‘ ’).

6.4 Learning rate, Initial weight SD

User can change the range of learning rate or initial weight SD.

6.5 Layer damage

This is related with the layer control of process setup. The selected layer will be damaged at selected process. By checking the check-box of ‘On’ or ‘Off’, other two options also can be inserted.

6.6 Connection damage

Like layer damage, this is related with the connection control of process setup. The selected connection will be damaged at selected process. By checking the check-box of ‘On’ or ‘Off’, other two options also can be inserted.

After clicking the ‘Learning’ button, H-Net starts the simulation automatically. In learning phase, user can pause or save the simulation like usual model.