Human-Computer Interaction and Neural Networks in Architectural design

A Tool for Design Exploration.

NEANDER F. SILVA
Department of Architectural Design
University of Brasília
Brasília, Brazil

ALAN H. BRIDGES
Department of Architecture and Building Science
University of Strathclyde
Glasgow, United Kingdom

Abstract

Design research has demonstrated that neural networks are able to support creativity. However, there are two main problems with using neural networks in design. One is how you interact with such systems. The second relates to the integration between neural network techniques and other approaches. This paper will describe an integrated model in which those problems are addressed. The resulting system provides an interface in which the neural network output is translated into textual and graphic representations that can play a meaningful role in the design process.

1. Introduction:

It has been suggested that neural networks yield support for emergence and creativity in design, as demonstrated by Coyne and Newton (1990), Coyne (1991), Coyne and Yokozawa (1992), and Coyne et al. (1993). Those authors argued that support for creativity is achieved by extracting information from implicit knowledge that can be translated as new explicit solutions (Coyne et al., 1993).

However, there are two major problems with using neural networks in design. One is how you interact with such systems. The bare interfaces of neural networks are very poor and passive from the designer’s point of view, rendering those systems virtually unusable in design. The second pertains to the integration between neural network
techniques and other approaches or knowledge representations which are essential to the
design process such as language and graphic descriptions.

2. Stand-alone neural networks in design:

An auto-associative neural network design was adopted by Coyne and Yokozawa
(1992). After being trained through the exposure to a certain number of examples,
represented by different sets of combinations of features, this network becomes ‘aware’
of what features are mutually excitatory or inhibitory. Coyne and Yokozawa (1992)
suggested that if a designer used such trained network by selecting and manipulating
features (neurons) on its input layer, the outcome would be not only combinations of
features matching examples from the training set, but eventually the emergence of new
combinations of features.

For instance, a neural network could be trained with the following set of instances
shown in figure 1, which is an example of a door entrance classification domain
composed of nine styles. This domain is obviously simplistic containing only 27
descriptors and night solutions. However, we have chosen to use it because it is just
big enough to illustrate the working of a stand-alone neural network, the alternative
expert system described later, and the proposed integrated model. Another reason is that
it calls for no specialist knowledge on the part of the reader.
Figure 1. Door entrance domain: illustrations.
The examples could be binary classified according to the presence or absence of certain features, such as those used in the classification shown in figure 2, below:

<table>
<thead>
<tr>
<th>Feature</th>
<th>Classic</th>
<th>Gothic</th>
<th>Art Nouveau</th>
<th>Functional</th>
<th>Brutalist</th>
<th>Organic</th>
<th>High Tech</th>
<th>Post Modern</th>
<th>Hybrid</th>
<th>Classic + High tech</th>
</tr>
</thead>
<tbody>
<tr>
<td>top flat moulding</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>top curved moulding</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>lateral vertical moulding</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>angular connection with glass tower</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>vertical glass tower</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>triangular pediment</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>pointed arch tympanum</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>tracery or steelwork on fanlight or tympanum</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>squared fanlight</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>fanlight with undulate top</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>lateral cylindrical section column</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Surrounding material: glass</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Surrounding material: brick</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>surrounding material: smooth stone</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>surrounding material: concrete exposed</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>surrounding material: timber</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>surrounding material: plasterwork</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>opening mechanism: swinging door</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>opening mechanism: revolving door</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>leaf type: plain opaque</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>leaf type: panelled opaque</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>leaf type: framed</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>steelwork as leaf decoration</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>leaf material: glass</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>leaf material: metal</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>leaf material: timber</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

Figure 2. Binary classification of the set of instances.

If after training, the user chooses to make active the input units ‘surrounding material: metal’ and ‘leaf material: metal’, the output will be a binary string which represents the solution number 3, ‘Art Nouveau’. This is illustrated in figure 3, below.
Figure 3. The 'Art Nouveau’ entrance description built by the trained neural network.

However, if the user chooses to make active the input units 'triangular pediment', 'opening mechanism: revolving door', and 'leaf material: glass', the output will be the one shown in figure 4, which is a new combination of features, but still represents a sensible solution from the architectural point of view.
<table>
<thead>
<tr>
<th></th>
<th>Input layer</th>
<th>Output layer</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>top flat moulding</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>top curved moulding</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>lateral vertical moulding</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>angular connection with glass tower</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>vertical glass tower</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>triangular pediment</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>pointed arch tympanum</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>tracery or steelwork on fanlight or tympanum</td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>squared fanlight</td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>fanlight with undulate top</td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>lateral cylindrical section column</td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>Surrounding material: glass</td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>Surrounding material: brick</td>
<td></td>
</tr>
<tr>
<td>14</td>
<td>Surrounding material: smooth stone</td>
<td></td>
</tr>
<tr>
<td>15</td>
<td>Surrounding material: concrete exposed</td>
<td></td>
</tr>
<tr>
<td>16</td>
<td>Surrounding material: timber</td>
<td></td>
</tr>
<tr>
<td>17</td>
<td>Surrounding material: plasterwork</td>
<td></td>
</tr>
<tr>
<td>18</td>
<td>Surrounding material: metal</td>
<td></td>
</tr>
<tr>
<td>19</td>
<td>opening mechanism: swinging door</td>
<td></td>
</tr>
<tr>
<td>20</td>
<td>opening mechanism: revolving door</td>
<td></td>
</tr>
<tr>
<td>21</td>
<td>leaf type: plain opaque</td>
<td></td>
</tr>
<tr>
<td>22</td>
<td>leaf type: paneled opaque</td>
<td></td>
</tr>
<tr>
<td>23</td>
<td>leaf type: framed</td>
<td></td>
</tr>
<tr>
<td>24</td>
<td>steelwork as leaf decoration</td>
<td></td>
</tr>
<tr>
<td>25</td>
<td>leaf material: glass</td>
<td></td>
</tr>
<tr>
<td>26</td>
<td>leaf material: metal</td>
<td></td>
</tr>
<tr>
<td>27</td>
<td>leaf material: timber</td>
<td></td>
</tr>
</tbody>
</table>

*Figure 4. New solution emerged from the trained neural network.*

The results shown in figures 3 and 4 are consequences of the weights set by the neural network for the connections between each input unit and all output units during the training process.

The above method of using such networks in a direct input layer manipulation, has as main advantage the freedom of testing and of possibly ‘forcing’ hybrid solutions by picking up pairs of input neurons otherwise considered unusual in the training set.
However, there are disadvantages: the user may get an output that is actually not a stable state, once the process can be terminated arbitrarily. The user may also have to undertake a trial process in which he or she may get lost, once there is no inherent trace facility. Moreover, the user may end up in a dead lock in which the network cannot compute a stable state because the units made active are highly incompatible.

At last, the method does not provide either a plain English interface or a graphic one. In addition, its bare interface is completely passive, that is, all the initiative relies on the user, who must know what he or she wants and also keep track of all events manually.

3. **Mustoe's alternative knowledge-based system:**

Alternative knowledge-base systems have been devised (Frey, 1986; Mustoe, 1990, 1993). Mustoe (1990) argues that evidently the function of the network of rules in conventional knowledge-based systems is to place a set of individual productions into a correct relationship with a particular solution. Solutions are classified according to their question set, while questions are classified by reference to the solutions they verify.

Therefore, Mustoe (1990, 1993) proposes that a domain knowledge may be better represented through a Boolean classification structure, resembling the table shown in figure 2.

This kind of representation may suggest that we actually went back to a relational database. However, there are substantial differences. The table in figure 2 just illustrates how Mustoe’s system, ‘Cortex’, maps questions into solutions. The solutions are classified independently from each other in a set of binary relationships with the questions that verify them. These relationships are actually encoded in the system as true bit-strings and not as rules. The control system will not operate on them through a query language, but through a direct bit-string manipulation.

Cortex (Mustoe, 1990, 1993) uses a control algorithm that functions by rejecting falsified solutions. This system would verify each of the features shown in the table of figure 2 through a series of questions subsequently presented to the user. It will present to the user the currently most frequent question among those still relevant. It will finish either with one or more not falsified solutions, or with a confession that it cannot find a solution. It operates in a process of zeroing-in upon a successively shortened list of still-possible solutions.

The advantages of this kind of knowledge representation are two: firstly, the system will run faster than a rule-based one due to the direct bit-string manipulation.
Secondly, the addition of new solutions will not require a time consuming and expensive process of re-writing ‘if... then’ statements, as it happens with conventional rule-based systems. The addition of a new solution will imply only the incorporation of another bit-string to the existing set.

A drawback of this kind of knowledge representation is that, in complex domains, a huge number of conditions, or questions, can be generated by this approach. However, this is largely compensated by the control system, as described earlier.

If no inconsistency is necessarily introduced on the knowledge-base already in the system with the addition of new solutions, new horizons are open for building systems in which the knowledge-bases can be consistently expanded. However, as the options must be manually set before hand, Cortex (Mustoe, 1990, 1993) has no inherent knowledge acquisition mechanism as any other knowledge-base system. Nevertheless, its integration with a particular design of connectionist model may provide the answers for this problem.


In spite of having quite different control algorithms, there are striking similarities between Cortex and binary neural networks concerning knowledge representation. This similarities call for a potential integration. One way of integrating Cortex with Coyne’s model could be the use of the first as the front end of the second. Cortex would act as an intelligent and active interface by filtering user’s input, by focusing the user’s interest on the most promising solutions and by formatting data for the neural network.

There are two main obstacles to the direct use of the user’s answers to Cortex in the input layer of neural networks. The first is related to some differences in knowledge representation. Cortex accepts three possible states for each feature: definitely present, if the answer is ‘yes’, definitely absent, if the answer is ‘no’, and neutral, if the answer is ‘don’t know’. However, a binary neural network input unit has obviously only two states: active and inactive.

The second obstacle is the way in which the neural network is trained and the amount of information provided by the set of answers to Cortex. Very often few of the questions in the question’s set of a knowledge-base will be used by Cortex in order to find a solution. Therefore, the user’s answers provide limited information, which is rarely enough to produce a stable solution in a neural network.

Both obstacles are overcome if we use a plain feed-forward multi-layered network at training time and a semi-recurrent network, with limited feed-back, at running time. Why do we not use one of this architectures at both training and running time? Firstly,
a plain feed-forward multi-layered network is a sufficiently effective architecture to train the kind of knowledge representation that has been described in this paper.

Secondly, the recurrent element that is added at running time, and will be described shortly, is used to provide external control to the process of automating the search for a stable state that would otherwise be undertaken manually. It is not a standard neural network component based in fuzzy logic, but an element that emulates the role played by the user such as seen in Coyne and Yokozawa’s (1992) experiment.

If we considered a network design in which each input unit had two attributes, the first specifying if it is active or inactive, and the second specifying if it is ‘clamped’ or ‘unclamped’, the user’s answers to Cortex could then be used as neural network input. In other words, each input unit would have, besides two possible states (active or inactive), an extra attribute that could have a constant value, if the unit’s status is clamped, or a variable value, if the unit’s status is unclamped.

A recurrent element between each output unit and its correspondent input unit is then added at running time in which each output is checked against the clamping criteria described above. Figure 5 illustrates this feed-back process.
In other words, all the output values of ‘unclamped’ units are accepted and successively re-entered as input. Also, all the output values of ‘clamped’ units are checked and reset to their initial state (if it has been changed), and successively re-entered as input.

The network could then search for stable states in which reliable solutions could be produced. The process would be terminated in two situations: firstly, when no further changes or mismatches are observed between the input layer and the output layer. Obviously, if the output layer mirrors the input one, there is no reason for re-entering the output values as input any more. Secondly, when the network reaches an infinite loop, that is, the last output equals an output of a previous of stage in the same running process. When these situations happen the process is terminated by the model’s algorithm.
This process could lead to three outcomes. Firstly, it could lead to a solution that matches the solution presented by Cortex. Secondly, it could lead to a solution that does not match the solution presented by Cortex. Since the solution found by Cortex is the only one in the knowledge-base able to satisfy the user’s input, the network’s solution will not match any of the solutions in the original set of examples either. It is thus a new solution. The binary string of such new solutions can be then easily converted into plain English textual descriptions composed of a list of the features present in the new combinations. From this point onwards the descriptions of precedents can be coupled with graphical illustrations, and the textual descriptions of possible new solutions can be coupled with sets of 3D components without too much difficulty. Therefore, an easy transition from textual information to 3D modelling as design medium may be also provided, in which the first finds a more relevant use and the last is enhanced as an intelligent activity.

5. Cortex and ‘semi-recurrent’ network integration: a working example.

Suppose that a user approaches this system. What this user will see on the screen and interact with will be as follows. The system starts by bringing the first question to the screen. This is the question that verifies the most common feature in the set of cases:

Is it supposed to have glass among the surrounding materials?

As the user is not sure yet to which extent glass is to be used around the entrance, he or she answers ‘don’t know’ to this question. The system then brings to screen the next question:

Is it supposed to have a swinging door?

This dialogue continues until the system has tested all the relevant features describing the set of cases. It then brings to the screen the textual description of a first possible solution:

The solution may be a functionalist door entrance, with the following features:
- vertical glass tower
- surrounding materials: glass, plasterwork and metal
- swinging door
- leaf type: framed with one or more light cross panels
- leaf materials: non-stained glass, metal

The solution’s description is followed by the graphic illustration of the precedent, in this case in a bit-mapped format:
The solution’s description is followed by the notice:

*There is an additional possible solution available. To display it press Y, for ‘yes’.*

The system then brings to the screen the textual description of a second possible solution:

*The solution may be a high tech door entrance, with the following features:*
  - vertical glass tower
  - surrounding materials: glass, metal
  - swinging door
  - revolving door (with four leaves)
  - leaf type: framed with one or more light cross panels
  - leaf materials: non-stained glass, metal

The second solution’s description is followed by the graphic illustration of the precedent, also in a bit-mapped format:
Figure 7. High Tech solution.

The system will then bring to the screen the textual description of a new possible solution, as follows:

A new solution may be possible with the following features:
- vertical glass tower.
- lateral cylindrical section column.
- surrounding materials: glass, metal.
- opening mechanism: swinging door, revolving door.
- leaf type: framed.
- steelworks as leaf decoration.
- leaf materials: glass, metal, timber.

Real world applications:

The small domain described earlier provided a good illustration of the internal procedures of the proposed integrated model. However, its reduced number of precedents (9 cases described by 27 features) did not represent a sample large enough for a neural network learning process. Also, the fewer the number of precedents in the knowledge-base the greater the possibility of retrieving unsatisfactory solutions due to the case-based reasoning representation adopted in Cortex and its closest match searching procedure. Its results are of little use in practical design situations.

However, an ‘extended domain’, with a much larger number of precedents (122 cases) and descriptors (80 features), describes the problem ‘door entrance’ in much more detail than in the small domain and provides an environment much closer to a realistic design situation.

For instance, suppose that we are now given the task of designing an entrance door in a non-existing building. We approach the ‘extended domain’ with the following basic constraints: firstly, the materials should be, as much as possible, industrialised and non hand-crafted. Secondly, it is supposed to be a single door and to give access to a
reception desk. Thirdly, an emphasis should be placed on natural heat conservation in the winter.

The system starts by bringing the first question to the screen:

*Does the door leaf have a flat top?*

Since the presence of a flat door top favours industrialisation, we answer ‘yes’ to the question above. The system then brings to the screen the next question:

*Does it give access to an air lock, vestibule or foyer?*

This entrance door is supposed to give access to a reception desk, as mentioned above. In these situations it is usual to have a foyer where people can make enquiries and meet other people. Therefore, we answer ‘yes’. The system then brings the next question to the screen:

*Is it supposed to be the main entrance?*

We know that the entrance being designed is an important one. However, as we are not aware yet of the situation in the remaining of the building, the answer is ‘don’t know’. The system then asks us:

*Is it supposed to have glass in the surroundings of the entrance?*

We interpret the presence of glass, as a dominant surrounding material, as a means of favouring natural heat conservation in the winter. Therefore the answer is ‘yes’. The next question is:

*Is it supposed to have non-stained glass in the leaf?*

Since the presence of non-stained glass in the leaf can also be interpreted as favouring natural heat conservation in the winter, the answer is also ‘yes’. The system then brings to the screen the following question:

*Is it supposed to have metal in the leaf?*

Since metal favours industrialisation, the answer is once again ‘yes’. The system then asks us:

*Is it supposed to have metal in the surroundings of the entrance?*

Once more, the industrialisation factor drives us to answer ‘yes’. The next question is:
Is it supposed to have a Window in both sides?

Since we have decided at an earlier stage that glass was going to be one of the dominant materials in the surrounding of the entrance to provide natural heat conservation, we decide to use a Window in both sides of the door. The answer is therefore ‘yes’. The system then asks the next question:

Is it supposed to allow public access?

Although we are aware of the importance of the door, we do not know yet about the type of flow control for this entrance. The answer is thus a ‘don’t know’. The system then brings to the screen the following questions:

Is it pulled in from the facade?

We did not have a prior opinion on this matter, but we are now inclined to think that the feature above may give some protection from the flow of people in the street and from bad weather. Therefore, the answer to the question above is ‘yes’. The next question is:

Does the entrance have a squared fanlight on its upper part?

Since glass is going to be one of the dominant materials in the surroundings of the door entrance, we think of a squared fanlight as a means of increasing heat loss in the summer. Therefore, the answer is ‘yes’. The system then brings the next question to the screen:

Does the entrance have one double swinging door?

Considering the initial constraints, which call for a single door, we answer ‘no’ to this question. The system then brings the next question to the screen:

Is this entrance supposed to have bricks in its surroundings?

Bricks are nowadays industrialised construction materials. However, they do not favour a mechanised construction process to the same extend of their own manufacturing, particularly due to their dimensions and to the need of manual and careful bricklaying. Therefore, we answer ‘no’ to this question. The system will then bring a solution to the screen:

The most likely solution in the Extended domain is:
The closest match is Case 63, which has the following features:
Flow function: main entrance
Flow control: restrict access
Flow connection: gives access to: air lock, vestibule or foyer
Formal insertion: pulled in from the facade
Door top shape: flat door top
Top complements: squared fanlight
Other complements: walls supporting porch
Lateral complements: a Window in both sides
Surrounding materials: glass
metal
Door operation type: swinging door, one single
Door leaf type: framed with three or more light cross panels
Door leaf materials: non-stained glass
metal
Door handle, if any: round knob or ring handle

The following illustration will come up on the screen:

![Figure 8. Case 63](image)

The system will then propose the following new combination of features:

A new solution may be possible with the following features:
Flow function: main entrance
Flow control: public access
Flow connection: gives access to: air lock, vestibule or foyer
Formal insertion: pulled in from the facade
Door top shape: flat door top
Top complements: squared fanlight
The first solution is retrieved because it is the closest match in the case-base. It thus satisfies all the conditions verified by the questions presented to the user. The second solution, which is a result of the neural network computation, also satisfies all the conditions verified by the questions above.

However, it provides some more interesting features regarding natural heat conservation in winter, than the solution retrieved from the case-base. These features are: ‘a vertical glass tower’ (instead of ‘walls supporting porch’), only ‘glass’ and ‘metal’ among the surrounding materials (no ‘smooth stone’), and the addition of a ‘revolving door’ (instead of just a ‘swinging door: one single’).

There may be several possible graphic interpretations of the new solution, but the textual description above already provides some illustration of how the proposed model may augment the designer’s creativity. In the example above the ‘vertical glass tower’ and the ‘revolving door’ represent features not thought by the designer at the outset of the design task. Yet, they comply with the initial set of constraints, which were that the materials should be, as much as possible, industrialised and non hand-crafted, and an emphasis should be placed on natural heat conservation in the winter.

Several domains could be implemented in the proposed model representing different levels of abstraction for a particular design task. At the end of each section we may accept the proposed solution and move towards its detailing, or decide to go back to the beginning and start another section at the same level of abstraction.

7. Conclusions:

About 15 domains have been already implemented in the proposed model and they deal with different architectural problems (see Silva, 1995, for a description of the extended domain and experimental data). Each of those domains is being gradually linked to libraries of 3D architectural components. Therefore, in addition the textual descriptions of the new solutions an easy transition to graphic representations is being built.
The main benefits of this hybrid model are: firstly, the system always reaches a stable solution or the most possible stable solution. There is no risk that a search will be terminated at an unstable state. Secondly, there is no risk of the user getting lost in the search for a stable solution, because all the basic choices are made prior to their entering in the neural network, the search is automated and the system itself keeps track of previous actions. Thirdly, there is no risk of getting into an infinite loop since the system filters features by guiding the user through the most promising path and controlling and terminating deadlock situations.

This integrated environment automates the process of searching for stable solution by emulating the user’s actions through the semi-recurrent network described earlier.

At last, the system offers a plain English interactive interface throughout the process of solution search. In addition, links to 3D libraries are being developed, which will make neural network output more useful and will enhance 3D modelling as an intelligent activity.

The result of this approach will encourage designers to use intelligent systems in a more enjoyable way and to explore design alternatives towards creativity within meaningful interface and environment.

References:


