Decision Making Connectionist Networks

Yves Chauvin
Psychology Department
Stanford University
Stanford, CA 94305
chauvin@psych.stanford.edu

Abstract

A connectionist architecture is proposed and provides representations for probabilities and utilities, the basic elements of formal decision making theories. The outputs of standard feedforward feature-extraction networks then become inputs to this decision making network. A formalism shows how the gradient of expected utility can be backpropagated through the decision making network "down" to the feature extraction network. The formalism can be adapted to algorithms which optimize total or minimum expected utility. Utilities can be either given or estimated during learning. When utility estimation and decision making behavior adapt simultaneously, learning dynamics show properties contrasted to "puzzling" observations made in experimental situations with human subjects. The results illustrate the interest in computational properties emerging out of the integration of elements of decision making formalisms and connectionist learning models.

Introduction

By tradition, formal theories of decision making build concepts of utility and probability out of a set of normative axioms characterizing "rational" behavior (e.g., Slovic, Lichtenstein & Fischhoff, 1988). They usually do not emphasize adaptive behavior, notions of representation, or information processing properties. Inspired by connectionist "principles", this paper suggests a general connectionist architecture for decision making which can be integrated with pre-existing feature extraction feedforward networks. The architecture provides representations for probabilities and utilities. Learning integrates elements of decision making formalisms in a computational framework and then operates through the resulting representations.

Using this architecture, a variant of the backpropagation learning algorithm can be derived which propagates the gradient of a response expected utility through the decision network "down" to the feature extraction network. This principle generates two algorithms, one maximizing total expected utility, the other maximizing the minimum expected utility computed over the set of possible responses. The goal of this paper is then to examine learning dynamics and convergence properties of these algorithms.

These properties are described through simulations of two-response environments. In particular, when utilities are estimated during behavior adaptation, simulations show interesting results which are then compared to experimental data observed with human subjects. The results illustrate the relevance of computational models of decision making.

Probability and Utility Representations

Probability Representation

Consider a multinomial distribution where a given input pattern has to be classified in 1 of n categories. We would like the n-unit output layer of a network to represent the set of classification probabilities associated with this multinomial distribution. If we call p_i the activations of these units, we need to have $p_i \geq 0$ and $\sum_i^n p_i = 1$. One way to satisfy these constraints is to make sure that unit activations of a previous layer are positive and to normalize these activations at the output layer. If we write s_i the "net input" to the first layer, positivity can be obtained with exponential *e-units*: $e_i = e^{\beta s_i}$. Normalization is then obtained with *p-units*:

$$p_i = \frac{e_i}{\sum_j^n e_j} = \frac{e^{\beta s_i}}{\sum_j e^{\beta s_j}} \qquad (1)$$

This corresponds to the familiar ratio rule used in numerous decision tasks (e.g., Busemeyer and Myung, 1989; Luce, 1959; Estes, 1987). The parameter β can be seen as a sensitivity parameter. For $\beta = 0, p_i = 1/n$: the response distribution is uniform across categories. When $\beta \rightarrow \infty, p_M \rightarrow 1$ where M corresponds to $s_M = max_i\{s_i\}$: the network response is deterministic and corresponds to the largest net input. A simple computation yields the derivative of the *p-units* with respect to the activations of the *e-units*:

$$\frac{\partial p_i}{\partial e_j} = -\frac{p_i \cdot p_j}{e_j} \qquad j \neq i \qquad (2)$$

$$\frac{\partial p_i}{\partial e_i} = \frac{(1-p_i).p_i}{e_i}$$

We can then write the derivative of the *p-units* with respect to the net inputs of the *e-units*:

$$\frac{\partial p_i}{\partial s_j} = \beta(\delta_{ij} - p_i).p_j \qquad 1 \leq i,j \leq n \qquad (3)$$

where δ_{ij} is the Kronecker symbol ($\delta_{ij} = 1$ for $i = j$, 0 otherwise).

Subjective Expected Utility

We now consider the activations p_i of the *p-units* correspond to network "propensities" to classify an input pattern in category i. We assume these response propensities (or "beliefs" or "subjective probabilities") are computed so as to maximize a given performance index. We consider payoffs are obtained when the network classifies an input pattern in category j and when the environment responds with category i. We write u_{ij} (utilities) the perceived values of the terms of the $n.n$ payoff matrix (e.g., utility might typically be a bounded, monotonically increasing function of payoff; utility functions are not examined in this paper). We assume for now the u_{ij} are known. We will see below how they can also be estimated during learning.

The network expected utility for category i can be written as $E[U_i] = \sum_j^n u_{ij}.p_j$. This equation can be implemented in a network by adding a layer of n linear *u-units* with a connectivity matrix between *p-units* and *u-units* corresponding to the utility matrix. Outputs of a feature extraction network can then serve as inputs to the decision making network computing probabilities and expected utilities. The resulting network architecture is presented in Figure 1.

Learning consists in maximizing the expected utility ($EU_t = E[U_t]$) associated with a targetted category t. The network learns by adjusting its set of "beliefs" p_j through parameter (weight) adaptation. We can use gradient ascent on EU_t to compute weight changes after each pattern presentation:

$$\Delta w_q = \eta \frac{\partial U_t}{\partial w_q} \qquad (4)$$

Using the chain rule, we get:

$$\frac{\partial E[U_t]}{\partial w_q} = \sum_j^n \frac{\partial E[U_t]}{\partial s_j} \cdot \frac{\partial s_j}{\partial w_q} \qquad (5)$$

The terms $\partial s_j / \partial w_q$ might be computed with the standard back-propagation algorithm (Rumelhart, Hinton & Williams, 1986), considering a given feature-extraction architecture "below" the *e-units*. We then use the chain rule to obtain the back-propagated "error" term δe_j associated with the *jth e-unit*:

$$\delta e_j = \frac{\partial E[U_t]}{\partial s_j} = \sum_k^n \frac{\partial E[U_t]}{\partial p_k} \cdot \frac{\partial p_k}{\partial s_j}$$

$$= \sum_k^n u_{tk}\beta(\delta_{kj} - p_k).p_j = \beta p_j.(u_{tj} - U_t) \qquad (6)$$

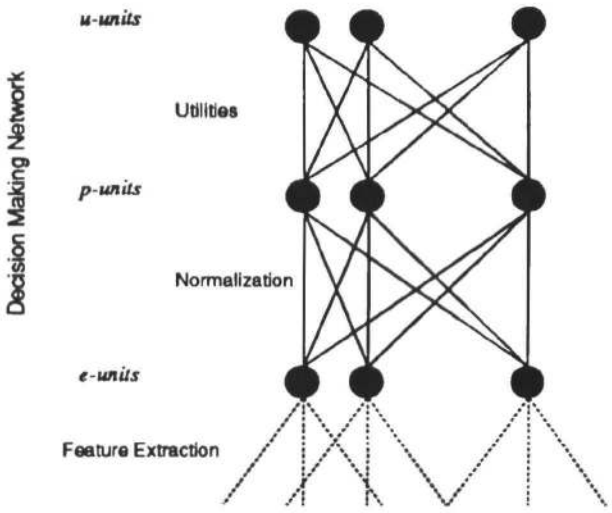

Figure 1: Decision making network architecture: The architecture provides representations for probabilities (*p-units*) and utilities (weights from *p-units* to *u-units*). The decision making network might be connected to a feature extraction network, for example a standard multi-layer back-propagation network.

Note since $\sum_j p_j = 1$, $\sum_j \delta e_j = 0$. Moreover, if $u_{tj} > E[U_t]$, the network "receives more than expected": $p_j.(u_{tj} - U_t) > 0$ and p_j will increase, as it should.

Given a set of u_{ij}, the previous equation computes the gradient of expected utility EU_t with respect to the input of the decision making network. This gradient can then be back-propapaged "further down" in hidden layers to extract features relevant to the decision strategy. The parameters (weights) of the feature-extraction network can then be changed by gradient ascent. The goal of this paper is not to investigate how the decision making network influences the properties of the feature extraction network. Rather, we look at properties of the decision network itself, at various decision strategy implementations and at utility estimations.

Maximizing Total Expected Utility: Fixed Utilities

Considering a set of training patterns, the total expected utility is $EU = \sum_t EU_t$: maximizing the total expected utility consists in maximizing the sum of response expected utilities. If the training sample is known, batch learning can be implemented to compute $\partial EU / \partial w_q$. Alternatively, the well-known Widrow-Hoff procedure can be used to adjust the weights after each pattern presentation.

Sample Learning Dynamics

We consider the u_{ij} are known *a priori*. Suppose the network is now being trained with a finite data base in

which a given input pattern is presented N times and targetted N_i times in each category i ($\sum_i N_i = N, f_i = N_i/N, \sum_i f_i = 1, f_i \neq f_j$ for $i \neq j, f_M = max_i\{f_i\}$). We can then compute the total sample gradient of EU (associated with that given input pattern) with respect to the net input s_j:

$$\delta e_j = \frac{\partial EU}{\partial s_j} = \sum_i N_i \frac{\partial E[U_i]}{\partial s_j} \quad (7)$$

Assuming a utility matrix equal to the identity matrix ($u_{ij} = \delta_{ij}$), we get:

$$\begin{aligned} \frac{\partial U_i}{\partial s_j} &= \beta(\delta_{ij} - p_i)p_j \\ \delta e_j &= N\beta p_j(f_j - \sum_i f_i p_i) \\ &= N\beta p_j \sum_{i \neq j} p_i(f_j - f_i) \quad (8) \\ \delta e_M &= N\beta p_M \sum_{i \neq M} p_i(f_M - f_i) \end{aligned}$$

Equilibria are obtained for $\delta e_j = 0$ for all j. But $\delta e_M = 0$ implies $p_M = 0$ or $p_i = 0$ for $i \neq M \Leftrightarrow p_M = 1$. The first equilibrium is unstable; the second one is stable. Therefore the response probability associated with the biggest response frequency of the sample will increase to 1 during training. Using the same arguments, the response probability associated with the smallest response frequency will decrease to 0. All other response probabilities may either decrease or first increase then decrease depending on the initial values of p_j. These results may be generalized for any value of u_{ij}.

The learning principle involved in this framework is gradient ascent on the total expected utility when utilities are known in advance. It is well-known that resulting so-called Bayesian optimality can be obtained when response probabilities correspond to "pure" decisions ($p_j = 0$ or 1), even in stochastic environments (e.g., DeGroot, 1970).

Illustration: Two-response Environment

The algorithm was implemented for a simple two-response environment without feature extraction. In this case, the two weights to be learned in the network are simply the biases of the *e-units*. The environmental binomial distribution can be characterized by a single parameter $p^* = p_1^*$, the probability of occurrence of the first response/category. Rewards from the environment are received as a function of the network responses. The previous EU-back-propagation algorithm should find the optimal values of the probabilities as a function of the utility matrix.

As expected, simulations show the network always learns how to make pure optimal decisions, independently of the utility matrix and environmental probabilities (except in obvious trivial cases). Convergence speed is essentially dependent on the environmental probabilities, on the range of the u_{ij}, on the learning rate η and on the sensitivity parameter β.

Standard Bayesian techniques might first estimate posterior probabilities and then compute expected costs from given *a priori* cost matrices. Input patterns are then classified in categories associated with low expected costs. The network presented above directly estimates the model parameters so as to directly maximize total expected utility. The EU-back-propagation might "bypass" the estimation of posterior probabilities to compute response behavior. From a performance point of view, it is quite possible to imagine that the model might actually make better use of its resources (weights) by not "wasting" them in estimating posterior probabilities. Performance characteristics are not investigating further in this paper.

Maximizing Total Expected Utility: Estimated Utilities

We now assume the u_{ij} have to be estimated during learning. This task can be considered more typical of human learning situations in decision making environments. In this framework, the EU-back-propagation algorithm can be used with the EU gradient being back-propagated through currently estimated utilities. The task of the decision network is still to maximize EU through adaptation of the response probabilities p_j given current u_{ij}.

Trial Estimation

Let R (with elements r_j) be the current n-dimensional response vector generated by the network. The response R is a random variable with a multinomial distribution characterised by the p_j. At each trial, we assume the network response R is chosen in function of this distribution and that the environment responds with a "target" vector R^* (with elements r_i^*) drawn from the environmental multinomial distribution characterised by a set of probabilities p_i^*. We write u_{ij} the current estimates of the utilities and u_{ij}^*, the perceived values of the true rewards obtained after each network response R and environment response R^*.

Trial learning can be implemented in a network having the set r_j as input units and the set r_i^* as output units. The received utility predicted by the network after each trial is then $U_i = \sum_j u_{ij} r_j$; the truly received utility is $U_i^* = \sum_j u_{ij}^* r_j$. Estimation of the received utilities is performed by gradient descent on L, the sum of squares of the differences between predicted and truly received utilities. Considering a single trial, we obtain:

$$\begin{aligned} L &= \sum_i r_i^*(U_i - U_i^*)^2 \\ \Delta u_{ij} &= -\alpha_1 \frac{\partial L}{\partial u_{ij}} = -\alpha(U_i - U_i^*)r_i^* . r_j \quad (9) \\ E[\Delta u_{ij}] &= -\alpha(u_{ij} - u_{ij}^*)p_i^* p_j \end{aligned}$$

Equilibrium is obtained for $E[\Delta u_{ij}] = 0 \Leftrightarrow u_{ij} = u^*_{ij}$. Speed of convergence will not only depend on the learning rate α but also on the current set of beliefs p_j and on the the environmental probabilities p^*_i. After each trial, utilities are evaluated, and the *EU*-back-propagation algorithm is used with the currently estimated set of utilities. Eventually, the estimated utilities will converge to the environmental utilities (for non-zero p_j and p^*_i). At this point, response behavior adaptation will become identical to the case where utilities are known a *priori*. Therefore, the network will end up making optimal pure decisions. However, in general, response probabilities being learned simultaneously with utility estimation, the complete learning dynamics might become quite complex. These characteristics are examined in more detail below.

Average Estimation

With trial learning, u_{ij} is reevaluated after each trial, in function of the network response R and the environment response R^*. Because a specific element of the utility matrix has to be estimated after each trial, trial learning makes the implicit assumption that the learner already understands the deterministic property of trial obtained rewards. Trial learning also makes the computational assumption the learner stores and estimates each elements of the utility matrix independently. We now turn to a maybe more natural learning hypothesis where expected utilities are estimated in the average. Utilities are evaluated in function of current response probabilities because the learner does not assume (or trust) the determinism of the utility matrix or does not have the information processing capacity to estimate each element of the utility matrix.

After each trial, the expected utility (in the mathematical sense of expected) is $E[U_i] = \sum_j u_{ij} p_j$. For given p_j, it is possible to estimate the new u_{ij} by gradient descent on the squares of the differences between expected utility $E[U_i]$ and the truly received utility U^*_i. Considering a single trial, we obtain:

$$\begin{aligned} L &= \sum_i r^*_i (E[U_i] - U^*_i)^2 \\ \Delta u_{ij} &= -\alpha_1 \frac{\partial L}{\partial u_{ij}} \\ &= -\alpha (E[U_i] - U^*_i) r^*_i . p_j \\ E[\Delta u_{ij}] &= -\alpha (E[U_i] - E[U^*_i]) p^*_i p_j \qquad (10) \end{aligned}$$

Equilibrium is reached when $E[U_i] = E[U^*_i] \Leftrightarrow \sum_j u_{ij} p_j = \sum_j u^*_{ij} p_j$. An obvious solution to this equation is found for $u_{ij} = u^*_{ij}$. Again, in this case, the network will then converge to pure decision optimality. However, the equilibrium condition might lead to other estimations of u_{ij} which might correspond to various response behaviors. For each run and for a given limited learning time, obtained behaviors will depend on the network initial conditions and on the random order of category presentations during training.

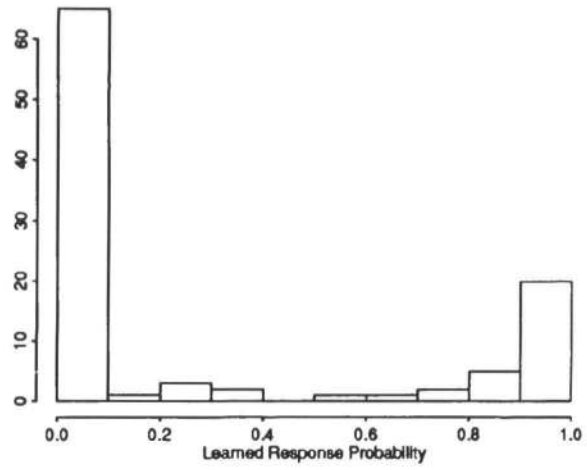

Figure 2: Histogram of learned response probability p after 200 trials for 100 runs. The decision network is trained with the *EU*-back-propagation algorithm using average estimated utilities.

Simulations of the algorithm were then performed in the previous two-response environment with $u^*_{ij} = \delta_{ij}$. The initial response probability parameter p was randomly chosen at each run. The following general results were obtained, that we will contrast to observations made with humans in two-response experiments.

A great majority of runs are stable after a fixed number of trials: for a sufficient number of runs, average performance seems to have reached an asymptotic level. Settlement occurs in two modes: pure decision optimality where p converges to 1 if $p^* > .5$ and to 0 otherwise; a less common "reversed" optimal mode where p converges to 0 if $p^* > .5$ and to 0 otherwise. The mean of the resulting bimodal distribution depends of the values of the parameters, in particular of u^*_{ij}, of the initial values for u_{ij} and of the parameters α and β. Larger learning rates for the estimation of utilities or more accurate initial estimations lead to improved estimations of u_{ij} during learning and to higher proportion of cases converging to optimal behavior. Figure 2 represents the histogram of learned response probability p after 200 trials for 100 runs (varying the initial values of p; $\alpha = 1$, $\beta = 4$, initial $u_{ij} = .25\delta_{ij}$, $p^* = .2$). The mean value of the distribution is .27 in this case, roughly matching the environmental p^*.

A model predicting "reversed" optimality seems a *priori* to deserve rejection as a potential model of human behavior. However, Luce and Suppes (1963), reviewing a large number of two-response studies, have the following warning:

> The data reported in the literature are always averages for groups of subjects, [...]. Considerable evidence (not much of it published) indicates that these group results can be quite misleading. The

distribution of estimated p_∞ over subjects often seems not to be binomial, and sometimes there is little doubt but that it is bimodal, with the valley of the distribution coinciding roughly with the group mean probability. (Luce & Suppes, 1963, p. 391).

Until now, these experimental individual data still remain a puzzle for theoretical psychologies of learning and decision making. The present approach suggests an explanation for these data in the simultaneous average estimation of rewards and adaptation of response behavior. The mean result (probability matching) is then observed along with the puzzling bimodal distribution of individual responses. Precise simulations of experimental results (e.g., 25 studies examined in Luce et al., 1963) are beyond the scope of this paper.

Minimax Learning

So far, we have assumed the goal of the decision making network was to maximize the total expected utility over the environmental distribution. We now turn to the other well known framework for decision making: we would like the network to maximize the minimum expected utility over the set of possible decisions (e.g., von Neumann & Morgenstern, 1944).

The architecture of the decision network presented in Section 2 is still valid since it simply provides a representation for probabilities and utilities (usually written as costs in game setups) and a formalism for computing the gradient of a given response expected utility. The difference is now in the definition of optimality (*minimax* for costs, equivalently *maximin* for utilities) and in the resulting learning algorithm.

Back-propagation for Maximum Minimum Utility

We would now like the *p-units* to compute probabilities maximizing the minimum of expected utilities over a set of possible responses. For any given input pattern, the expected utility associated with each response is $E[U_i] = \sum_j u_{ij} p_j$. Learning then consists in gradient ascent on the minimum expected utility computed over the set of responses:

$$\Delta w_q = \eta \frac{\partial min_i\{E[U_i]\}}{\partial w_q} \qquad (11)$$

The expression $min_i\{E[U_i]\}$ is not differentiable (but still continuous) with respect to the weights of the network. The proposed implementation consists in computing $E[U_t]$ for each training pattern and to check if $E[U_t] = min_i\{E[U_i]\}$. If $E[U_t] = min_i\{E[U_i]\}$, the gradient of $E[U_t]$ is computed (Section 2) and weights are changed accordingly. If $E[U_t] \neq min_i\{E[U_i]\}$, weights remain unchanged.

Theoretical minimax probabilities usually correspond to mixed decisions ($p_j \neq 0, 1$) function of the cost (utility) matrix. These probabilities do not depend on environmental distributions. However, if network resources are limited (limited number of weights in the feature-extraction network), the computations of the network minimax probabilities might depend on the training distribution, justifying the relevance of a training sample. Again, the consequences of decision strategies on feature extraction are not examined in this paper.

	Net. Resp. 1	Net. Resp. 2
Env. Resp. 1	1	-2
Env. Resp. 2	-3	3

Table 1: Cost/utility matrix for the two-response and game situations as a function of environmental and network responses.

Example Implementation

The minimax-back-propagation procedure was implemented in a two-response situation using the cost/utility matrix shown in Table 1. The theoretical minimax equilibrium is found for $p_1 = p = 5/9$ and $p_2 = 1 - p = 4/9$. Simulations show the algorithm always converges to the optimal minimax probabilities, independently of initial p_j. The non-differentiability of the algorithm creates an angular point when p reaches the desired equilibrium but the network remains stable around equilibrium point. This behavior is described in more detail in the following situation.

Game Learning

The minimax-back-propagation algorithm was implemented in a two-player zero-sum game playing situation. The cost/utility matrix of Table 1 was used for player 1 and its opposite for player 2. The well-known *minimax theorem* (von Neumann & Morgenstern, 1944) shows there always exists a minimax equilibrium for such games. With the given cost/utility matrix, the minimax equilibria are found for the mixed strategies $p^1 = p_1^1 = 5/9$ for player 1 and $p^2 = p_1^2 = 2/3$ for player 2.

The game was implemented using two networks playing each other, simultaneously trained with the minimax-back-propagation algorithm. The "environmental" probability of one network/player now corresponds to the response probability of the other network/player: $p_i^{*1} = p_j^2, p_i^{*2} = p_j^1$. Figure 3 shows p^1 and p^2 as a function of the number of plys played by the networks for various initial conditions.

Simulations show the networks always converge to optimal minimax response behavior. When equilibria are reached, the response dynamics present an angular point but the equilibria remain stable in all cases. Furthermore, the network/player learning dynamics show interdependence since the response probabilities of one network/player correspond to the environmental prob-

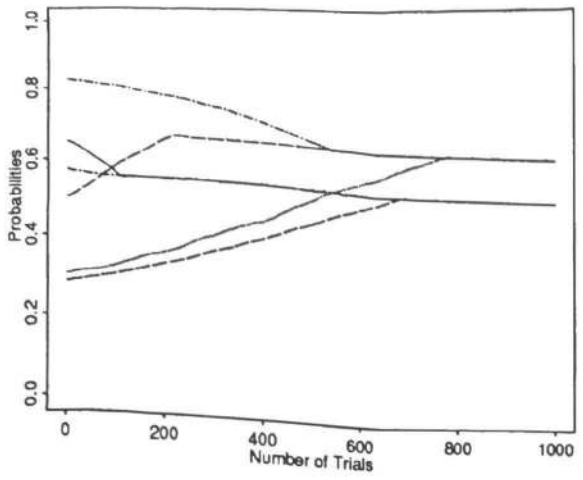

Figure 3: Examples of learning dynamics for the minimax-back-propagation algorithm during a game playing situation. Both network/players reach and remain at optimal minimax probabilities (5/9 and 2/3) independently of initial conditions (3 runs are shown, one per line type).

abilities of the other network/player. This result provides interesting hypotheses and predictions on learning dynamics where players learn from each other's playing strategy. The implications on observed experimental data in game learning situations are beyond the scope of this paper.

Conclusion

A general decision making network architecture is suggested along with representations for probabilities and utilities. The involved principles integrate formal properties of optimal decision making and connectionist computational characteristics. Variants of the back-propagation learning algorithm are then obtained to maximize total and minimum expected utility. The dynamic properties of the resulting algorithms are examined through simulations of a two-response environment.

Obviously, the complex properties of human decision making are barely mentioned in this paper. Nonetheless, in the case of simultaneous utility estimation and behavior adaptation, the approach provided interesting insights on observed experimental data. In general, the proposed architecture seems able to provide a rich computational framework which might lead to a number of interesting hypotheses and predictions about human learning and decision making behavior.

Acknowledgements. Thanks to David Rumelhart for helpful discussions and to Julie Holmes for her help throughout this project.

References

Busemeyer, J. R. & Myung, I. J. (1989). An adaptive theory of human decision making. In D. Vickers & P. L. Smith (Eds.), *Human Information Processing: Measures, Mechanisms and Models.* North-Holland: Elsevier.

DeGroot, M. H. (1970). *Optimal Statistical Decisions.* N. Y.: McGraw-Hill.

Estes, W. K. (1987). Application of a cognitive-distance model to learning in a simulated travel task. *JEP: Learning, Memory and Cognition, 13,* 380–386.

Luce, R. D. (1959). *Individual choice behavior.* N. Y.: Wiley.

Luce, R. D. & Suppes, P. (1963). Preference, utility, and subjective probability. In R. D. Luce, R. R. Bush, & E. Galanter (Eds.), *Handbook of mathematical psychology (Vol. 3).* New York: Wiley.

Rumelhart, D. E., Hinton, G. E., & Williams, R. J. (1986). Learning internal representations by error propagation. In D. E. Rumelhart & J. L. McClelland (Eds.), *Parallel Distributed Processing: Explorations in the Microstructures of Cognition (Vol. 1).* Cambridge, MA: MIT Press.

Slovic, P., Lichtenstein, S., & Fischhoff, B. (1988). Decision making. In R. C. Atkinson, R. J. Hernstein, G. Lindsey, & R. D. Luce (Eds.), *Steven's handbook of experimental psychology.* New York: Wiley. 2nd. Edition

von Neumann, J. & Morgenstern, O. (1944). *Theory of games and economic behavior.* Princeton, N.J.: Princeton University Press.