Demonstration of Cognitive Modeling in Categorization: Fitting Two Neural Network Models to the Data from Yang and Lewandowsky (2003)

認知模擬在類別學習上的應用：以Yang與Lewandowsky (2003)之研究為例

doi:10.6129/CJP.2007.4903.04

中華心理學刊, 49(3), 2007

Chinese Journal of Psychology, 49(3), 2007

作者/Author： 楊立行(Lee-Xieng Yang)

頁數/Page： 285-300

出版日期/Publication Date：2007/09

中 華 心 理 學 刊
民96，49卷，3期，285-300

Chinese Journal of Psychology
2007, Vol. 49, No. 3, 285-300

Demonstration of Cognitive Modeling in Categorization: Fitting two neural network models to the data from Yang and Lewandowsky (2003)

Lee-Xieng Yang

Institute of Cognitive Science, National Cheng-Kung University

MS No.: 070430; Received: April 30, 2007; Revised: July 25, 2007; Accepted: July 28, 2007
Correspondence Author: Lee-Xieng Yang, Institute of Cognitive Science, National Cheng-Kung University, No.1, University Road, Tainan City 701, Taiwan（E-mail: lxyang@mail.ncku.edu.tw）

In investigating human mental processes and mental representations, a cognitive model represents a theoretical view, provides explanations to the observed phenomena and makes predictions about an unknown future. When evaluating how well a theory can account for the phenomenon of interest, modeling is a powerful research tool. However, local (Taiwanese) psychology students have limited exposure to what cognitive modelling is, how to do implement cognitive models, and why cognitive modelling is important. This is partly due to a lack of university courses that teach cognitive modelling and partly due to the demands that modelling places on one's skills. The purpose of this article is to provide a conceptual guideline of how to do modeling, by fitting two neural network models - ALCOVE and ATRIUM to the data from the study of Yang and Lewandowsky (2003), which tested the theoretical concept of knowledge partitioning in categorization. The modeling results show that ATRIUM outperforms ALCOVE in accounting for the knowledge partitioning results. Some relevant theoretical-level discussions, such as the heterogeneity of categorization, are also included.

Keywords: ***cognitive modeling, cateogrization, neural network***

In cognitive psychology, it is not new to propose models to explain and predict human behavior. These models, while not always computational, tend to describe human behavior with the language of mathematics. The local (Taiwanese) psychology students have limited exposure to what cognitive modelling is, how to implement cognitive models, and why cognitive modelling is important. This is partly because modeling courses are not provided in university for psychology stu-

Acknowledgements

I especially would like to thank Daniel Little for his valuable comments and his help on editing this article. Also, I would like to thank the two anonymous reviewers for their valuable comments.

dents and partly because modeling requires additional skills, such as computer programming, that might decrease the students' motivation to undertake cognitive modeling. The present article seeks to provide (1) a general introduction to cognitive models including the neural network models in categorization, (2) demonstration of the implementation of models, and (3) evaluation of and comparison between the models with the empirical data. This article proceeds as follow. First, the characteristics of models are introduced. Second, two specific neural network models in categorization as well as a general scheme of doing modeling are outlined. Third, the results fitting the two models to empirical data collected by Yang and Lewandowsky (2003) are provided as a demonstration of model evaluation.

Introduction to Cognitive Models

In the past decades, cognitive psychologists have proposed a large variety of models to account for the difference aspects of human behavior. A model represents the theoretical idea about the process and/or the representation underlying the observed behavior. For instance, an early memory model separates human memory into two parts - primary memory and secondary memory, and the information in the primary memory can be transferred to secondary memory by rehearsal (Waugh & Norman, 1965). This model clearly describes the inner structure of memory and proposes a process for information transformation. An alternative memory model is the LOP (Level-Of-Processing) approach, which captures the basic idea that the deeper an item is processed, the better it will be recalled (Craik & Lockhart, 1972). Notably, the LOP account represents the memory trace on a continuous scale rather than a dichotomous scale which is used in the Primary-Secondary (or STM-LTM) account. It is also important to note that a model is nothing but a window for us to understand the target behavior of interest. Different models provide different perspectives to realize particular aspects of cognition; however, none of them can totally capture the many aspects of real human behavior. Why then do we need models? There are at least two reasons for developing models. First, a model provides a platform to understand why people perform a cognitive task in a particular way. Although we can collect many physical and behavioral data using technologically advanced instruments (e.g., MRI) to dig out more information underneath the behavior, we still need to organize these data into a coherent system to help us to understand the process we are interested in. After all, knowing why is a fundamental need of human beings as well as the goal of scientists. Second, with a model, it becomes possible to reproduce human intelligence. If we already know the elements and the processes that humans use to accomplish a particular task, then we must be able to build a machine be capable of performing the same behavior as humans, which would facilitate our life and civilization in the future.

Verbal vs. Computational Models

We need to know how well a model can account for observed data; otherwise, a model is just like a religious belief not a falsifiable scientific proposition. A normal method to evaluate a model's performance is to see whether participants behave as predicted by the model. A straightforward way to test the model's predictions is to conduct an experiment and see if the predicted effect occurs or not. Furthermore, quantitative evaluation of a model's fit to the data is even better. That is, we want to know not only whether the predicted effect occurs, but also how large is the discrepancy between the human data and the model's predictions. However, quantitative evaluation becomes relatively difficult to achieve for verbal models, such as the memory models instantiated previously. This is because verbal models do not provide the mathematical functions which afford the model the ability to make quantitative predictions. By contrast, in a computational model, several mathematical functions are inter-connected with the output of one function forming the input

of another function and with all of the functions working together to generate the final output. Thus, when an input is received by the computational model, the signal is passed through and conveyed by those functions until the final response is made. With the functions representing the mental processes and the format of the signal in the functions as the mental representation, the computational model has more rigorous constraints from psychology and mathematics and can tell us more information about how a behavior is performed, hence increasing the validity of the model beyond that of a verbal model.

Of course, much of the time, evaluating the functions of a computational model by hand or by spread sheet (e.g., EXCEL) is very time-consuming and error prone. It is strongly suggested to use computer software or any computer language to implement a computational model. How well the model accounts for a behavior depends on how similar the model's predictions are to the empirical data. Modeling with a computer can help us to explore novel ideas or complex models and provide us a chance to find out the relationships between superficially unrelated phenomena (Lewandowsky, 1993). For the sake of demonstrating how these aims can be achieved, I am going to fit two models to the data of a category-learning experiment reported by Yang and Lewandowsky (2003).

There are many sorts of computational models, yet, in order to make a fair comparison between model fits to the experimental data of Yang and Lewandowsky (2003), two neural network models in categorization (ALCOVE; Kruschke, 1992; ATRIUM; Erickson & Kruschke, 1998) are chosen as examples of the computational models. I will briefly introduce the basic structure and the learning in a neural network and then introduce these two models in more detail.

Neural Network Model

A neural network is created by establishing a number of inter-connected nodes which imitate the functions of the real neurons to achieve some goal (see Anderson, 1995). The most attractive characteristic of a neural network is that it can learn and acquire knowledge (or form memory) from experiences, without being fed in advance a pre-established database as done for the traditional AI.

According to the learning type, the neural network can be classified to two types: supervised learning and unsupervised learning networks. The supervised learning network normally has an input and an output layer. In the normal case, there are connections between layers but no connections within a layer. The supervised neural network processes an input through to the output layer, and activation of the output layer is compared to the target activation; the difference between output activation and target activation becomes the error signal. The supervised learning network uses the error signal to adjust the associative weights between layers in order to increase the probability of correctly predicting the target activation when the same input is received again. A well-known error-driven learning algorithm is the backpropagation algorithm (Rumelhart, Hinton, & Williams, 1986). The unsupervised learning network, in which all the nodes are inter-connected and the associative weights between the nodes are adjusted by the outer product of input patterns (e.g., Hebbian Rule; Hebb, 1949), has no obvious input and output layers and no targets. In this article, only the supervised learning network is used.

Here, I provide two neural network models, which are built honestly following their respective psychological theories for category learning and have strong psychological meanings. The first is ALCOVE and the second is ATRIUM. These two models are also fit to the experimental data of Yang and Lewandowsky (2003) to demonstrate modeling.

ALCOVE

ALCOVE is a neural network model proposed by Kruschke (1992) to explain how categorization is accomplished. The basic assumption of ALCOVE is that an item would be more likely

assigned to a category containing exemplars which are more similar to that item. This assumption is widely shared by different exemplar models (e.g., the context model; Medin & Schaffer, 1978; the generalized context model; Nosofsky, 1986, 1987). In addition, ALCOVE assumes selective attention to diagnostic dimensions in order to optimize categorization performance.

Following these assumptions, ALCOVE adopts a three-layered architecture which can be seen in Figure 1. The first layer is the input layer which is responsible for receiving the outer input. The second layer is the hidden layer in which each node represents an exemplar. The third layer is the output layer in which the nodes correspond to the categories. The input node corresponds to the dimensions of stimulus; if the stimulus consists of three dimensions, then there will be three input nodes. Similarly, the number of the hidden nodes matches the number of the training stimuli used for human participants.

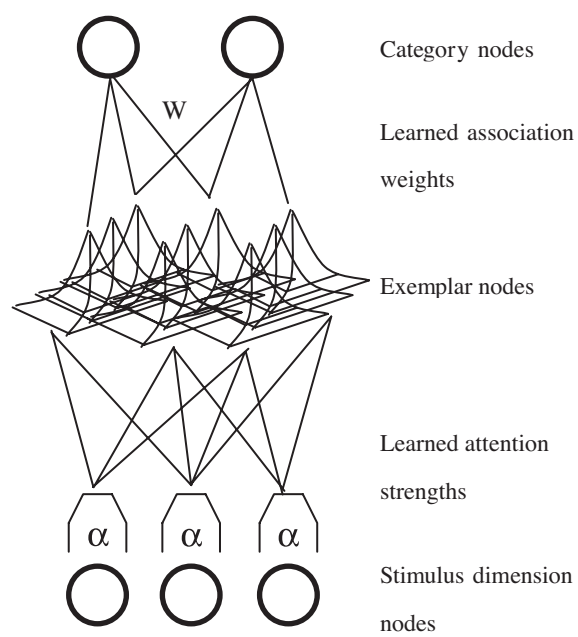

Figure 1. The architecture of ALCOVE. The input layer contains 3 nodes corresponding to 3 dimensions consisting of the stimuli used by Yang and Lewandowsky (2003). The activation of the input nodes will be weighted by the attention weightto generate the activation of every hidden node. The output activations are generated as a linear summation of the hidden activations weighted by the associative weights.

According to the similarity-based assumption, ALCOVE computes the similarity between an input a_m^{in} and an exemplar H_j^{hid} based on the psychological distance between them, d_{mj}, which is the sum of the distance on every dimension i with a learned selective-attention weight, α_i, as

$$d_{mj} = \left(\sum_i \alpha_i \left| a_m^{in} - H_j^{hid} \right|^p \right)^{1/p}, \qquad \text{E1}$$

where $p = 1$ is used for psychologically separable dimensions and $p = 2$ is used for psychologically integral dimensions (see Kruschke, 1992).

This similarity is then weighted by a sensitivity constant C and negatively transformed by the exponential function to the activation of the j_{th} hidden node, a_j^{hid}, as

$$a_j^{hid} = \exp^{-Cd_{mj}}. \qquad \text{E2}$$

As shown in E2, when the psychological distance between items is large, indicating that the two items are quite dissimilar to each other, a_j^{hid} becomes small (with a minimum of 0) as $d_{mj} \rightarrow \infty$; when the distance is small, indicating that the two items are similar to each other, a_j^{hid} becomes large (with a maximum of 1) as $d_{mj} \rightarrow 0$. The hidden activations are propagated from the hidden nodes to each output node, a_k^{out}, via their individual associative weights, w_{kj}, as

$$a_k^{out} = w_{kj} a_j^{hid}. \qquad \text{E3}$$

The final output is the predicted probability of one category, which can be computed by dividing the transformed activation of the output node by the sum of all transformed output activations as

$$p(A \mid input) = \frac{\exp^{\phi a_k^{out}}}{\sum \exp^{\phi a_k^{out}}} \qquad \text{E4}$$

where ϕ is the determinant coefficient; increasing the determinant coefficient results in increased deterministic responding.

In addition to simply generating probabilities of different categories, ALCOVE as a supervised-

learning neural network can learn from the errors to adjust the associative weights between the output and the hidden layers by E5 and to adjust the selective attention weights on the input nodes by E6. These two equations are shown as

$$\Delta w_{kj} = -\lambda_{\mathrm{w}}(a_k^{Target} - a_k^{out})a_j^{hid} \text{ and} \qquad \text{E5}$$

$$\Delta \alpha_i = \lambda_\alpha C \sum_j \sum_k \left(a_k^{Target} - a_k^{out}\right) w_{kj} a_j^{hid} d_{ij} a_i^{in}, \text{E6}$$

where λ_w and λ_α are the learning rates for associative weights and attention weights which are small and positive rational numbers and a_k^{Target} is max (1, a_k^{out}) if the target value on the k_{th} output node is 1 and min (o, a_k^{out}) if the target value on the k_{th} output node is 0.

The associative weights linking the hidden nodes to the output nodes represent the likelihoods of the hidden nodes to activate each of the categories; also, the attention weights reveal the relative importance of a stimulus dimension for the current categorization task. In other words, with error-driven learning, ALCOVE learns, based on inter-item similarities weighted by dimensional attention weights, to assign the exemplars (hidden nodes) to their correct categories during training and applies the same knowledge (i.e., the associative weights and the attention weights) to classifying the novel items in the transfer phase.

The parameters used in ALCOVE include the sensitivity constant, C, determinant coefficient, ϕ, and the learning rates for associative weight, λ_w, and for the selective attention weight, λ_α. Understanding the meanings of these parameters is quite important for handling ALCOVE. When C is small, a large psychological distance between objects is decreased, thus the exemplars are not that sensitive; whereas when C is large, even a small distance between objects is amplified, thus the exemplars are quite sensitive. Similarly, when the determinant coefficient ϕ is large, the decision making becomes more deterministic; whereas when it is small, the decision making becomes more probabilistic. The learning rates are normally constrained to the range {0, 1}.

In order to implement ALCOVE, we need the aid of computer programming. The program consists of at least four parts: initializing the parameter values, the associative weights and the attention weights; forming the output probability; learning; and outputting the results into a file. Any computer language can be used for this aim, such as C/C++, DELPHI, Visual Basic, and so on. In addition, some powerful software, such as R and MATLAB, are also recommended. The detailed procedure of implementing a model within a computer language will be introduced after the discussion of another neural network model, ATRIUM.

ATRIUM

ATRIUM (Erickson & Kruschke, 1998, 2002) is a neural network model which makes different assertions than ALCOVE about category representation; namely, ATRIUM assumes that categorization is accomplished by exemplar similarity as well as by categorization rules. The basic assumption of ATRIUM is that different kinds of the representations are processed in different modules, and the final outcome is the sum of the outputs of the modules weighted by a gating mechanism. Hence, ATRIUM is a hybrid model comprised of several smaller neural network models.

The architecture of ATRIUM can be seen in Figure 2. The exemplar module actually can be viewed as ALCOVE and has three layers with the hidden layer containing the exemplars. In addition to the exemplar module, there are two rule modules[1]. Each of the rule modules learns to predict the outcome by using a particular rule (or category boundary). The rule module is a two-layered neural network model, in which the input signal is directly linked to the output layer via the associative weights. Since the two-layered structure can

[1] The number of the rule modules depends on the category structure. Because the category structure of Yang and Lewandowsky (2003) needs two rules, I include two rule modules here.

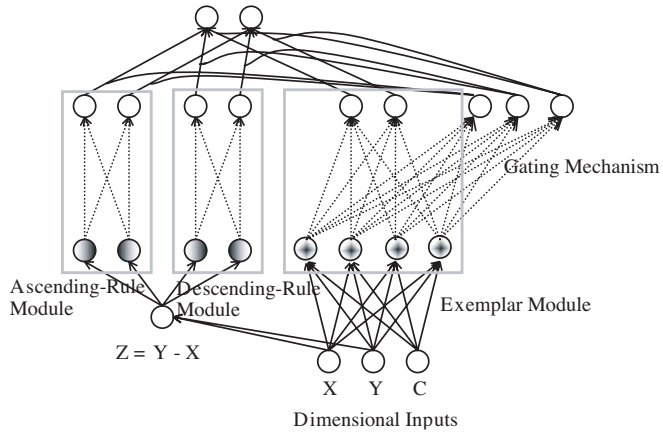

Figure 2. The architecture of ATRIUM.

only afford a neural network an ability to learn linear boundaries, all the rules in ATRIUM are linear. Besides the exemplar and the rule modules, there is a gating mechanism which is used to determine how to weight the output activations from different modules when generating the system's output activations.

Category learning in ATRIUM is different from ALCOVE. When ATRIUM receives a stimulus, the input nodes will be activated according to the physical intensities of the stimulus. Thereafter, the rule and exemplar modules independently process that input signal through the associations between layers and generate the output activations of their own. The gating mechanism then assigns different gains for the output activations of different modules based on the activations of the hidden nodes in the exemplar module. The gain for a module, which ranges between 0 and 1, is the proportion that each module's output contributes to the final output. Thus, if a module has a gain of zero, that module does not contribute to the final output. Since the activation of a hidden node in the exemplar module positively correlates with the similarity between the input and the exemplars, the job of the gating mechanism is apply the gain distribution for the most similar exemplar to weight the rule and the exemplar modules for classifying the current input. Therefore, ALCOVE assigns the same category label to similar items; whereas ATRIUM assigns the same categorization strategy to similar items.

During learning, ATRIUM also adopts the backpropagation algorithm to adjust the associative weights in each module and the attention weights on the input nodes in the exemplar module. The parameters of ATRIUM include all parameters of ALCOVE plus the learning rates in the rule module and the gating mechanism and the determinant coefficients for the gating mechanism and for the final decision-making. As in ALCOVE, these parameters should be freely estimated by fitting the model to the experimental data. To facilitate exposition, all of the functions of ATRIUM are not listed here due to their complexity and because the exemplar module is a repetition of the ALCOVE model presented above. The readers who have further interest in ATRIUM are encouraged to read the original paper (e.g., Erickson & Kruschke, 1998).

ALCOVE and ATRIUM represent different theoretical perspectives regarding human categorization. Comparing their abilities on predicting human categorization performance in an empirical task is the best way to compare these two models. To this end, I will first introduce some general procedures about modeling.

Modeling

Once we have data and models that can make predictions, using the same stimuli we can fit the model predictions to the data to see how precisely the models can predict the data. This is the basic logic of modeling. To accomplish this goal, we need to know computer programming to a certain extent; for example, we need to know how to declare variables, how to use loops, how to construct the subroutines, and how to read and create files[2]. Note that the models introduced in this arti-

[2] Sometimes, a computer program of the model can be obtained from the authors or the internet. However, it is still recommended to learn some computer programming skills to allow modification of the model according to our specific needs.

cle are too complex to estimate the parameter values analytically. Hence, we need parameter-estimation procedures to help us to estimate the values of the parameters that optimize the model's performance. Therefore, we need two parts in our computer program: model implementation and parameter estimation. The scheme for the process of modeling can be seen in Figure 3.

Model Implementation

In the model implementation stage, the user must translate the mathematical functions of a model into the computer programming language. When implementing the model, the stimuli for the experiments are fed into the model in the same random sequences as used in the experiment. On each trial, the model receives a stimulus and generates an outcome which is the probability of a particular response (e.g., Category A) and the category probabilities of all items are recorded in a file by another subroutine.

Parameter Estimation

The aim of modeling is to see whether a model can predict the human data and how different the model predictions are to the human data. This procedure is referred to as model fitting. If a model can capture the pattern of the human responses, we say that this model can fit the data well. On the contrary, if a model cannot capture the pattern, then the model cannot fit the data. When fitting a model to the human data, we change the parameter values which consequently change the model's outcomes in order to make the model predictions as close to the human data as possible. Whenever the model performance is improved, the parameter values used at that time become the best current parameter values and a new combination of parameter values are then used to test the model's performance. This loop is repeated until the model performance cannot be significantly improved or the expected round times have been achieved.

There are several algorithms which can be

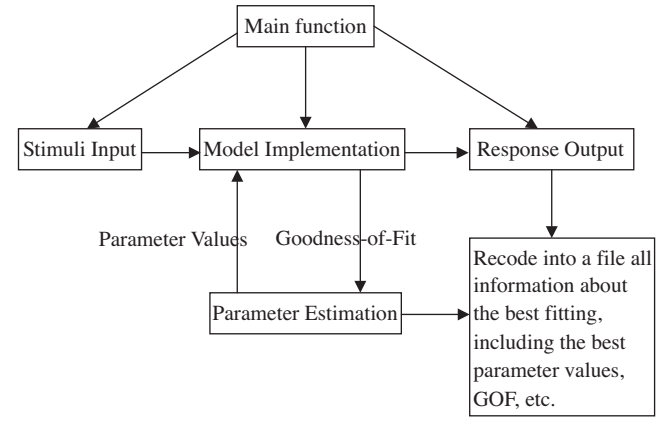

Figure 3. The scheme of the modeling procedures.

used to estimate parameters, such as hill-climbing (see Wickens, 1982) or the genetic algorithm (Coley, 1999). The hill-climbing algorithm is referred to a searching process for the optimal parameter combination in the parameter space where the model performance is most similar to humans. The hill-climbing algorithm starts the searching at one point in the parameter space and gradually moves to the optimal parameter values. However, the searching sometimes will be bound in a locally optimal point which is not the globally optimal solution but better than the other points in that local area. This is the main disadvantage of the hill-climbing algorithm. The genetic algorithm instead uses a population of hypothetical genes representing the parameter combinations to generate the model's predictions at each round and gradually selects the winner from the population, namely the parameter combination providing the optimal performance. During the gene selection, some random change (mutation) is allowable to the genes (i.e., parameter values), which consequently provides opportunities for the selection process to escape from the locally optimal combinations. The algorithm used in this study is the genetic algorithm. A more detailed comparison between these algorithms will not be introduced here as they are beyond the scope of this article.

The index of how well a model fits the observed data is called the goodness-of-fit. The

goodness-of-fit can be measured by the distance between the model predictions and the observed data, such as RMSD (Root Mean Squared Deviation)[3]; smaller distances indicate better model performance. Additionally, the goodness-of-fit can be measured by the similarity between the model predictions and the observed data, such as the percentage of the data explained by the model prediction (e.g., R^2) or the maximum likelihood (Wicken, 1982); the larger the similarity is, the better the model performs. Normally, a model with more parameters should be better able to capture the data pattern. Thus, we want to use a goodness-of-fit index which accounts for the number of parameters used by the model. In this study, I use AIC (Akaike Information Criterion) = -2LogL + 2N (Akaike, 1974), where LogL is the log of maximum likelihood and N is the number of parameters.[4] Smaller the AIC values indicate better model performance. Therefore, even though two models may have equivalent performance in terms of maximum likelihood, the model with more parameters will receive a larger penalty in the AIC measure of goodness-of-fit. The next section demonstrates to students how to do modeling with the data from a real experiment and how to interpret modeling results.

For implementing the procedures in the modeling scheme shown in Figure 3, one could use computer languages, such as C/C++. However, these computer languages normally do not have any pre-installed parameter-estimation functions and the users would need to create their own functions, which can be a very time-consuming and error-prone endeavour. These troubles can be avoided by using computer software, such as MATLAB, that have pre-installed parameter-estimation routines. Thus far, the models used in this article and the general modeling procedures have been introduced to students already. Starting from the next section, I will introduce an empirical study by Yang and Lewandowsky (2003) and introduce how to fit the two neural network models to their data.

Knowledge Partitioning in Categorization

Knowledge partitioning is the theoretical concept that knowledge might not be integrated and well-organized, but instead knowledge might be separated into different independent parcels and gated by context cues where are normatively irrelevant to the response. Since the pioneer study of Lewandowsky and Kirsner (2000) revealed that expert fire fighters make contradictory predictions to the identical fires in different contexts, an increasing number of studies support the occurrence of knowledge partitioning in a variety of domains including function learning (Kalish, Lewandowsky, & Kruschke, 2004; Lewandowsky, Kalish, & Ngang, 2002), and category learning (Lewandowsky, Roberts, & Yang, 2006; Yang & Lewanodwsky, 2003, 2004). Specifically, the occurrence of knowledge partitioning in category learning provides a good platform to evaluate the contemporary categorization models (Yang & Lewandowsky, 2004). For this reason, I adopt the data from the second experiment of Yang and Lewandowsky (2003) to demonstrate how modeling can be done.

In the second experiment of Yang and Lewandowsky (2003), the participants were asked to classify hypothetical fish to two categories A

---

3 $RMSD = \sqrt{\frac{\sum\left(P_i^{obs} - P_i^{pre}\right)^2}{n}}$, where P_i^{obs} is the observed probability of response A for stimulus i and P_i^{pre} is the predicted probability of response A for stimulus i.

4 $\mathrm{LogL} = \sum_i \log\left(\sum_k f_{ik}\right)! - \sum_i \sum_k \left(\log f_{ik}!\right) + \sum_i \sum_k \left(f_{ik} \log p_i(k)\right)$, where f_{ik} is the frequency of the i_{th} item being classified as the k_{th} category and $p_i(k)$ is the predicted probability of category k.

and B, given the environmental information of the depth where the fish was found, the density of the fish's food, and the season (winter or summer) when the fish was found. All of this information is presented as numbers on the center of the computer screen. Therefore, this is a conceptual category learning task. The category structure these authors used can be seen in Figure 4. Every point in Figure 4 represents a stimulus whose X and Y values represent for the density and the depth information respectively. The true category boundary is Y = 500 - |X - 400|. All items below the boundary belong to Category A and other items belong to Category B. The experiment has two phases. During the training phase, 40 training items which are in the region below Y = 300 are presented in different random sequences in 10 blocks. Specifically, these training items can be separated into two clusters by X = 400 and each cluster of the training items is presented with one context cue (i.e., season). Because there are half of the items in each clusters belonging to different categories, context (i.e., season) cannot predict the category label on its own. However, a perfect categorization is also possible if people rely on context to generate two rules for categorization: the ascending boundary is used for classifying items in the left season[5] and the descending boundary is used for classifying items in the right season.

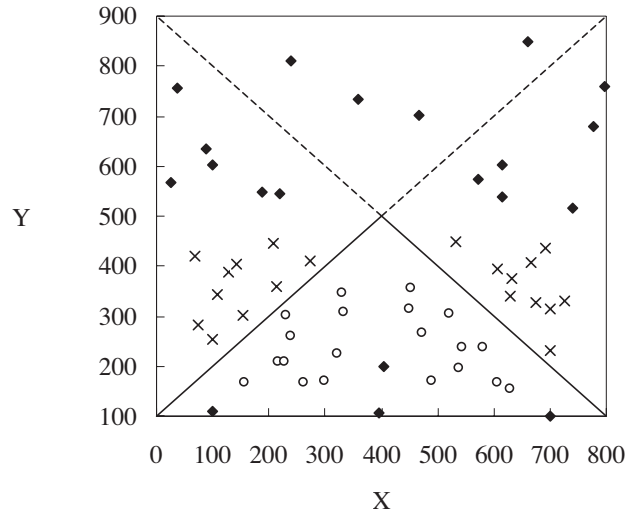

Figure 4. Category structure used in the second experiment of Yang and Lewandowsky (2003). The training items are denoted by crosses (for Category B) and circles (for Category A). The transfer items are denoted by diamonds. The true boundary is the solid triangular line separating Area 1 and Area 2 and Area 4.

During the transfer phase, every item is presented twice; once in each context. If people learn to ignore context, they would tend to respond with Category B to the transfer items about the category boundary regardless of their context. However, if people rely on context to generate two rules for categorization, knowledge partitioning occurs. That is, they would apply the ascending boundary (Y = X + 100) for categorization in the left context and the items in Area 1 and Area 4 would be classified as category A and the items in the other areas would be classified as category B; the descending boundary (Y = 900 - X) would be applied in the right context and the items in Area 1 and Area 2 would be A and the items in the other areas would be B. Therefore, the items in Area 2 and Area 4 are diagnostic for detecting the occurrence of knowledge partitioning. Yang and Lewandowsky (2003) ran two additional conditions, in which only the left or the right cluster of the training items is present in its corresponding context during training. For simplicity, I discard these two single-context conditions in this study.

The transfer results are shown in Figure 5. Apparently, participants tend to classify the items in Area 1 as Category A regardless of the items context. Importantly, the diagnostic areas (Area 2 and Area 4) show a significant difference on probability of Category A between contexts. In Area 2, participants tend to make a response of B in the left

[5] In their experiment, the season is counterbalanced across participants. Thus, here I use the left and right instead of summer and winter to refer to context.

context and a response of A in the right context to the identical item. On the contrary, this response pattern is reversed in Area 4. The unexpected significant difference on the probability of Category A between contexts might result from poor learning of the right context boundary. In the left context, the participants clearly tend to respond Category A to the items in Area 1 and Area 4 but respond Category B to the items in Area 2 and Area 3. This indicates that the ascending boundary is learned well by the participants. If the descending boundary is learned well also, then the participants should tend to make the A response to the items in Area 1 and Area 2 but make the B response to the items in Area 3 and Area 4. However, this tendency is relatively blurred, that indicates that the descending boundary is not well learned[6].

The results shown in Figure 5 might be the blend of different individual performances. Thus, Yang and Lewandowsky (2003) also conduct the *K*-means cluster analysis to analyze the participants' transfer responses. The result indicates that there are two groups of participants who perform very differently in the transfer phase; one exhibits performance consistent with the use of the true boundary (TB) and the other exhibits performance consistent with knowledge partitioning (KP). The transfer performance of both groups is shown in Figure 6. It is clear from the transfer profile of the KP participants that the items in Area 1 are always classified as Category A and the context-dependent difference on probability of Category A on the items in Area 2 and Area 4 become more salient. However, the TB participants show no context-dependent difference on the items in all areas. These results support the occurrence of knowledge partitioning in categorization and also highlight the group differences on learning categories.

The theoretical implication of these results is that people can attend to an irrelevant context cue and generate different categorization strategies (or rules) in different contexts. This finding might challenge ALCOVE's basic assumption about selective attention in categorization - namely, that

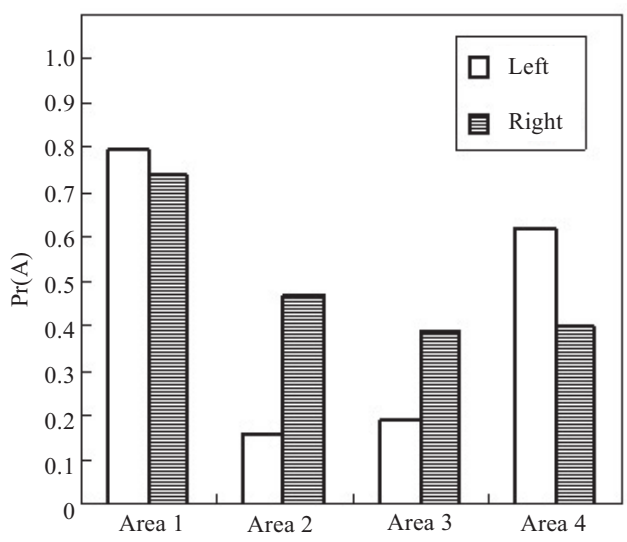

Figure 5. The transfer results from the second experiment of Yang and Lewandosky (2003). The averaged probability of Category A for the items in each area is shown in two contexts. The data are replotted from the report of Yang and Lewandowsky (2003).

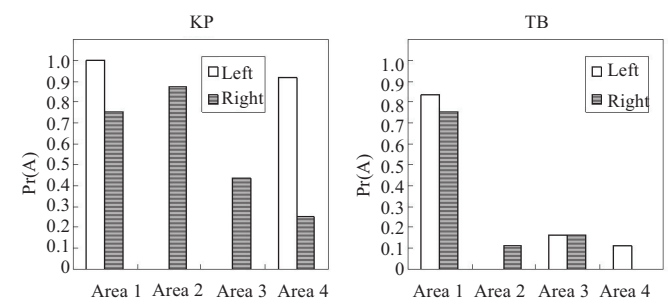

Figure 6. The group differences in the experiment of Yang and Lewandowsky (2003). The left and the right panels represent the KP and the TB groups, respectively.

6 Yang and Lewandowsky (2003) suggest that this unbalanced learning of the two boundaries is affected by the characteristic of function learning in that the ascending function is easier to learn than the descending function.

only the predictive dimensions are attended. Therefore, it can be expected that ALCOVE should always learn to ignore context and learn the true boundary. However, ATRIUM does not have such a constraint on selective attention. Furthermore, the modularized structure of ATRIUM suits the framework of knowledge partitioning. Thus, ATRIUM can be expected to better accommodate the experimental data than ALCOVE. By contrast, if ALCOVE can perform equally well or even outperform ATRIUM, the present results cannot be used to draw the conclusion that knowledge partitioning occurs in categorization, given that we a pure exemplar-based account (i.e., ALCOVE) can explain the results. No conclusion can be drawn until we have compared the modeling performance of these two models.

Results and Discussion

Both ALCOVE and ATRIUM are fit to the transfer data from the second experiment of Yang and Lewandowsky (2003). The hypothesis is that ALCOVE will always learn to ignore context; thus, ALCOVE should be able to account for the TB group's responses but not the KP group's responses; however, ATRIUM should be able to learn to attend to context for categorization and, hence, be better able to account for both types of transfer responses. Therefore, these two models are fit to the KP and the TB groups' data respectively. As the stimulus dimensions are perceptually separable and the participants can easily attend to one dimension regardless of the others, the city-block measure is used for computing the psychological distance, which is then transformed to a measure of similarity (see Nosofsky, 1986; Shepard, 1987). Additionally, in accordance with the concept of knowledge partitioning, I set up two rule modules in ATRIUM; one for the ascending boundary and the other for the descending boundary. The predictions of both models are shown in Figure 7.

Both models can capture the TB response pattern very well (RMSD = 0.05 and AIC = 58.96 for ALCOVE; RMSD = 0.05 and AIC = 67.50 for ATRIUM) with ALCOVE performing better than ATRIUM because of fewer freely-estimated parameters. However, as expected, ALCOVE performs worse (RMSD = 0.44 and AIC = 295.12) than ATRIUM (RMSD = 0.08 and AIC = 70.76) when fit to the KP pattern. The estimated parameter values are shown in Table 1. For ALCOVE, four parameters are freely estimated: the specificity, c, the decision constant, ϕ, the learning rate for association weights between exemplars and output nodes, λ_w, and the learning rate for dimensional attention weights represented by λ_α. Similarly, for ATRIUM, nine parameters are freely estimated: the specificity, c, the decision constant, ϕ, the decision constant for the gating mechanism, ϕ_g, the bias of the rule boundary, γ_r, the learning rate of the exemplar module, λ_e, the learning rates of the rule 1 module and the rule 2 module, λ_{r1} and λ_{r2}, the learning rate of the gating mechanism, λ_g, and the learning rate of the attention weights, λ_α.

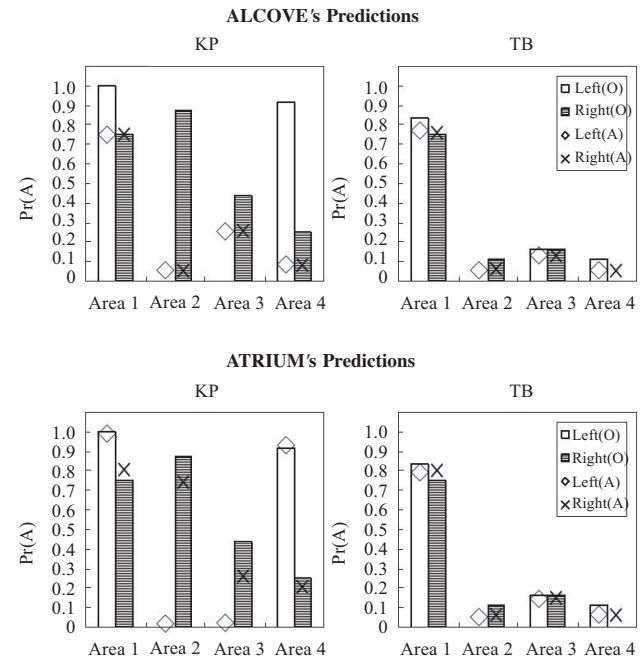

Figure 7. The transfer predictions of ALCOVE and ATRIUM fit to the KP and the TB groups in the second experiment of Yang and Lewandowsky (2003).

Table 1
The Best Fitting Parameter Values for ALCOVE and ATRIUM.

Parameter	TB	KP
	ALCOVE	
C	2.03	3.68
ϕ	3.06	1.59
λ_w	0.01	0.03
λ_α	0.01	0.02
	ATRIUM	
C	7.01	9.44
ϕ	9.79	7.71
ϕ_g	0.30	0.45
γ_r	0.27	4.72
λ_e	0.06	0.55
λ_{r1}	1.25	1.86
λ_{r2}	1.14	0.01
λ_g	1.34	0.80
λ_α	0.58	0.14

Visual inspection of the model predictions reveals that ALCOVE always predicts the same probability of Category A regardless of context. The initial attention weight on context of ALCOVE is always set to .33 in the simulations for both groups; however, at the end of training, the attention weight is estimated to be 0 when fit to the TB pattern and .03 when fit to the KP pattern. This is the main reason why ALCOVE cannot account for the KP response pattern. However, the learned attention weights of ATRIUM, despite starting at .33 as in ALCOVE, are .03 and .66 at the end of training when fit to the TB and the KP patterns, respectively. Apparently, the ability of ATRIUM to account for the KP pattern comes from the ability to maintain a reasonable amount of attention on context.

Although it is evident that ALCOVE has difficulty accounting for the KP pattern under the normal learning situation, it is still interesting to know if there is any possibility for ALCOVE to fit the KP pattern. For this simulation, the initial attention weight to context is set to .98, and the attention weights on the other two dimensions are set to .01. The modeling results are shown in Figure 8. Visual inspection indicates that the difference on the probability of Category A on the items in Area 2 and Area 4 becomes larger between two contexts. The model performance is now improved with RMSD = 0.28 and AIC = 121.08. However, it still performs worse than ATRIUM, possibly because ALCOVE cannot accurately capture the response pattern in the left context; it overestimates the probability of Category A for the items in Area 3 and underestimates that for the items in Area 4.

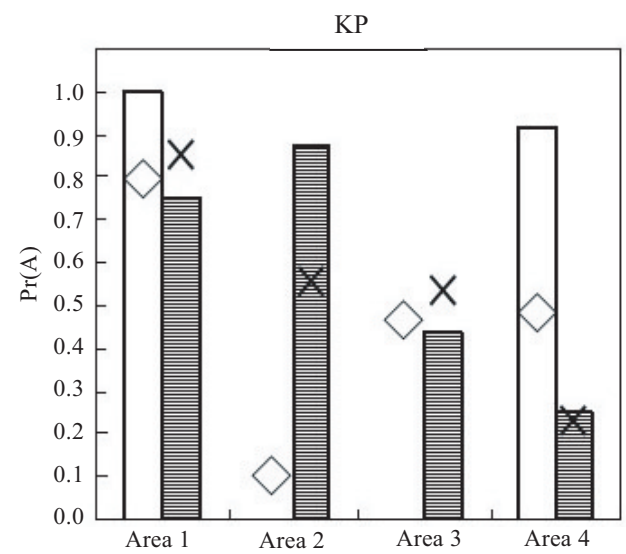

Figure 8. The predictions of ALCOVE when fit to the transfer data of the KP group with the initial attention weight to context set to .98.

To sum up, both ALCOVE and ATRIUM have no difficulty accounting for the TB response pattern. However, only ATRIUM can accommodate the KP pattern well. This is because ATRIUM does not learn to ignore context despite the fact that context is not predictive of the category labels at all. How can ATRIUM do that? This is mainly because of the architecture of ATRIUM. In ATRIUM, the exemplar module is just like ALCOVE which presumably should learn to ignore context. However, the attention weight on context is learned to be .66 which is higher than the initial status of .33. This means that the exemplar module

increases the attention weight on context. Although this adjustment of the attention weight on context decreases the power of the exemplar module to predict the outcomes, it helps the gating mechanism to make a fast shift between the two rule modules. Consequently, the use of the two rule modules becomes more context-dependent; exactly what is expected by knowledge partitioning.

One theoretical concern revealed from these modeling results is whether the learned categories are heterogeneous. Most of the contemporary categorization models (e.g., the GRT; Ashby & Gott, 1988; COVIS; Ashby, Alfonso-Reese, Turken, & Waldron, 1998; ALCOVE; Kruschke, 1992; the GCM; Nosofsky, 1986, 1987, RULEX; Nosofsky, Palmeri, & McKinley, 1994), although superficially different, all have a common assumption that the category representation is homogeneous. Take the exemplar-based model as an example. When learning the categories, all items are classified following the same manipulation of the selection attention. That is, an exemplar-based model is not able to selectively attend to one dimension for one set of items and to another for another set of items (see Aha & Goldstone, 1990). Likewise, rule-based models, such as the GRT, assume that the category boundary define the relationship between categories regardless of the items that are presented. However, knowledge partitioning in categorization indicates that people learn different relationships between categories in different contexts[7], namely, that category representation can be heterogeneous. Since ATRIUM preserves the uniqueness of every item by the gating mechanism, it is able to account for heterogeneous representation. It may be possible for other category learning models, such as SUSTAIN (Love, Medin & Gureckis, 2004), to predict knowledge partitioning by classifying items through the creation of 4 clusters representing the 4 types of the training items: Category A + left context, Category A + right context, Category B + left context, and Category + right context.

Conclusion

In this article, the main purpose is to introduce the local (Taiwanese) students to cognitive modeling, and explain how it can be done and why it is important. I use two neural network models to fit to the data from the study of Yang and Lewandowsky (2003). The modeling results clearly help to distinguish one model from another. In addition, these results might give us some inspiration for future study. For instance, since ALCOVE can approximate to the KP pattern with a huge attention weight on context at the onset of experiment, we may want to empirically test this possibility by instructing people to attend to context and see whether this increases the occurrence of knowledge partitioning.

The structure of program used for modeling in this article is a typical structure. Thus, people who want to do their own modeling are welcome to apply it to composing their own computer programs. Of course, the structure can be adjusted according to the users' particular needs. This article provides an example of how to fit models to the human data. Computer modeling is a good way to provide quantitative measurements to the models' performances that is more precise than what the verbal models can provide.

However, fitting a model to the data is not the only way to evaluate a model. For example, we can set up different parameter values for different experimental conditions and see how the model will perform with those parameter values. Based on this observation, we can predict how real human subjects will perform in the same conditions. This is extremely useful when we would like to know in advance what people might react to a new experimental treatment and when we cannot run some experiments in reality due to the economical, sub-

[7] Here, the relationship between categories is the categorization boundary.

ject-recruiting, or theoretical reasons.

Finally, neural network models are not the only models suitable for computer modeling. In contrast, any model, as long as it provides mathematical functions, can be implemented in a computer program and fit to the data. The reasons for choosing neural network models for demonstration include (1) neural network models are widely accepted as a decent account of psychological processes, (2) neural network models are particularly suitable for accounting for human learning phenomena, and (3) neural network models contains almost all elements of a computational model, namely the mental processes, the mental representations, and their structural relationships.

For a prolonged impact on teaching and researching cognitive modeling, it is suggested that relevant undergraduate-level or graduate-level courses are provided in university, such as cognitive modeling, computer programming, mathematical methods, and probability. Additionally, students should be encouraged to utilize and practice computer modeling.

References

Aha, D. W., & Goldstone, R. L. (1990). Concept learning and flexible weighting. In J. K. Kruschke (Ed.), *Proceedings of the Fourteenth Annual Conference of the Cognitive Science Society* (pp. 534-539). Hillsdale, NJ: Erlbaum.

Akaike, H. (1974). A new look at the statistical model identification. *IEEE Transactions in Automatic Control*, *19*, 716-723.

Anderson, J. A. (1995). *An introduction to neural networks*. Cambridge MA: MIT Press.

Ashby, F. G., Alfonso-Reese, L. A., Turken, A. U., & Waldron, E. M. (1998). A neuropsychological theory of multiple systems in category learning. *Psychological Review*, *105*, 442-481.

Ashby, F. G., & Gott, R. (1988). Decision rules in the perception and categorization of multidimensional stimuli. *Journal of Experimental Psychology: Learning, Memory, and Cognition*, *14*, 33-53.

Coley, D. A. (1999). *An introduction to genetic algorithm for scientists and engineers*. Singapore: World Scientific Publishing.

Craik, F. I. M., & Lockhart, R. S. (1972). Levels of processing: A framework for memory research. *Journal of Verbal Learning and Verbal Behavior*, *11*, 671-684.

Erickson, M. A., & Kruschke, J. K. (1998). Rules and exemplars in category learning. *Journal of Experimental Psychology: General*, *127*, 107-140.

Erickson, M. A., & Kruschke, J. K. (2002). Rule-based extrapolation in perceptual categorization. *Psychonomic Bulletin and Review*, *9*, 160-168.

Hebb, D. (1949). *The organization of behavior*. New York: Wiley.

Kalish, M. L., Lewandowsky, S., & Kruschke, J. K. (2004). Population of linear experts: Knowledge partitioning and function learning. *Psychological Review*, *111*, 1072-1099.

Kruschke, J. K. (1992). ALCOVE: An exemplar-based connectionist model of category learning. *Psychological Review*, *99*, 22-44.

Lewandowsky, S. (1993). The rewards and hazards of computer simulations. *Psychological Science*, *4*, 236-243.

Lewandowsky, S., Kalish, M., & Ngang, S. K. (2002). Simplified learning in complex situations: Knowledge partitioning in function learning. *Journal of Experimental Psychology: General*, *131*, 163-193.

Lewandowsky, S., & Kirsner, K. (2000). Knowledge partitioning: Context-dependent use of expertise. *Memory & Cognition*, *28*, 295-305.

Lewandowsky, S., Roberts, L., & Yang, L.-X. (2006). Knowledge partitioning in categorization: Boundary conditions. *Memory & Cognition*, *34*, 1676-1688.

Love, B. C., Medin, D. L., & Gureckis, T. M. (2004). SUSTAIN: A network model of category learning. *Psychological Review*, *111*, 309-332.

Medin, D. L., & Schaffer, M. M. (1978). Context theory of classification learning. *Psychology Review*, *85*, 207-238.

Nosofsky, R. M. (1986). Attention, similarity, and the identification-categorization relationship.

Journal of Experimental Psychology: General, *115*, 39-57.

Nosofsky, R. M. (1987). Attention and learning process in the identification and categorization of integral stimuli. *Journal of Experimental Psychology: Learning, Memory, and Cognition*, *13*, 87-108.

Nosofsky, R. M., Palmeri, T. J., & McKinley, S. C. (1994). Rule-plus-exception model of classification learning. *Psychological Review*, *101*, 53-79.

Rumelhart, D. E., Hinton, G. E., & Williams, R. J. (1986). Learning internal representations by error propagation. In D. E. Rumelhart, J. L. McClelland, & the PDP Research Group (Eds.), *Parallel distiributed processing* (Vol. 1., pp. 316-362). Cambridge, MA: MIT Press.

Shepard, R. N. (1987). Toward a universal law of generalization for psychological science. *Science*, *237*, 1317-1323.

Waugh, N. C., & Norman, D. A. (1965). Primary memory. *Psychological Review*, *72*, 89-104.

Wickens, T. D. (1982). *Models for behavior: Stochastic processes in psychology*. San Francisco: Freeman and Company.

Yang, L.-X., & Lewandowsky, S. (2003). Context-gated knowledge partitioning in categorization. *Journal of Experimental Psychology: Learning, Memory, and Cognition*, *29*, 663-679.

Yang, L.-X., & Lewandowsky, S. (2004). Knowledge partitioning in categorization: Constraints on exemplar models. *Journal of Experimental Psychology: Learning, Memory, and Cognition*, *30*, 1045-1064.

認知模擬在類別學習上的應用：以Yang與Lewandowsky（2003）之研究爲例

楊立行

成功大學認知科學所

爲了探索人類心理歷程與心智表徵，各種不同的認知模型不斷地被研究者提出。這些認知模型代表著不同的理論觀點，它們不僅可以對現象提出解釋，還可以對未知進行預測。當我們想要檢視一個理論模型對於現象的可以達到多好的解釋力，以實徵資料進行電腦模擬就成爲了一項強而有力的研究工具。然而本地（台灣）的心理學背景的學生往往缺少學習這項工具的管道，而不清楚什麼是電腦模擬、不知道如何進行，不了解它的重要性何在。這部分可能源自於心理系所鮮少開設相關的課程，也或者是它需要較高的程式設計能力。因此，本文目的在於提供進行認知模擬的概念性引導方針：文中將首先介紹兩個在類別學習領域上相當知名的類神經網路模型ALCOVE與ATRIUM，並以Yang與Lewandosky（2003）知識分化的研究爲例進行電腦模擬，透過此二模型與實徵資料的分別比對結果，進一步對此二模型背後所支持的理論進行比較分析。模擬結果顯示，ATRIUM對於在類別學習上的知識分化現象的解釋力明顯高於ALCOVE。此外，一些相關的理論層次的議題，如分類表徵的異質性等，也因爲獲得模擬的結果而能夠被更深入的討論。

關鍵詞：認知模擬、類別學習、類神經網路