Building an Adaptive Evaluation System: A Design Education Application

Shang-Yuan Chen[1] and Ming-jen Cheng[2]

[1]National Cheng Kung University, shangyuanc@pchome.com.tw
[2]Feng-Chia University, mjcheng@fcu.edu.tw

ABSTRACT

This paper relies on observation and analysis of a collaborative international digital design workshop to propose a situation-mapped neural network model functioning in a virtual environment. This model can be used to guide mutual validation and revision with the underlying design cognition model. We expect that the results of this work can lead to the establishment of a scoring mechanism that can "adapt" to the difficulty of assigned problems, and the mechanism can be used to assess students' progress. The mechanism is based on the neural network's calculation of the difference between the "output value" and the "desired value." Such a student progress-based assessment mechanism can encourage students to select relatively difficult design problems and thereby promote more design originality.

Keywords: neural network, adaptive, generalization, digital design education, design cognition

1. BACKGROUND AND GOALS

The nature of pedagogy is to strive for a well-functioning "adaptive" system. While there should be positive, flexible interaction between the content of instruction and the quality and quantity of learning within such a system, design instruction is full of inherent indeterminacy and complexity. Design workshops, which are characterized by short-term instruction, are becoming an increasing popular form of international exchange activity. In view of the nature of design instruction, are the design results of such design workshops truly able to reflect students' progress? In addition, how should design workshops be run, and how should they be assessed? It is absolutely essential that we construct and analyze a model of design instruction if we wish to answer these questions. Because neural systems are highly "adaptive," they are able to extract, interpret, and use contextual information, adapt their functions, and achieve an optimal correspondence between "contextual change" and "needs" (Principe, 2000), e.g. (Fig. 1). This paper therefore seeks to explore the following four questions: (1) Can the situations of real world instruction be simulated in a neural algorithmic system? (2) If yes, will an algorithmic system simulating design instruction therefore possess the ability to analyze, predict, and assess? (3) Can students' progress be measured? If student progress can be used as the basis for evaluation, this will encourage students to take on more challenging problems and stimulate more digital design originality. (4) After the algorithmic system representing the instructional process is subjected to testing, will it be possible to improve the system?

This research first proposes a hypothesis on the basis of a retrospective of the literature, and then participates in observation of on-site implementation of experimental design instruction. The stages of this research include (1) establishment of a theoretical framework, (2) recording of operating processes at a digital design workshop, (3) establishment a neural network-based instructional model, (4) validation of theory, (5) assessment of differences between theory and practice, and presentation of conclusions and possibilities for future research.

2. THEORY AND METHOD

Artificial intelligence experts in the field of design have proposed a long series of "cognitive models" attempting to explain designers' design behavior. These models may also be used to guide the development of instructional curricula or to develop instructional platforms or computer-aided design tools. A design cycle can be seen from one angle as a "problem-solving information transmission process" (Newell, Shaw and Simon 1957). A design cycle therefore needs clearly specified steps and well defined plans if it is to avoid becoming an endlessly sprawling decision tree. A revised problem-solving process should therefore be linked to a "decision-making circle" (Asimow, 1962). Nevertheless, the foregoing type of model suffers from some inherent limitations. While the model is particularly applicable to well-defined problems, most problems in design are ill-defined (Rowe, 1987). In particular, design "creativity" is often felt to be outside the boundaries of normal practices. Since neural network models are adept at handling ill-defined, unstructured problems, and possess scientific algorithms and assessment indicators, they represent a better approach for developing "design cognition."

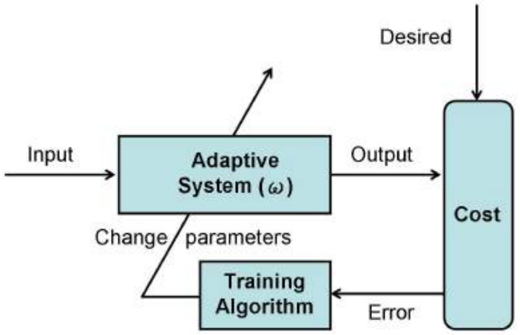


Fig. 1. Adaptive System, (Principe, 2000).

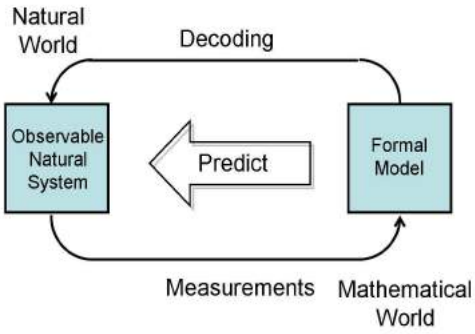


Fig. 2. Natural system and formal models, (Principe, 2000).

From another aspect, the results of design creativity are often influenced by the difficulty of the topic. Since, within the same period of time, it is easier to get points working on an easy topic than on a hard topic, the number of points received in a design competition with different problems (such as a thesis design) certainly cannot accurately reflect a student's actual progress. Design educators must dethrone the myth that choosing an easy topic is the way to get easy points if they want to encourage more design originality.

The content of this paper is based on observations of a joint international digital design workshop—the Archi, FCU & Bartlett, UCL, Digital Architecture Workshop (Su, 2005). We propose a neural network model operating in a virtual environment and conforming to circumstances in accordance with the workshop's instructional framework, features, and requirements. We then validate and revise the neural network-based instructional model via on-site observation and participation in the instructional process, e.g. (Fig. 2). The goal of this research is to verify that a neural network-based instructional model can improve students' levels within an extremely short period of time. In addition, in a virtual environment, the model can simulate the design process via neural network software. The network acquires an inference-based predictive ability as it learns. Neural training and testing allows it to derive assessment indicators of student progress, and these indicators can be used to establish a scoring mechanism that "adapts" to the difficulty of a topic, and can encourage originality in digital design.

3. RESEARCH PROCESS AND RESULTS

3.1 The Archi, FCU & Bartlett, UCL, Digital Architecture Workshop

International design workshops give students or academics an opportunity to share ideas and achieve progress in design learning. This digital architecture workshop involving Feng Chia University and the Bartlett School of Architecture of the University College London (UCL) featured an eight-day digital design instructional demonstration given by the two Bartlett School lecturers Marcos Cruz and Mariano (Marjan) Colletti, (Note 1). In spite of the short length of this activity, it elicited exceptionally high expectations. The workshop required students to establish a division of labor and quickly learn new things, while also emphasizing cooperation and the design works' rapid convergence on a certain standard, e.g. (Fig. 3).

The intent of this eight-day learning activity (March 2-9, 2005) was to let the participants first "produce their own separate works" and then "integrate them into buildings" that are technological "context-aware entities," and also probe the feasibility of visual/dynamic "bionics.", or zoomorphic form", (Hugh, 2004). The activity consisted of three phases. During the first phase, 26 qualified and selected students formed two groups: the "inhabitable wall" and "sp-line animal" groups. The inhabitable wall group attempted to investigate the relationship between the thickness and body of walls, and the sp-line animal group focused on the exploration of bionic forms. Six elements (consisting of architectural structures and infrastructure) were identified during the second phase in accordance with the attributes of the students' works; these elements were external wall (ex-wall), internal wall (in-wall), canopy, furniture, sensor and cable, and animation. The elements were then "integrated." The integrated designs remained in virtual space during the third phase, but the computer-aid drawing files were used to control CNC (computer numerical control) machinery cutting and bending real materials (wood and metal, etc.) to produce components. The components were then assembled in a real world exhibition room. Accompanying the assemblage, all draft models and the animation recording the design process were used to display instructional results (Chen, 2005), e.g. (Fig. 4).

http://www.arch.fcu.edu.tw
2005逢甲暨巴特雷數位建築設計展
Archi, FCU & Bartlett, UCL Digital Architecture Exhibitions
M&M
arcos Cruz
arjan Colletti

Fig. 3. Poster for the Archi, FCU & Bartlett, UCL, Digital Architecture Workshop.

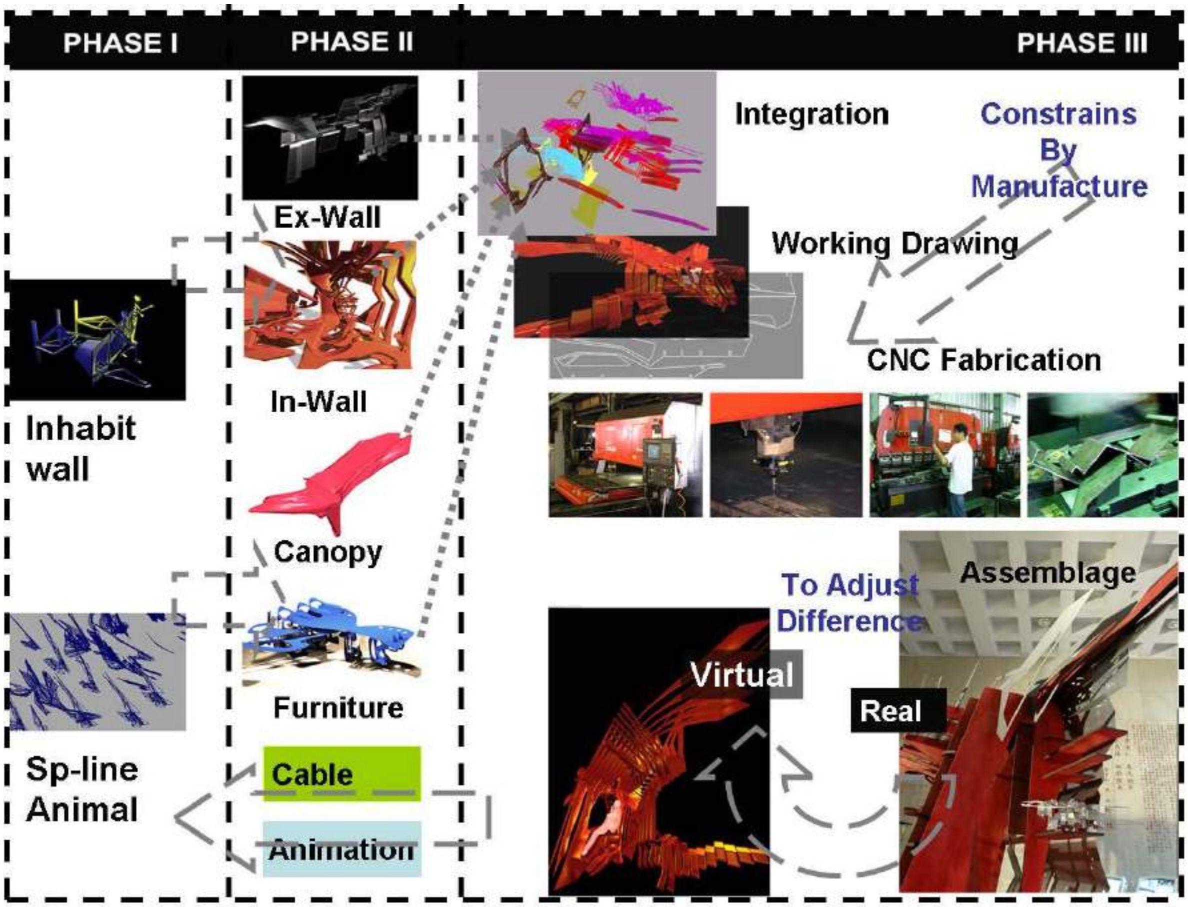


Fig. 4. Process Flowchart of the Archi, FCU & Bartlett, UCL, Digital Architecture Workshop.

3.2 Neural Network Model Reflecting Construction Circumstances

An artificial neural network uses a computer to simulate the neural activity of living organisms. Neural network systems employ parallel and distributed operating elements – "neurons" – and rely on the connections between those elements to perform operations. Neural networks are good at simultaneously processing many batches of data. Because neural networks adjust the weights of neural connections to converge on a desired output value, there is no need to make any prior assumptions about the relationship between the input data and output value. A neural network can analyze the mapping between the input data and output value as long as it has enough samples cases to work with. This type of "adaptive" computing system is especially suitable for making decisions concerning unstructured problems. A neural network is able to learn, recall, and make inferences from input environmental signals (Chang, 2004). Neural networks are typically designed according to the following principles: Network architecture is first determined in accordance with the complexity of the incident to be processed. Network architecture parameters include the number of neurons, number of layers, and whether the network employs a forward-propagation or feedback method. The next step is to judge whether the network should employ supervised or unsupervised learning on the basis of whether a desired output value exists. The final step is to select an appropriate learning algorithm in accordance with the characteristics of problems to be processed (Girosi, 1995). We established a neural network to simulate the instructional process and characteristics of the digital design workshop. This network took students' works to be neurons, and used instructors' evaluations as an activation function. The successful works gradually converged as they complied with the factory's manufacturing requirements. The network was constructed according to the following steps:

3.2.1 Determination of network model

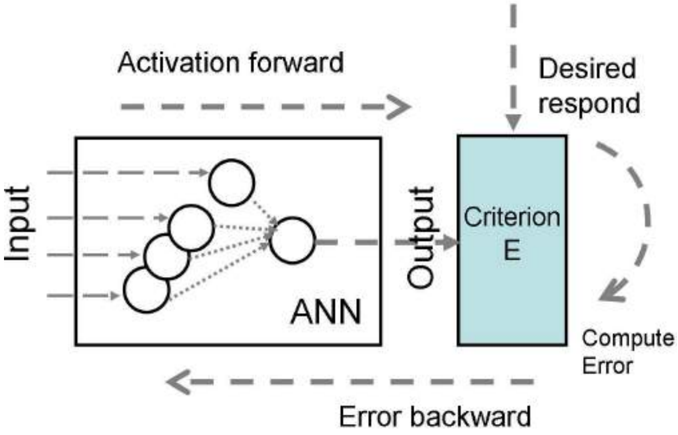


Fig. 5. Chaining of operations in a back propagation algorithm, (Modified from Principe, 2000).

The parallel processing of designs was adopted at the beginning in order to obtain a large quantity of instructional results within a very short period of time. Furthermore, in consideration of the very limited budget for constructing physical entities and the need to secure students' cooperation, we first encoded the instructional results, i.e. the students' works, selected elements from among them constituting different architectural components in accordance with the designers' needs, integrated them as a design work, and then selected one of the integrated works, followed by production of drawings and construction. The input end consists of multidimensional vectors, while the output end consists of a one-dimensional vector. The architecture takes the form of a "multi-layer back propagation network,", e.g. (Fig. 5).

3.2.2 Determination of learning attributes

We adopted "supervised" learning for fast convergence. Supervised learning means that the network weights are adjusted in accordance with the "teacher's" desired value (the desired value in this example represents adjustment of weights transmitted through the network in accordance with the factory's production restrictions and instructor's requirements of the works). Adjustment of weights is performed until the difference between the output value and the desired value is less than a certain “threshold value.”

3.2.3 Selection of an algorithm

We selected commonly used least-mean square (LMS) algorithm on the basis of the network's back propagation feedback method and the supervised learning attribute. The LMS algorithm adjusts weight W_{ji} , e.g. Eqn. (5), in accordance with the steepest descent method in order to find the mean-square-error (MSE), E^{Min}, e.g. Eqn. (3). The MSE is also known as the "cost function," and is an indicator of the error between a neuron's output value Y_k , e.g. Eqn. (1), and desired value d_k . The weight adjustment rate is termed the learning rate, □, e.g. Eqn. (4). The □ value affects the rate and stability of learning. The following are the major functions and formulas of the LMS algorithm (Principe, 2000):

The input value of the jth neuron in the nth layer is a nonlinear function of the (n-1)th layer neuron's output value

- Output value of (n-1)th layer:

$$Y_j^n = f\left(net_j^n\right) \tag{1}$$

- Summation function of the (n-1)th layer:

$$net_j^n = \sum_i w_{ji}^n y_i^{n-1} - b_j^n \tag{2}$$

- Mean-square error function:

$$E = \left(\frac{1}{2}\right)\sum_K \left(d_k - y_k\right)^2 \tag{3}$$

- Weight adjustment value:

$$\Delta w_{ji} = -\eta \frac{\partial E}{\partial w_{ji}} \tag{4}$$

- Weight after adjustment:

$$w_{ji}(p) = w_{ji}(p-1) + \Delta w_{ji} \tag{5}$$

3.2.4 Principles of data clustering and generalization principles

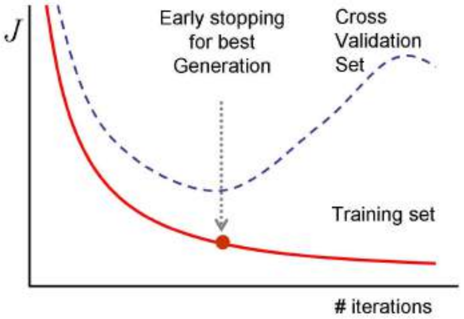


Fig. 6. Cross validation, Early stopping (Principe, 2000).

The collected data must be clustered. The data is first clustered into two sets—the "training set" and the "testing set." We also had to pay attention to the principle of generalization, which means that approximately 10%~20% of the back propagation network's training data must be assigned to a "cross validation set." The training set will stop training as soon as the cross validation set error starts increasing. This prevents the network from increasing the unknown testing set error due to over-fitting of the training set (Hagan, 1996), (Haykin, 1999) , e.g. (Fig. 6).

3.3 Encoding and Neural Network Model Testing

We had to encode the data in order to perform simulation and verification using the neural network software. The students were separated into two discussion groups (the inhabitable wall and sp-line animal group) during the first phase of the design workshop We encoded the nine works that the instructors selected from each group with two-digit numbers; here a prefix "1" indicates the inhabitable wall group (example: 11, 12, 13, etc.) and a prefix "2" indicates the sp-line animal group (example: 21, 22, 23, etc.), e.g. (Tab. 1).

Inhabit Wall	11 	12 	13 	14 	15 	16 	17 	18 	19
Sp-line Wall	21 	22 	23 	24 	25 	26 	27 	28 	29

Tab. 1. Data Encoding.

3.3.1 Training and establishment of testing set

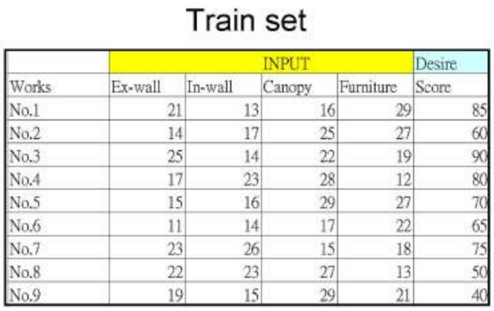
Fig. 7. Open review and critique.

During the second phase, on each occasion the students carefully selected four of the foregoing 18 works to serve as architectural elements, such as the external wall, internal wall, canopy, and furniture, influencing "style," and integrated those elements as a single work. The integrated works were arranged in sequential order, the codes of the four architectural elements were recorded, and the integrated works were subjected to open critique, e.g. (Fig. 7). The four codes were then used as the "input end" of the training data, while the scores of the integrated works served as the "desired values." As shown in Table 2, winning work 3 was composed of 25, 14, 22, and 19, and had a score of 90. The data was grouped as two sets: the training set consisting of data randomly selected according to score, e.g. (Table 3), and the testing set, e.g. (Table 4). The training set and testing set each accounted for 50% of the data times in this example.

No.3	25 	14 	22 	19 	90

Tab. 2. Work No. 3 was composed of 25, 14, 22, and 19, and had a score of 90.

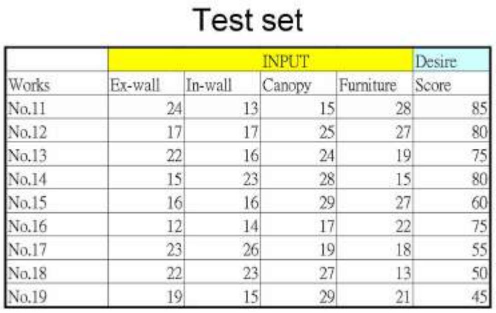
We used the neural network software Neuro-Solutions to simulate training. We first selected a network model, namely a multi-layer back propagation network, then input the training set data, set the cross validation set perception, selected an algorithm (LMS, MSE), selected an activation function (tanh), and adjusted the learning rate. Training ended when the cross validation error rose to the early stop point (at MSE = 0.06 in this example) based on generalization, e.g. (Fig. 8).



Train set

	INPUT				Desire
Works	Ex-wall	In-wall	Canopy	Furniture	Score
No.1	21	13	16	29	85
No.2	14	17	25	27	60
No.3	25	14	22	19	90
No.4	17	23	28	12	80
No.5	15	16	29	27	70
No.6	11	14	17	22	65
No.7	23	26	15	18	75
No.8	22	23	27	13	50
No.9	19	15	29	21	40

Tab. 3

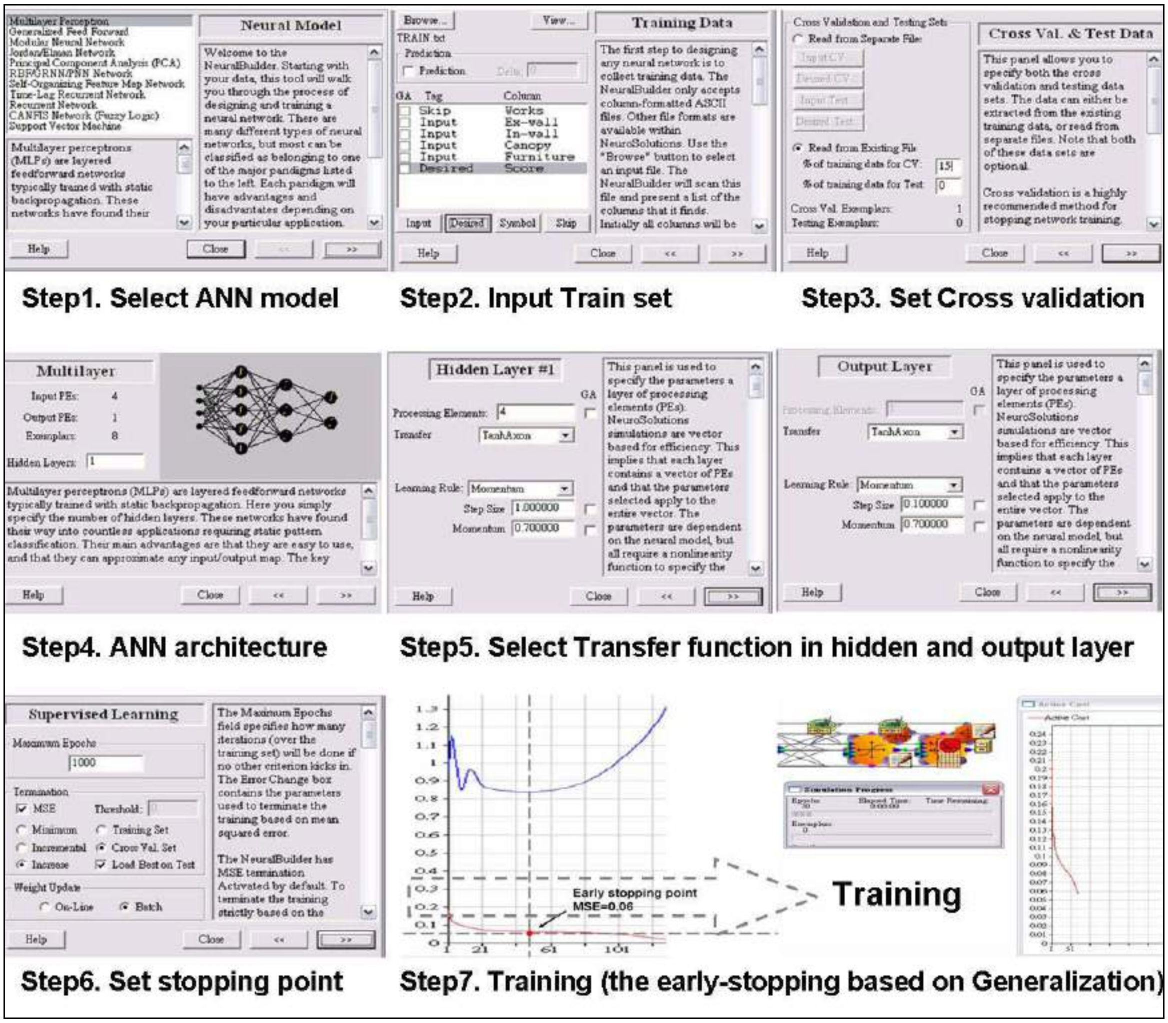


Test set

	INPUT				Desire
Works	Ex-wall	In-wall	Canopy	Furniture	Score
No.11	24	13	15	28	85
No.12	17	17	25	27	80
No.13	22	16	24	19	75
No.14	15	23	28	15	80
No.15	16	16	29	27	60
No.16	12	14	17	22	75
No.17	23	26	19	18	55
No.18	22	23	27	13	50
No.19	19	15	29	21	45

Tab. 4

3.3.2 Training set and the end of training



Multilayer Perceptron
Generalized Feed Forward
Modular Neural Network
Jordan/Elman Network
Principal Component Analysis (PCA)
RBF/GRNN/PNN Network
Self-Organizing Feature Map Network
Time-Lag Recurrent Network
Recurrent Network
CANFIS Network (Fuzzy Logic)
Support Vector Machine

Multilayer perceptrons (MLPs) are layered feedforward networks typically trained with static backpropagation. These networks have found their

Neural Model

Welcome to the NeuralBuilder. Starting with your data, this tool will walk you through the process of designing and training a neural network. There are many different types of neural networks, but most can be classified as belonging to one of the major paradigms listed to the left. Each paradigm will have advantages and disadvantates depending on your particular application.

Step1. Select ANN model

TRAIN.txt
Prediction
□ Prediction Delta: 0

GA	Tag	Column
□	Skip	Works
□	Input	Ex-wall
□	Input	In-wall
□	Input	Canopy
□	Input	Furniture
□	Desired	Score

Training Data

The first step to designing any neural network is to collect training data. The NeuralBuilder only accepts column-formatted ASCII files. Other file formats are available within NeuroSolutions. Use the "Browse" button to select an input file. The NeuralBuilder will scan this file and present a list of the columns that it finds. Initially all columns will be

Step2. Input Train set

Cross Validation and Testing Sets
○ Read from Separate File:
● Read from Existing File
% of training data for CV: 15
% of training data for Test: 0
Cross Val. Exemplars: 1
Testing Exemplars: 0

Cross Val. & Test Data

This panel allows you to specify both the cross validation and testing data sets. The data can either be extracted from the existing training data, or read from separate files. Note that both of these data sets are optional.

Cross validation is a highly recommended method for stopping network training.

Step3. Set Cross validation

Multilayer
Input PEs: 4
Output PEs: 1
Exemplars: 8
Hidden Layers: 1

Multilayer perceptrons (MLPs) are layered feedforward networks typically trained with static backpropagation. Here you simply specify the number of hidden layers. These networks have found their way into countless applications requiring static pattern classification. Their main advantages are that they are easy to use, and that they can approximate any input/output map. The key

Step4. ANN architecture

Hidden Layer #1
Processing Elements: 4
Transfer: TanhAxon
Learning Rule: Momentum
Step Size: 1.000000
Momentum: 0.700000

This panel is used to specify the parameters a layer of processing elements (PEs). NeuroSolutions simulations are vector based for efficiency. This implies that each layer contains a vector of PEs and that the parameters selected apply to the entire vector. The parameters are dependent on the neural model, but all require a nonlinearity function to specify the

Output Layer
Transfer: TanhAxon
Learning Rule: Momentum
Step Size: 0.100000
Momentum: 0.700000

This panel is used to specify the parameters a layer of processing elements (PEs). NeuroSolutions simulations are vector based for efficiency. This implies that each layer contains a vector of PEs and that the parameters selected apply to the entire vector. The parameters are dependent on the neural model, but all require a nonlinearity function to specify the

Step5. Select Transfer function in hidden and output layer

Supervised Learning
Maximum Epochs: 1000
Termination
☑ MSE Threshold:
○ Minimum ○ Training Set
○ Incremental ● Cross Val. Set
● Increase ☑ Load Best on Test
Weight Update
○ On-Line ● Batch

The Maximum Epochs field specifies how many iterations (over the training set) will be done if no other criterion kicks in. The Error Change box contains the parameters used to terminate the training based on mean squared error.

The NeuralBuilder has MSE termination Activated by default. To terminate the training strictly based on the

Step6. Set stopping point

Early stopping point
MSE=0.06

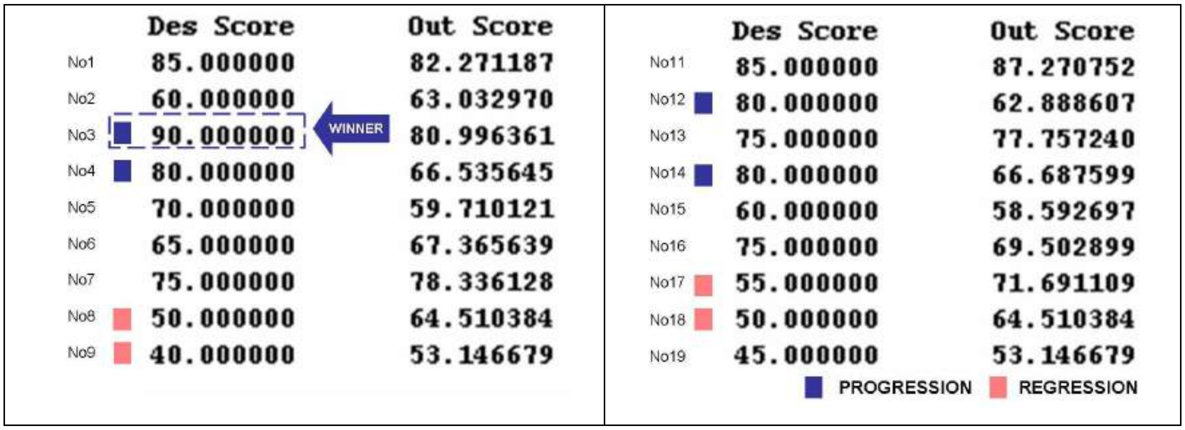
Training

Step7. Training (the early-stopping based on Generalization)

Fig. 8. Training process flowchart for Neuro-Solutions software.

3.3.3 Results

In theory, the training set output value will be extremely close to the desired value once training has been completed, and this indicates that the simulated design thinking process now possesses inference ability. The new output value should therefore approach the desired value when the "testing set" is used to confirm the training results. Actually, though, in line with the principle of generalization, network training ends earlier so as to avoid the over-fitting of the training set data. As a result, although the trend predicted by the network using either the training data or testing data may largely conform to the expected trend, the actual value may not be exactly the same as the desired value. We may assume that if the output value is significantly higher than the desired value, the score inferred by the network is likewise significantly higher than the actual scores of the students' works, which suggests that the students must work harder. Conversely, when the output value is significantly lower than the desired value, we may conclude that the students' level has surpassed the score inferred by the network, e.g. (Table 5). In accordance with these observations, student achievements under this instructional framework are influenced by two main types of factors, one of which being the students' talent and effort, the other being the difficulty of integrating the selected architectural elements. The former is implicit and difficult to measure, while the latter is explicit. Design results can be obtained and expressed as the output value of a neural network. This output value can serve as a standard of the difficulty of integrating architectural elements. In addition, the difference between the output score of the neural network's "machine calculations" and the desired score obtained by "human cognition" provides a yardstick for determining whether a student is making progress. It is worth noting that these findings indicate that design learning is not a totally goal-oriented process. Design learning can be considered a process of dynamically adjusting weights to achieve a corresponding "desired value." Table 5 shows that the most progress occurs when the absolute value of the difference is large. Although work 3 earned the highest score, work 12 displayed the most progress due to the difficulty of the topic. Conversely, instructors need to pay more attention to the relatively poor learning displayed by works 8, 9, 17, and 18.



		Des Score	Out Score
No1		85.000000	82.271187
No2		60.000000	63.032970
No3	🟦	90.000000	80.996361
No4	🟦	80.000000	66.535645
No5		70.000000	59.710121
No6		65.000000	67.365639
No7		75.000000	78.336128
No8	🟥	50.000000	64.510384
No9	🟥	40.000000	53.146679
No11		85.000000	87.270752
No12	🟦	80.000000	62.888607
No13		75.000000	77.757240
No14	🟦	80.000000	66.687599
No15		60.000000	58.592697
No16		75.000000	69.502899
No17	🟥	55.000000	71.691109
No18	🟥	50.000000	64.510384
No19		45.000000	53.146679

Tab. 5. Marks indicates works for which the difference between the desired and output scores was greater than 10 points.

4. CONCLUSIONS AND RECOMMENDATIONS

This research uses a neural network system to simulate the design process, and determine students' relative progress on the basis of the difference between the network's output value and the desired value. The result of this research was the establishment of a scoring mechanism that can "adapt" to the difficulty of assigned problems, and thereby encourage students to take on challenging topics and stimulate design creativity. The principles discussed in this paper can also be applied to other design education evaluation cases.

Apart from this, the design problems assigned at the Archi, FCU & Bartlett, UCL, Digital Architecture Workshop successfully emphasized agile control of the design process. This enabled the relationship between the input data (or design) and the output results to be quickly mapped. The neural network architecture enabled the close collaboration between the instructors and students to achieve rapid convergence on outstanding design results. This type of topic model is different from the conventional one in which input conditions are constrained by rigid rules so that design results satisfy the requirements of those rules. The inference process is very time-consuming in the conventional model, and inexperienced designers tend to disperse the topic, ruling out convergence. Although this digital design workshop was very short, its design results earned considerable praise. Exhibitions of works from the workshop were held March 12~18, 2005 at Feng Chia University's Ren-Yan Exhibition Hall, e.g. (Fig. 8), and starting September 9, 2005 at the Hamburg Culture Policy Research Institute, Germany. An exhibition is also planned for the Bartlett School of Architecture in London after the conclusion of the German exhibition.

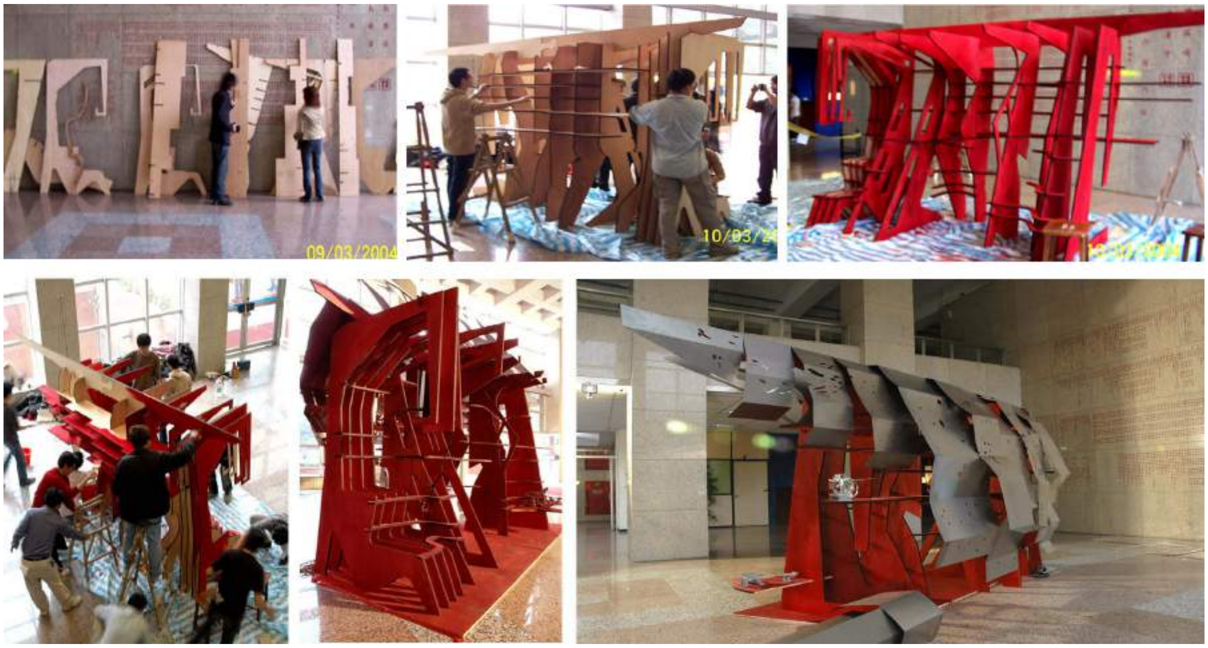


Fig. 8. Installation and exhibition in Feng Chia University's Ren-Yan Exhibition Hall.

The significance of this research lies in its construction of an appropriate cognitive model explaining design instruction, and finding an effective assessment approach, or revising the original cognitive model, in order to achieve even better instructional effectiveness, facilitating the production of optimal designs. Of course the Archi, FCU & Bartlett, UCL, Digital Architecture Workshop may be something of a special case among the mass of joint international design activities, and its instructional framework was not necessarily comparable to those employed by other design workshops. Design is fundamentally the synthesis of many complex factors and considerations under numerous constraining rules and conditions, while also striving for "creativity." We can assess the degree to which design results conform to rules or conditions, but creativity that appeals to people's feelings is hard to evaluate. The use of adaptive weights by a neural network offers a scientific solution to this type of unstructured mapping problem.

Many types of algorithms have been developed in response to different problems since the first use of neural networks. Nevertheless, can feasible neural network theories be appropriately used to improve design instruction? And can they be used to construct computer-aided design systems? For instance, it might be more appropriate to use a dynamic time-delay neural network if the evaluation of a design work is influenced by the evaluation of the previous work. Or a computer-aided design system constructed using a genetic algorithm could employ crossovers and mutations to eliminate poor architectural elements at an early stage in order to ensure or improve design quality. We believe that, in the field of design, neural networks should not be used exclusively in design education, but also in the development of computer-aided design systems.

Note 1:

Marcos Cruz and Mariano (Marjan) Colletti are partners in both education and practical work. Both have been in the doctoral program at the Bartlett School of Architecture, University College London (UCL), since 2000. They helped Peter Cook and Colin Fournier design the award-winning Kunsthaus in Graz, Austria in 2000. Completed in 2003, the Kunsthaus possesses a biomorphic style and envelope; the points of light on the surface of the building can display constantly changing images (Cook, 2000). This programmable, "context-aware" structure reveals a glimpse of the design creativity and experimental instructional thinking of Cruz and Colletti.

5. ACKNOWLEDGEMENTS

Directors of the workshop, Associate Prof. Su Chih-feng and Hsieh Pei-ni, Department of Architecture, Feng Chia University; lecturers—Marcos Cruz and Mariano (Marjan) Colletti; participants—26 students. Special thanks go to Chang Wen-ting, Tsao Fang-wei (Feng Chia University), and Wei Tzu-chun (National Cheng Kung University) for providing the reference materials and workshop records needed to enable the completion of this paper.

6. REFERENCES

[1] Asimow, M., *Introduction to design*, Englewood Cliffs, New Jersey, Prentice-Hall, 1962.

[2] Chang, F. C., Chang, L. C., Huang, H. L., *Artificial neural network theory and practical application*, Tung- Hwa bookstore, pp 1-11, pp120-135, 2004.

[3] Chen, S. Y., Multi-media application in Digital Design - Creations in FCU & Bartlett, and UCL digital architecture workshop, Taiwan Architecture, No. 122, 2005, pp 80-83.

[4] Cook, P. C., Fournier, http://www.danda.be/outdata/?dreview=130&category=, 2000

[5] Girosi, F., Jones, M. and Poggio, T., Regularization theory and Neural Networks Architecture, *Neural Computation*, Vol. 7, 1995, pp 219-269.

[6] Hagan, M. T., Demuth, H. B. and Beale M., *Neural Network Design*, Thomson Publishing Inc, 1996, pp 11-23.

[7] Haykin, S., *Neural Networks: A Comprehensive Foundation*, Printice Hall, 1999, pp 156-256.

[8] Hugh, A. W., *Zoomorphic: New Animal Architecture*, Laurence king publishing Ltd., London. 2004.

[9] Mari, K. A., Context-Aware Applications Survey, Department of Computer Science Helsinki University of Technology, 2000.

[10] Newell, A., Shaw, J. C. and Simon, H. A., Elements of a Theory of Problem Solving, Rand Corporation Report pp-971, March, 1957.

[11] Principe, J. C., Euliano, N. R. and Lefebvre, W. C., *Neural and Adaptive Systems: Fundamentals Through Simulations*, John Wiley & Sons, 2000, pp 1-18, pp 150.

[12] Rowe, P. G., *Design thinking*, MIT press, 1987.

[13] Su, C.-F., Sp-line wall, Taiwan Architect, No. 367, 2005, pp 86~89.