these links would go unnoticed. There are also options, which additional links are to be copied. If only the substructure is to be copied, the command `Units Copy Structure None` is used.

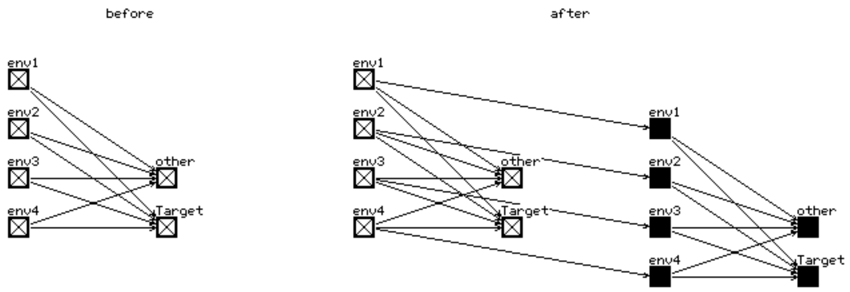

Figure 6.2: An Example for `Units Copy Structure with Forward binding`

The options with `binding` present a special feature. There, links between original and copied units are inserted automatically, in addition to the copied structure links. `Back`, `Forward` and `Double` specify thereby the direction of the links, where "back" means the direction towards the original unit. An example is shown in picture 6.2. If sites are used, the connections to the originals are assigned to the site selected in the popup. If not all originals have a site with that name, not all new units are linked to their predecessors.

With these various copy options, large, complicated nets with the same or similar substructures can be created very easily.

20. **Mode Units** (:)

    Mode Links (:)

    Switches to the mode **Units** or **Links**. All sequences of the normal modes are available. The keys `U` and `L` need not be pressed anymore. This shortens all sequences by one key.

21. **Units ...Return** (:)

    Links ...Return (:)

    Returns to normal mode after executing `Mode Units`.

22. **Graphics All** (:)

    Graphics Complete (:)

    Graphics Units (:)

    Graphics Links (:)

These commands initiate redrawing of the whole net, or parts of the net. With the exception of `Graphics Complete`, all commands affect only the current display. They are especially useful after links have been deleted.

23. **Graphic Direction** ([unit] :)

    This command assigns arrowheads to all links leading to/from the unit selected by the mouse. This is done independently from the setup values. XGUI, however, does not recall that links have been drawn. This means that, after moving a unit, these links remain in the window, if the display of links is switched off in the SETUP.

24. **Graphics Move** (TARGET [empty]/[unit] :)

    The origin of the window (upper left corner) is moved in a way that the `target` unit in the info panel becomes visible at the position specified by the mouse.

25. **Graphics Origin** ([empty]/[unit] :)

    The position specified by the mouse becomes new origin of the display (upper left corner).

26. **Graphics Grid** (:)

    This command draws a point at each grid position. The grid, however, is not refreshed, therefore one might have to redo the command from time to time.

6.6 Example Dialogue

A short example dialogue for the construction of an XOR network might clarify the use of the editor. First the four units are created. In the info panel the target name "input" and the `Target bias` "0" is entered.

Status Display	Command	Remark
`>`	`Mode Units`	switch on mode units
`Units>`		set mouse to position (3,5)
`Units>`	`Insert Target`	insert unit 1 with the attributes
		of the `Target` unit here.
		repeat for position (5,5).
`Units>`		name = "hidden", bias = −2.88
`Units>`	`Insert Target`	position (3,3); insert unit 3
`Units>`		name = "output", bias = −3.41
`Units>`	`Insert Target`	position (3,1); insert unit 4
`Units>`	`Return`	return to normal mode
`>`	`Mode Links`	switch on mode links
`Links>`		select both input units
		and set mouse to third unit

		("hidden")
`Links>`		specify weight "6.97"
`Links>`	`Make to Target`	create links
`Links>`		set mouse to unit 4 ("output"); specify weight "−5.24"
`Links>`	`Make to Target`	create links
`Links>`		deselect all units and select unit 3
`Links>`		set mouse to unit 4 and specify "11.71" as weight.
`Links>`	`Make to Target`	create links

Now the topology is defined. The only actions remaining are to set the IO types and the four patterns. To set the IO types, one can either use the command `Units Set Default io-type`, which sets the types according to the topological position of the units, or repeatedly use the command `Units Set io-Type`. The second option can be aborted by pressing the `Done` button in the popup window before making a selection.

Chapter 7

Network Creation Tools

SNNS provides five tools for easy creation of large, regular networks. All these tools carry the common name BigNet. They are called by clicking the button `BigNet` in the manager panel. This invokes the selection menu of figure 7.1, where the individual tools can be selected. This chapter gives a short indroduction to the handling of each of them.

Figure 7.1: The BigNet Menu Window

7.1 BigNet for Feed-Forward Networks

7.1.1 Terminology of the Tool BigNet

BigNet subdivides a net into several planes. The input layer, the output layer and every hidden layer are called a **plane** in the notation of BigNet. A plane is a two-dimensional array of units. Every single unit within a plane can be addressed by its coordinates. The unit in the upper left corner of every plane has the coordinates (1,1). A group of units within a plane, ordered in the shape of a square, is called a **cluster**. The position of a cluster is determined by the coordinates of its upper left corner and its expansion in the x direction (width) and y direction (height) (fig. 7.3).

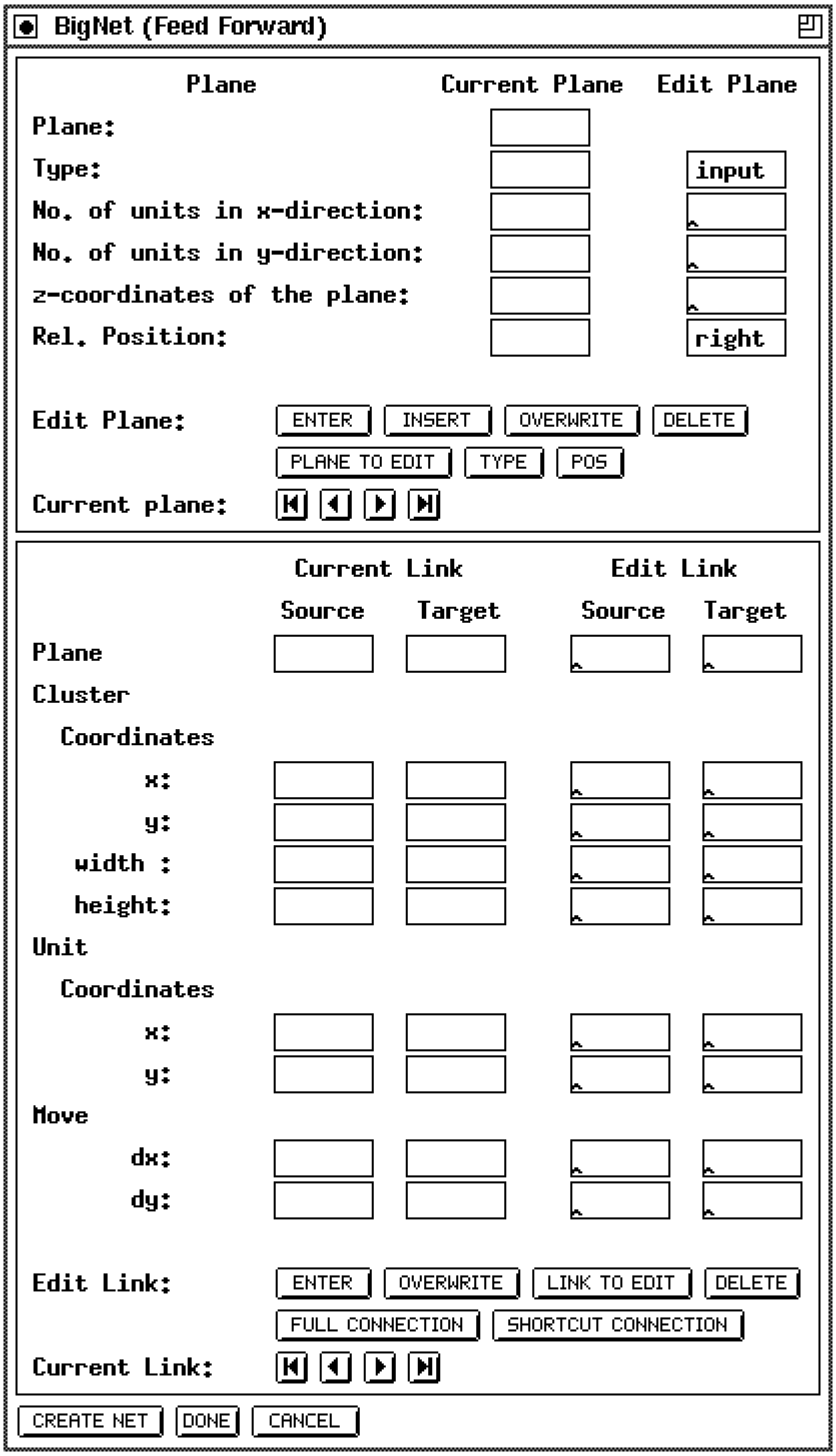

Figure 7.2: The BigNet window for Feed-Forward Networks

BigNet creates a net in two steps:

1. Edit net: This generates internal data structures in BigNet which describe the network but doesn't generate the network yet. This allows for easy modification of the network parameters before creation of the net.
   The net editor consists of two parts:
   (a) The plane editing part for editing planes. The input data is stored in the **plane list**.
   (b) The link editing part for editing links between planes. The input data is stored in the **link list**.
2. Generate net in SNNS: This generates the network from the internal data structures in BigNet.

Both editor parts are subdivided into an input part (Edit plane, Edit link) and into a display part for control purposes (Current plane, Current link). The input data of both editors is stored, as described above, in the plane list and in the link list. After pressing ENTER, INSERT, or OVERWRITE the input data is added to the corresponding editor list. In the control part one list element is always visible. The buttons , , , and enable moving around in the list. The operations DELETE, INSERT, OVERWRITE, CURRENT PLANE TO EDITOR and CURRENT LINK TO EDITOR refer to the current element. Input data is only entered in the editor list if it is correct, otherwise nothing happens.

7.1.2 Buttons of BigNet

`ENTER` : Input data is entered at the end of the plane or the link list.

`INSERT` : Input data is inserted in the plane list in front of the current plane.

`OVERWRITE` : The current element is replaced by the input data.

`DELETE` : The current element is deleted.

`PLANE TO EDIT` : The data of the current plane is written to the edit plane.

`LINK TO EDIT` : The data of the current link is written to the edit link.

`TYPE` : The type (input, hidden, output) of the units of a plane is determined.

`POS` : The position of a plane is always described relative (left, right, below) to the position of the previous plane. The upper left corner of the first plane is positioned at the coordinates (1,1) as described in Figure 7.4. BigNet then automatically generates the coordinates of the units.

`FULL CONNECTION` : A fully connected feed forward net is generated. If there are n planes numbered $1..n$ then every unit in plane i with $i > 0$ is connected with every unit in plane $i+1$ for all $1 \leq i \leq n-1$.

`SHORTCUT CONNECTION` : If there exist n planes $1 \ldots n$ then every unit in plane i with $1 \leq i < n$ is connected with every unit in all planes j with $i < j \leq n$.

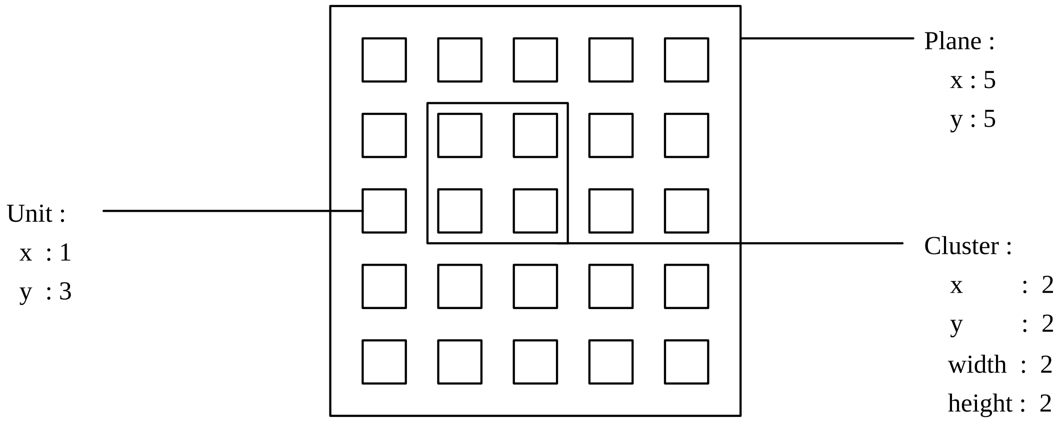

Figure 7.3: Clusters and units in BigNet

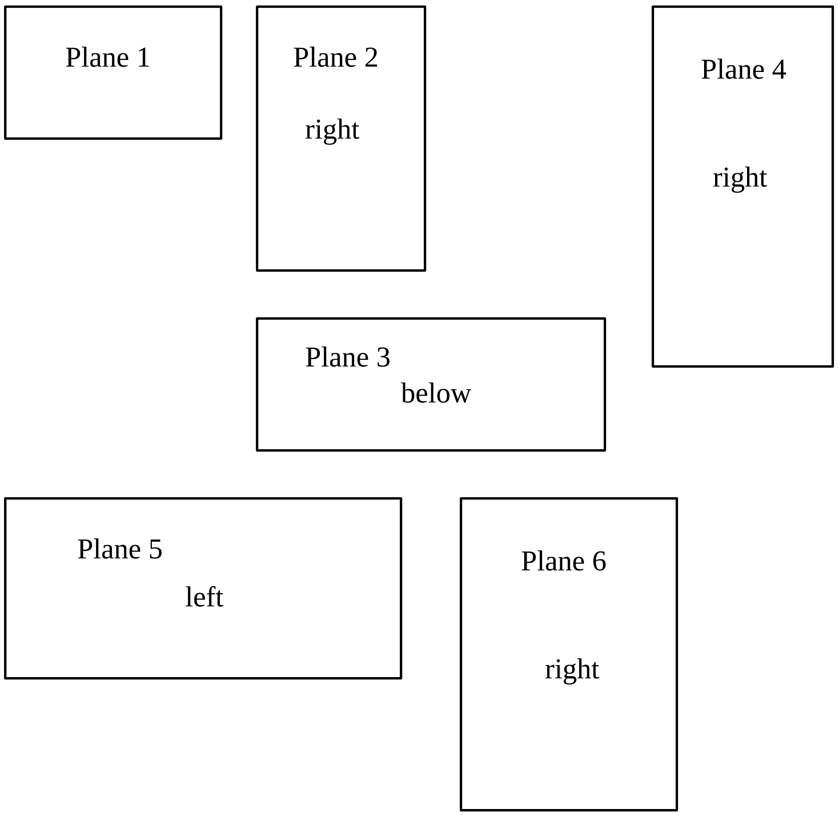

Figure 7.4: Positioning of the planes

CREATE NET : The net described by the two editors is generated by SNNS. The default name of the net is **SNNS_NET.net**. If a net with this name already exists a warning is issued before it is replaced.

CANCEL : All internal data of the editors is deleted.

DONE : Exit BigNet and return to the simulator windows.

7.1.3 Plane Editor

Every plane is characterized by the number of units in x and y direction. The unit type of a plane can be defined and changed by `TYPE`. The position of the planes is determined relative to the previous plane. The upper left corner of plane no. 1 is always positioned at the coordinates $(1, 1)$. Pressing `POS`, one can choose between 'left', 'right' and 'below'. Figure 7.4 shows the layout of a network with 6 planes which were positioned relative to their predecessors as indicated starting with plane 1.

Every plane is associated with a plane number. This number is introduced to address the planes in a clear way. The number is important for the link editor. The user cannot change this number.

In the current implementation the z coordinate is not used by BIGNET. It has been implemented for future use with the 3D visualization component.

7.1.4 Link Editor

A link always leads from a source to a target. To generate a fully connected net (connections from each layer to its succeeding layer, no shortcut connections), it is only sufficient to press the button `FULL CONNECTION` after the planes of the net are defined. Scrolling through the link list, one can see that every plane i is connected with the plane $i+1$. The plane number shown in the link editor is the same as the plane number given by the plane editor.

If one wants more complicated links between the planes one can edit them directly. There are nine different combinations to specify link connectivity patterns:

$$\text{Links from} \left\{ \begin{array}{c} \text{all units of a plane} \\ \text{all units of a cluster} \\ \text{a single unit} \end{array} \right\} \text{to} \left\{ \begin{array}{c} \text{all units of a plane} \\ \text{all units of a cluster} \\ \text{a single unit} \end{array} \right\}.$$

Figure 7.5 shows the display for the three possible input combinations with (all units of) a plane as source. The other combinations are similar. Note that both source plane and target plane must be specified in all cases, even if source or target consists of a cluster of units or a single unit. If the input data is inconsistent with the above rules it is rejected with a warning and not entered into the link list after pressing `ENTER` or `OVERWRITE`.

With the Move parameters one can declare how many steps a cluster or a unit will be moved in x or y direction within a plane after the cluster or the unit is connected with a target or a source. This facilitates the construction of receptive fields where all units of a cluster feed into a single target unit and this connectivity pattern is repeated in both directions with a displacement of one unit.

The parameter dx (delta-x) defines the step width in the x direction and dy (delta-y) defines the step width in the y direction. If there is no entry in dx or dy there is no movement in this direction. Movements within the source plane and the target plane is independent from each other. Since this feature is very powerful and versatile it will be illustrated with some examples.

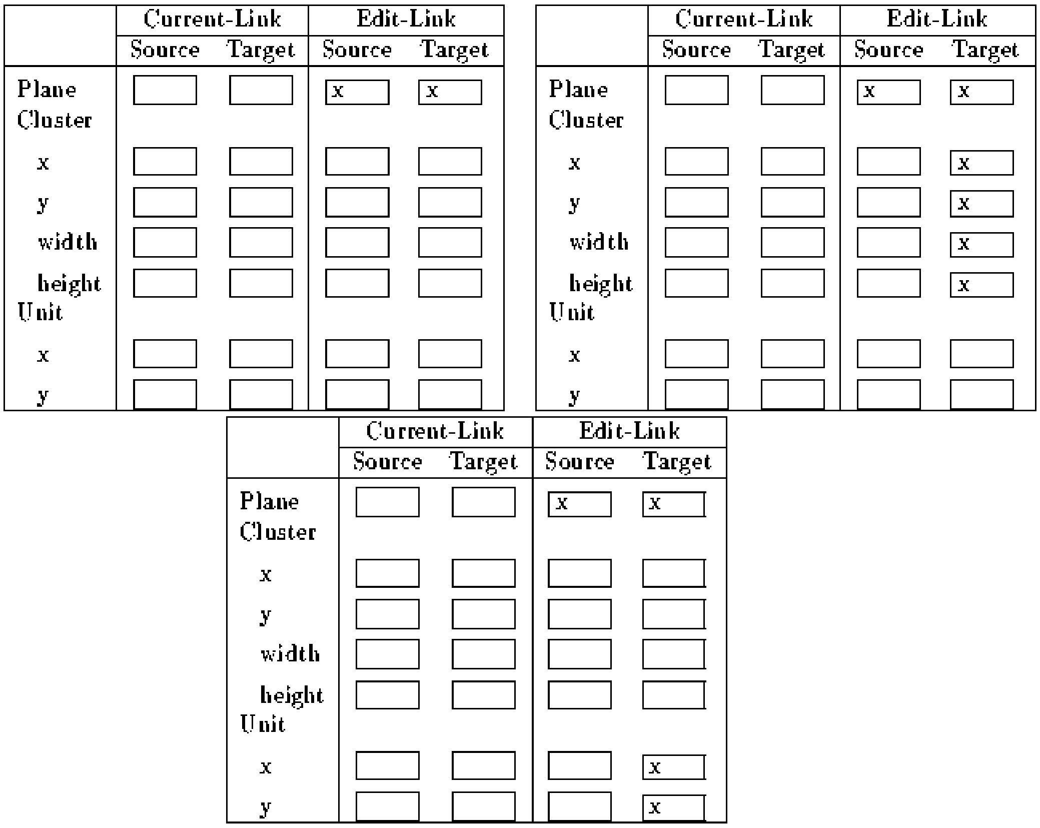

Figure 7.5: possible input combinations with (all units of) a plane as source, between 1) a plane and a plane, 2) a plane and a cluster, 3) a plane and a unit. Note that the target plane is specified in all three cases since it is necessary to indicate the target cluster or target unit.

Example 1: Receptive Fields in Two Dimensions

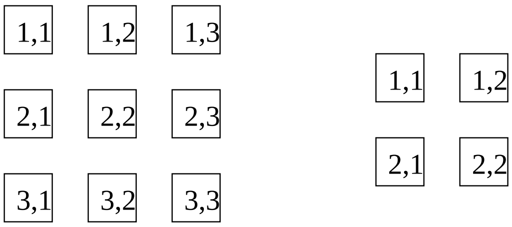

Figure 7.6: The net of example 1

There are two planes given (fig. 7.6). To realize the links

source: plane 1 (1,1), (1,2), (2,1) (2,2) $\longrightarrow$ target: plane 2 (1,1)
source: plane 1 (1,2), (1,3), (2,2) (2,3) $\longrightarrow$ target: plane 2 (1,2)
source: plane 1 (2,1), (2,2), (3,1) (3,2) $\longrightarrow$ target: plane 2 (2,1)
source: plane 1 (2,2), (2,3), (3,2) (3,3) $\longrightarrow$ target: plane 2 (2,2)

between the two planes, the move data shown in figure 7.7 must be inserted in the link editor.

	Current-Link		Edit-Link	
	Source	Target	Source	Target
Plane			1	2
Cluster				
x			1	
y			1	
width			2	
height			2	
Unit				
x				1
y				1
Move				
delta-x			1	1
delta-y			1	1

Figure 7.7: Example 1

First, the cluster (1,1), (1,2), (2,1) (2,2) is connected with the unit (1,1). After this step the source cluster and the target unit are moved right one step (this corresponds to dx = 1 for the source plane and the target plane). The new cluster is now connected with the new unit. The movement and connection building is repeated until either the source cluster or the target unit has reached the greatest possible x value. Then the internal unit pointer moves moves down one unit (this corresponds to dy = 1 for both planes) and back to the beginning of the planes. The “moving” continues in both directions until the boundaries of the two planes are reached.

Example 2: Moving in Different Dimensions

This time the net consists of three planes (fig. 7.9). To create the links

source: plane1 (1,1), (1,2), (1,3) $\longrightarrow$ target: plane 2 (1,1)
source: plane1 (2,1), (2,2), (2,3) $\longrightarrow$ target: plane 2 (1,2)
source: plane1 (3,1), (3,2), (3,3) $\longrightarrow$ target: plane 2 (1,3)
source: plane1 (1,1), (2,1), (3,1) $\longrightarrow$ target: plane 3 (1,1)
source: plane1 (1,2), (2,2), (3,2) $\longrightarrow$ target: plane 3 (1,2)
source: plane1 (1,3), (2,3), (3,3) $\longrightarrow$ target: plane 3 (1,3)

between the units one must insert the move data shown in figure 7.8. Every line of plane 1 is a cluster of width 3 and height 1 and is connected with a unit of plane 2, and every column of plane 1 is a cluster of width 1 and height 3 and is connected with a unit of plane 3. In this special case one can fill the empty input fields of “move” with any data because a movement in this directions is not possible and therefore these data is neglected.

	Current-Link		Edit-Link	
	Source	Target	Source	Target
Plane			1	2
Cluster				
x			1	
y			1	
width			3	
height			1	
Unit				
x				1
y				1
Move				
delta-x				1
delta-y			1	

	Current-Link		Edit-Link	
	Source	Target	Source	Target
Plane			1	3
Cluster				
x			1	
y			1	
width			1	
height			3	
Unit				
x				1
y				1
Move				
delta-x			1	1
delta-y				

Figure 7.8: Example 2

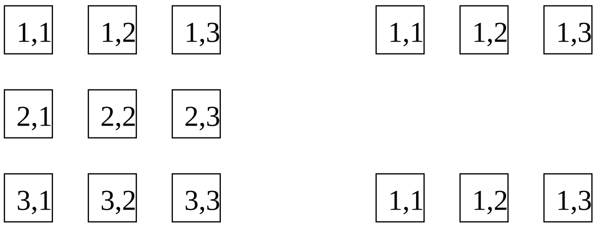

Figure 7.9: The net of example 2

7.1.5 Create Net

After one has described the net one must press `CREATE NET` to generate the net in SNNS. The weights of the links are set to the default value 0.5. Therefore one must initialize the net before one starts learning. The net created has the default name **SNNS_NET.net**. If a net already exists in SNNS a warning is issued before it is replaced. If the network generated happens to have two units with more than one connection in the same direction between them then SNNS sends the error message “Invalid Target”.

7.2 BigNet for Time-Delay Networks

The BigNet window for Time Delay networks (figure 7.10) consists of three parts: The Plane editor where the number, placement, and type of the units are defined, the link editor, where the connectivity between the layer is defined, and three control buttons at the bottom, to create the network, cancel editing, and close the window.

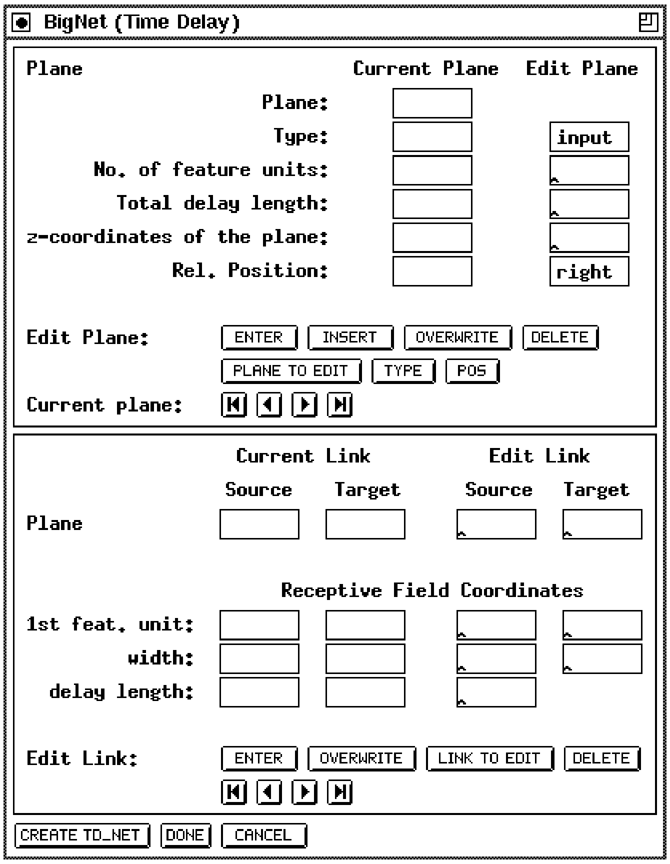

Figure 7.10: The BigNet window for Time Delay Networks

Since the buttons of this window carry mostly the same functionality as in the feed-forward case, refer to the previous section for a description of their use.

7.2.1 Terminology of Time-Delay BigNet

The following naming conventions have been adopted for the BigNet window. Their meaning may be clarified by figure 7.11.

- Receptive Field: The cluster of units in a layer totally connected to one row of units in the next layer.
- 1st feature unit: The starting row of the receptive field.

- width: The width of the receptive field.
- delay length: The number of significant delay steps of the receptive field. Must be the same value for all receptive fields in this layer.
- No. of feature units: The width of the current layer
- Total delay length: The length of the current layer. Total delay length times the number of feature units equals the number of units in this layer. Note that the total delay length must be the same as the delay length plus the total delay length of the next layer minus one!
- z-coordinates of the plane: gives the placing of the plane in space. This value may be omitted (default = 0).

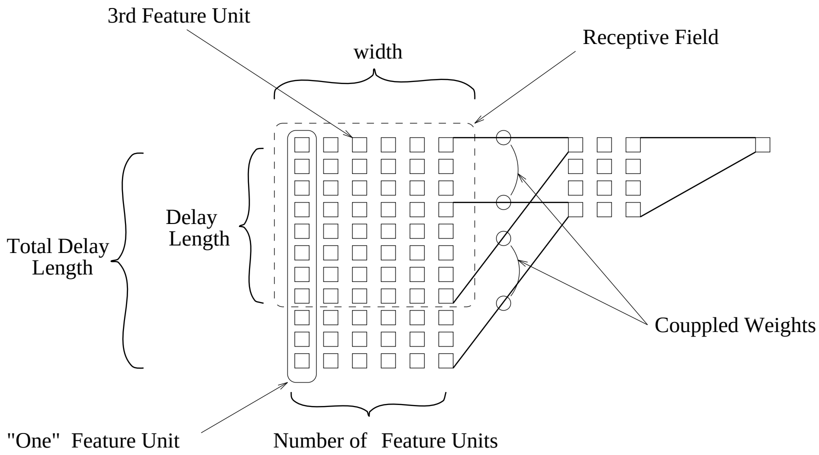

Figure 7.11: The naming conventions

7.2.2 Plane Editor

Just as in BigNet for feed-forward networks, the net is divided into several planes. The input layer, the output layer and every hidden layer are called a **plane** in the notation of BigNet. A plane is a two-dimensional array of units. Every single unit within a plane can be addressed by its coordinates. The unit in the upper left corner of every plane has the coordinates (1,1).

See 7.1.3 for a detailed description.

7.2.3 Link Editor

In the link panel the connections special to TDNNs can be defined. In TDNNs links always lead from the receptive field in a source plane to one or more units of a target plane. Note, that a receptive field has to be specified only once for each plane and is automatically applied to all possible delay steps in that plane. figure 7.12 gives an example of a receptive field specification and the network created thereby.

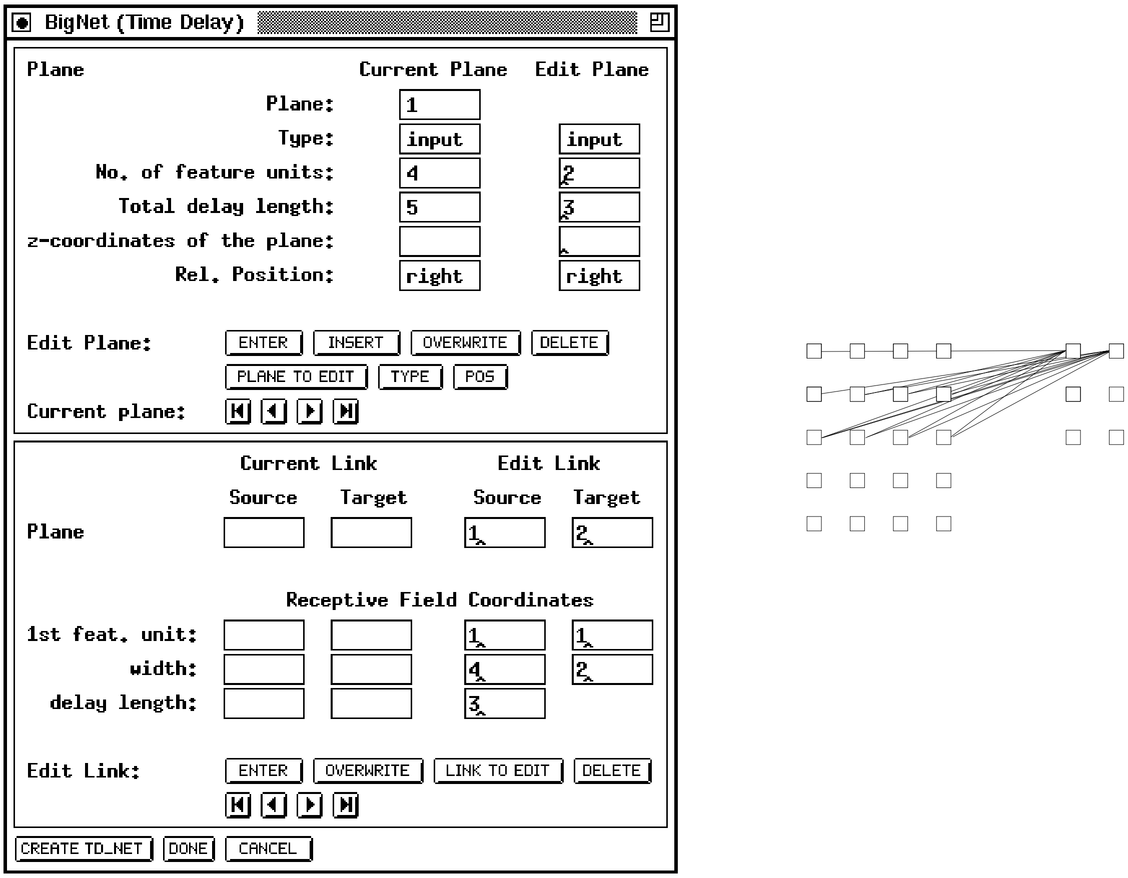

Figure 7.12: An example TDNN construction and the resulting network

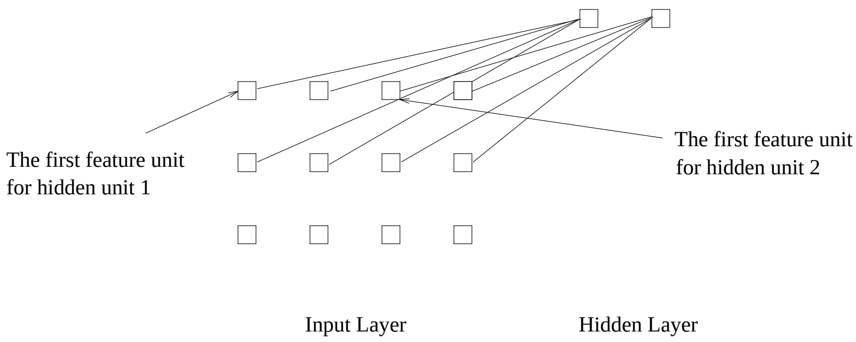

Figure 7.13: Two receptive fields in one layer

It is possible to specify seperate receptive fields for different feature units. With only one receptive field for all feature units, a "1" has to be specified in the input window for "1st feature unit:". For a second receptive field, the first feature unit should be the width of the first receptive field plus one. Of course, for all number of receptive fields, the sum of their width has to equal the number of feature units! An example network with two receptive fields is depicted in figure 7.13

7.3 BigNet for ART-Networks

The creation of the ART networks is based on just a few parameters. Although the network topology for these models is rather complex, only four parameters for ART1 and ART2, and eight parameters for ARTMAP, have to be specified.

If you have selected the ART 1, ART 2 or the ARTMAP button in the BigNet menu, one of the windows shown in figure 7.14 appears on the screen.

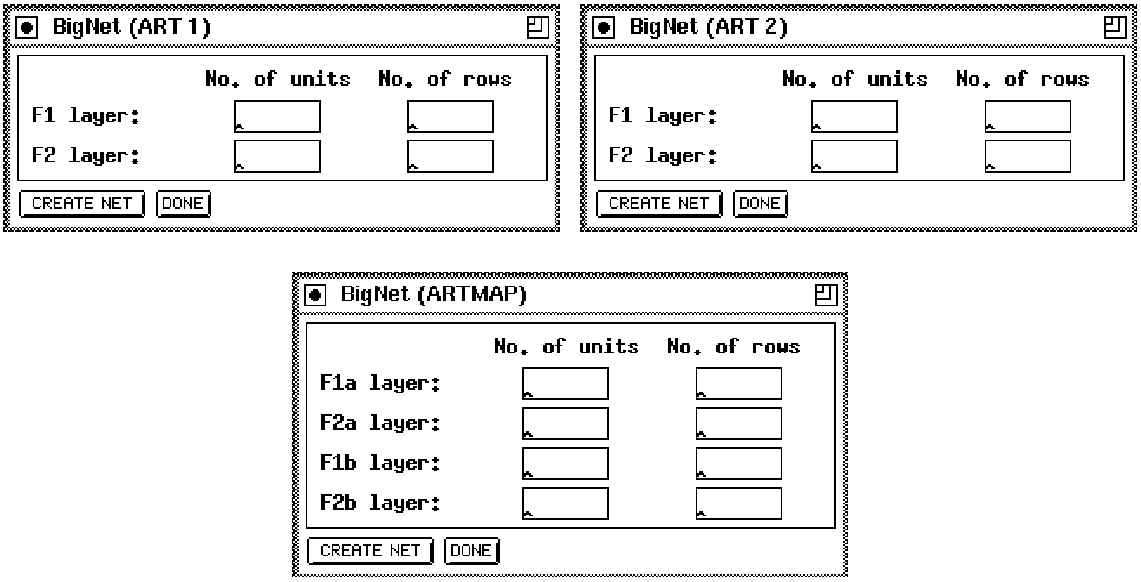

Figure 7.14: The BigNet windows for the ART models

The four parameters you have to specify for ART1 and ART2 are simple to choose. First you have to tell BigNet the number of units (N) the F_1 layer consists of. Since the F_0 layer has the same number of units, BigNet takes only the value for F_1.

Next the way how these N units to be displayed has to be specified. For this purpose enter the number of rows. An example for ART1 is shown in figure 7.15.

The same procedure is to be done for the F_2 layer. Again you have to specify the number of units M for the recognition part[1] of the F_2 layer and the number of rows.

Pressing the CREATE NET button will generate a network with the specified parameters. If a network exists when pressing CREATE NET you will be prompted to assure that you really want to destroy the current network. A message tells you if the generation terminated successfully. Finally press the DONE button to close the BigNet panel.

[1]The F_2 layer consists of three internal layers. See chapter 9.12.

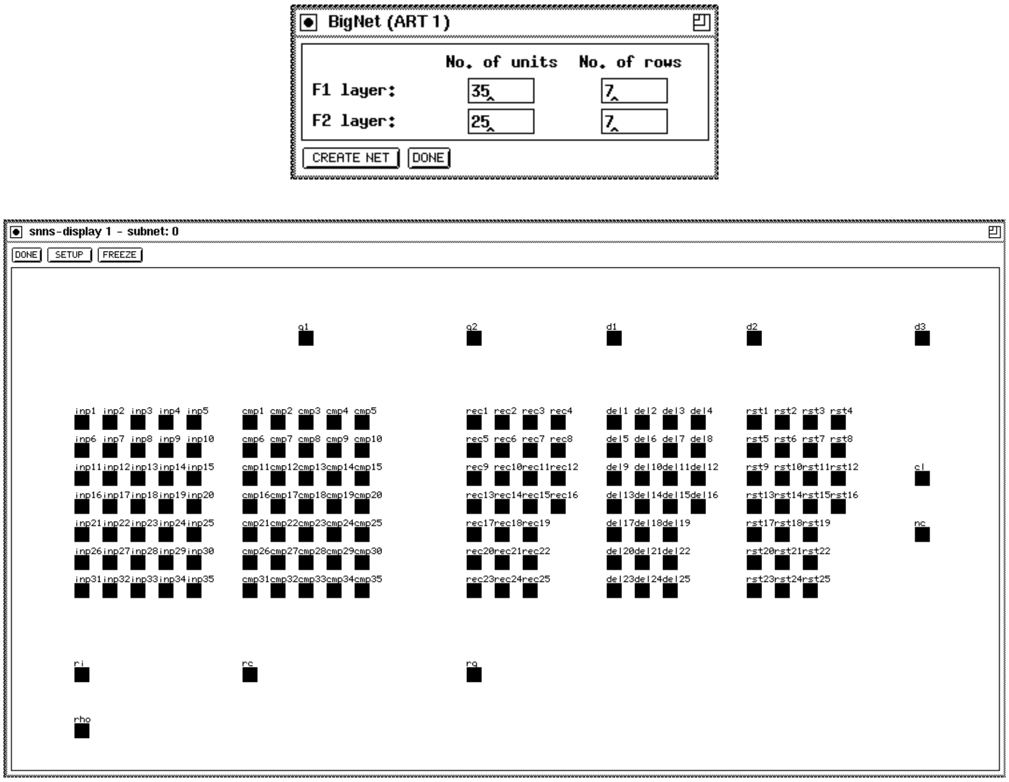

Figure 7.15: Example for the generation of an ART1 network. First the BigNet (ART1) panel is shown with the specified parameters. Next you see the created net as you can see it when using an SNNS display.

For ARTMAP things are slightly different. Since an ARTMAP network exists of two ART1 subnets (ART^a and ART^b), for both of them the parameters described above have to be specified. This is the reason, why BigNet (ARTMAP) takes eight instead of four parameters. For the MAP field the number of units and the number of rows is taken from the repective values for the F_2^b layer.

7.4 BigNet for Self-Organizing Maps

As described in chapter 9.13, it is recommended to create Kohonen-Self-Organizing Maps only by using either the **BigNet** network creation tool or **convert2snns** outside the graphical user interface. The SOM architecture consists of the component layer (input layer) and a two-dimensional map, called competitive layer. Since component layer and competitive layer are fully connected, each unit of the competitive layer represents a vector with the same dimension as the component layer.

To create a SOM, only 3 parameters have to be specified:

- *Components:* The dimension of each weight vector. It equals the number of input units.