Introduction to Sequence Learning

Ron Sun

CECS Department, University of Missouri
Columbia, MO 65211, USA

1 Introduction

Sequential behavior is essential to intelligence, and it is a fundamental part of human activities ranging from reasoning to language, and from everyday skills to complex problem solving. In particular, sequence learning is an important component of learning in many task domains — planning, reasoning, robotics, natural language processing, speech recognition, adaptive control, time series prediction, financial engineering, DNA sequencing, and so on.

Naturally, there are many different approaches towards sequence learning, resulting from different perspectives taken in different task domains. These approaches deal with somewhat differently formulated sequential learning problems (for example, some with actions and some without), and/or different aspects of sequence learning (for example, sequence prediction vs. sequence recognition).

Sequence learning is clearly a difficult task. More powerful algorithms for sequence learning are needed in all of these afore-mentioned domains. It is our view that the right approach to develop better techniques, algorithms, models, and theories is to first better understand the state of the art in different disciplines related to this topic. There seems to be a need to compare, contrast, and combine different existing techniques, approaches, and paradigms, in order to develop better and more powerful algorithms. Currently, existing techniques and algorithms include recurrent neural networks, hidden Markov models, dynamic programming, reinforcement learning, graph theoretical models, search based models, evolutionary computational models, symbolic planning models, production rule based models, and so on.

Especially important to this topic area is the fact that work on sequence learning has been going on in several different disciplines such as artificial intelligence, neural networks, cognitive science (human sequence learning, e.g., in skill acquisition), and engineering. We need to examine the field in a cross-disciplinary way and take into consideration all of these different perspectives on sequence learning. Thus, we need interdisciplinary gatherings that include researchers from all of these orientations and disciplines, beyond narrowly focused meetings on specialized topics such as reinforcement learning or recurrent neural networks.

A workshop on neural, symbolic, and reinforcement methods for sequence learning (co-chaired by Ron Sun and Lee Giles) was held on August 1st, 1999, preceding IJCAI'99, in Stockholm, Sweden. The following issues concerning sequence learning were raised and addressed:

(1) Underlying similarities and differences among different models, including (1.1) problem formulation (ontological issues), (1.2) mathematical comparisons, (1.3) task appropriateness, and (1.4) performance analysis and bounds.

(2) New and old models, and their capabilities and limitations, including (2.1) theory, (2.2) implementation, (2.3) performance, and (2.4) empirical comparisons in various domains.

(3) Hybrid models, including (3.1) foundations for synthesis or hybridization, and (3.2) advantages, problems, and issues of hybrid models.

(4) Successful sequence learning applications and future extensions, including (4.1) examples of successful applications, (4.2) generalization and transfer of successful applications, and (4.3) what is needed for further enhancing the performance of these successful applications.

In this book, we will be mostly concerned with items 1, 2 and 4, focusing on exploring similarities, differences, capabilities, and limitations of existing and new models, in relation to various task domains and various sequence learning problems.

2 Problem Formulations and Their Relationships

One aim of the present volume is to better understand different formulations of sequence learning problems.

With some necessary simplification, we can categorize various sequence learning problems that have been tackled into the following categories: (1) sequence prediction, in which we want to predict elements of a sequence based on the preceding element(s); (2) sequence generation, in which we want to generate elements of a sequence one by one in their natural order; and (3) sequence recognition, in which we want to determine if a sequence is a legitimate one according to some criteria; in addition, (4) sequential decision making involves selecting a sequence of actions, to accomplish a goal, to follow a trajectory, or to maximize (or minimize) a reinforcement (or cost) function that is normally the (discounted) sum of reinforcements (costs) that are received along the way (see Bellman 1957, Bertsekas and Tsitsiklis 1995).

These different sequence learning problems can be more precisely formulated as follows (assume a deterministic world for now):

- *Sequence prediction*: $s_i, s_{i+1}, ..., s_j \longrightarrow s_{j+1}$, where $1 \leq i \leq j < \infty$; that is, given $s_i, s_{i+1}, ..., s_j$, we want to predict s_{j+1}. When $i = 1$, we make predictions based on all of the previously seen elements of the sequence. When $i = j$, we make predictions based only on the immediately preceding element.
- *Sequence generation*: $s_i, s_{i+1}, ..., s_j \longrightarrow s_{j+1}$, where $1 \leq i \leq j < \infty$; that is, given $s_i, s_{i+1}, ..., s_j$, we want to *generate* s_{j+1}. (Put in this way, it is clear that sequence prediction and generation are essentially the same task.)
- *Sequence recognition*: $s_i, s_{i+1}, ..., s_j \longrightarrow$ *yes or no*, where $1 \leq i \leq j < \infty$; that is, given $s_i, s_{i+1}, ..., s_j$, we want to determine if this subsequence is

legitimate or not. (There are alternative ways of formulating the sequence recognition problem, for example, as an one-shot recognition process, as opposed to an incremental step-by-step recognition process as formulated here.)

With this formulation, sequence recognition can be turned into sequence generation/prediction, by basing recognition on prediction (see the chapter by D. Wang in this volume); that is, $s_i, s_{i+1}, ..., s_j \longrightarrow yes$ (a recognition problem), if and only if $s_i, s_{i+1}, ..., s_{j-1} \longrightarrow s_j^p$ (a prediction problem) and $s_j^p = s_j^a$, where s_j^p is the prediction and s_j^a is the actual element.

Sequence learning (either generation, prediction, or recognition) is usually based on models of legitimate sequences, which can be developed through training with exemplars. Models may be in the form of Markov chains, hidden Markov models, recurrent neural networks, and a variety of other forms. Expectation-maximization, gradient descent, or clustering may be used in training (see e.g. the chapters by Oates and Cohen and by Sebastiani et al in this volume). Such training may extract "central tendencies" from a set of exemplars.

– *Sequential decision making* (that is, sequence generation through actions): there are several possible variations. In the goal oriented case, we have $s_i, s_{i+1}, ..., s_j; s_G \longrightarrow a_j$, where $1 \leq i \leq j < \infty$; that is, given $s_i, s_{i+1}, ..., s_j$ and the goal state s_G, we want to choose an action a_j at time step j that will likely lead to s_G in the future. In the trajectory oriented case, we have $s_i, s_{i+1}, ..., s_j; s_{j+1} \longrightarrow a_j$, where $1 \leq i \leq j < \infty$; that is, given $s_i, s_{i+1}, ..., s_j$ and the desired next state s_{j+1}, we want to choose an action a_j at time step j that will likely lead to s_{j+1} in the next step. In the reinforcement maximizing case, we have $s_i, s_{i+1}, ..., s_j \longrightarrow a_j$, where $1 \leq i \leq j < \infty$; that is, given $s_i, s_{i+1}, ..., s_j$ we want to choose an action a_j at time step j that will likely lead to receiving maximum total reinforcement in the future. The calculation of total reinforcement can be in terms of discounted or undiscounted cumulative reinforcement, in terms of average reinforcement, or in terms of some other functions of reinforcement (Bertsekas and Tsitsiklis 1996, Kaelbling et al 1996, Sutton and Barto 1997).

  The action selection can be based on the immediately preceding element (i.e., $i = j$), in which case a Markovian action policy is in force; otherwise, a non-Markovian action policy is involved (McCallum 1996, Sun and Sessions 2000). Yet another possibility is that the action decisions are not based on preceding elements at all (i.e., $s_i, s_{i+1}, ..., s_j$ are irrelevant), in which case an open-loop policy is used (Sun and Sessions 1998) while in the previous two cases, closed-loop policies are involved, as will be discussed later.

  Note that, on this view, sequence generation/prediction can be viewed as a special case of sequential decision making.

The above exposition reveals the relationship among different categories of sequence learning, under the assumption of a deterministic world. Another assumption in the above discussion is that we considered only closed-loop situations: that is, we deal only with one step beyond what is known or done. For open-loop prediction, generation, or action, we can have the following formalizations:

- *Sequence prediction*: $s_i, s_{i+1}, ..., s_j \longrightarrow s_{j+1}, ..., s_{j+k}$, where $1 \leq i \leq j < \infty$ and $k > 1$; that is, given $s_i, s_{i+1}, ..., s_j$, we want to be able to predict $s_{j+1}, ..., s_{j+k}$.
- *Sequence generation*: $s_i, s_{i+1}, ..., s_j \longrightarrow s_{j+1}, ..., s_{j+k}$, where $1 \leq i \leq j < \infty$ and $k > 1$.
- *Sequential decision making*: $s_i, s_{i+1}, ..., s_j; G \longrightarrow a_j, ..., a_{j+k}$, where $1 \leq i \leq j < \infty$ and $k > 1$; that is, given $s_i, s_{i+1}, ..., s_j$ and a goal state, a goal trajectory, or a reinforcement maximizing goal, we want to choose actions a_j through a_{j+k} that may lead to that goal.

The formulation of sequence recognition (either incremental or one-shot) is not changed, as sequence recognition has nothing to do with being either closed-loop or open-loop.

To extend our discussion from a deterministic world to a stochastic world, we may specify a probability distribution in place of a deterministic prediction/generation as adopted above. Predictions and generations can then be probabilistic: a distribution is calculated concerning an element in a sequence: $p(s_{j+1}|s_i.....s_j)$. Recognition may also be probabilistic: a probability $p(s_{j+1})$ may be calculated so as to determine how likely a sequence is a legitimate one. In cases where action is involved, the transition to a new element (i.e., a new state) after an action is performed can be stochastic (determined by a probability distribution): $p_{a_j}(s_j, s_{j+1})$, where action a_j is performed in state s_j at step j, and s_{j+1} is a new state resulting from the action. The action selection in such cases can also be probabilistic, whereby an action is selected from an action probability distribution $p(s_j, a_j)$, where s_j is a state and a_j is an action, In such cases, we may find the best stochastic action policy, i.e., the optimal action probability distribution in each state.

In addition to addressing these types of tasks, there are also other models that address additional issues arising out of, or along with, these above categories of sequence learning tasks. For example, we may want to segment a sequence (i.e., perform *sequential segmentation*) for the sake of compressing the description of sequences (Nevill-Manning and Witten 1997), or for the sake of better dealing with temporal dependencies (Sun and Sessions 2000, McCallum 1996). We may even form hierarchies of sequential segmentation (see the chapter by Sun and Sessions in this volume). We may also create modular structures during sequence learning (i.e, perform *modularization*) to facilitate learning processes, such as in Singh (1994), Tham (1995), Thrun and Schwartz (1995), and Sun and Peterson (1999). In addition, sequence filtering is also a common task.

3 Characterizing Sequence Learning Models

There are many existing models for different types of sequence learning discussed above. They vary greatly in terms of applicable domain, emphasis, or learning algorithm. We can characterize them along several major dimensions as follows:

- learning paradigm (whether supervised, unsupervised, reinforcement based, or knowledge based learning is used)
- implementation paradigm (whether neural networks, symbolic rules, or lookup tables are used for implementation)
- whether the world is probabilistic or deterministic (in terms of the determination of next elements)
- whether the world is Markovian or non-Markovian
- whether the task is closed-loop or open-loop
- whether action is involved or not
- whether an action policy is deterministic or stochastic (when action is selected)
- applicable domains — e.g., market agents (see the chapter by Tesauro), speech and language processing (e.g., Wermter et al 1996), navigation learning (e.g., Sun and Peterson 1998, 1999), or motor sequence learning (see the chapters by Bapi and Doya and by Grossberg and Paine).

Let us briefly review and compare several major approaches to sequence learning, such as recurrent neural networks, symbolic planning, dynamic programming (value iteration or policy iteration), and reinforcement learning (such as Q-learning), while keeping in mind these afore-mentioned dimensions.

First of all, there are a number of neural network models for dealing with sequences, such as recurrent backpropagation networks (see the review chapters by Sona and Sperduti and by Chappelier et al). Such networks typically use hidden nodes as memory, which represents a condensed record of the previously encountered subsequence. The hidden nodes are connected back to the input nodes and are used in subsequent time steps, so that the previous states are taken into account. Recurrent backpropagation networks are widely used for sequence prediction and generation (Frasconi et al 1995, Giles et al 1995). A potential shortcoming of such networks is that they require supervised learning and therefore may not be suitable for typical sequential decision making learning scenarios (whereby there is no teacher input, beside sparse reinforcement signals). There have also been associative networks proposed for sequence learning (see, e.g., Guyon et al 1988, Kleinfeld and Sompolinsky 1988). In this volume, Baldi et al, Sona and Sperduti, and Chappelier et al discussed various forms of recurrent backpropagation networks. Wang discussed more structured networks of his own design for sequence learning.

A good solution for learning sequential decision making, as well as sequence prediction/generation, is the temporal difference method, as described by e.g. Sutton and Barto (1997) and Watkins (1989). In general, there is an evaluation function and/or an action policy. The evaluation function generates a value,

$e(x)$, for a current state (input) x, which measures the goodness of x. An action a is chosen according to a certain action policy, based on the evaluation function $e(x)$. An action performed leads to a new state y and a reinforcement signal r. We then modify (the parameters of) the evaluation function so that $e(x)$ is closer to $r + \gamma e(y)$, where $0 < \gamma < 1$ is a discount factor. At the same time, the action policy may also be updated, to strengthen or weaken the tendency to perform the chosen action a, according to the error in evaluating the state: $r + \gamma e(y) - e(x)$. That is, if the situation is getting better because of the action, increase the tendency to perform that action; otherwise, reduce the tendency. The learning process is dependent upon the temporal difference in evaluating each state. *Q-learning* (Watkins 1989) is a variation of this method, in which the policy and the evaluation function are merged into one function $Q(x, a)$, where x is the current state and a is an action. Much existing work shows that Q-learning is as good as any other reinforcement learning algorithms (Lin 1992, Tesauro 1992). However, reinforcement learning is problematic when the input state space is continuous or otherwise large in which case reinforcement learning using table lookup is no longer applicable, and connectionist implementations are not guaranteed successful learning (Lin 1992, Tesauro 1992). In this volume, the chapters by Schmidhuber, by Sun and Sessions, by Choi et al, and by Tesauro are concerned with various extensions of reinforcement learning, for dealing with non-Markovian dependencies, co-learning in multi-agent systems, and so on.

Another approach for sequential decision making is explicit symbolic planning as developed in artificial intelligence. From an overall goal, a number of subgoals are generated that are to be accomplished in a certain order; then, from each of these subgoals, a number of subsubgoals are generated, and so on, until each of these goals can be accomplished in one step. Subgoals are generated based on domain knowledge. Alternatively, partial ordering of primitive actions can be incrementally established from known domain knowledge, and then a total ordering of actions is produced based on partial ordering, which constitutes a plan for action sequencing. The shortcomings of the planning approach are the following: (1) a substantial amount of prior knowledge is required in this approach, which is not always readily available; (2) a high computational complexity is inherent in this approach; (3) symbolic planning may be unnatural for describing some simple reactive sequential behavior, as pointed out by advocates of situated action (e.g., Agre and Chapman 1990). Recently, Inductive Logic Programming techniques (Lavrac and Dzeroski 1994) have been developed, which can be used to learn symbolic knowledge from exemplars and thus partially remedies one problem of symbolic planning.

Hidden Markov models (HMM) can be used for sequence learning (including generation and recognition; Baum 1972). This type of model learns underlying state transition probability distributions from which overt observation data are generated. Sequence generation can evidently be handled by this type of model. Sequence recognition can also be accomplished based on hidden Markov models by computing probabilities of generating sequences from underlying models. In this volume, a number of chapters deal with extending and enhancing known

HMM algorithms, by considering, e.g., clustering of HMMs (as in Oates and Cohen), multiple HMMs with transitions among them (as in Choi et al), or incorporating input/output relations and bidirectional (forward and backward) predictions (as in Baldi et al).

Another approach suitable for a variety of types of sequence learning is evolutionary computation (EC). EC is a "weak" method for knowledge-lean heuristic search. It updates its knowledge based (mostly) on the acquired experience of an entire generation (each member of which goes through many training trials) at the end of all their trials. It can be used to tackle all types of sequence learning because of the generality of this learning method (see, e.g., Grefenstette 1992, Meeden 1995, as well as the chapter by Schmidhuber in this volume).

There are also other ways in which sequence learning can be performed, extensions and variations of existing methods can be explored, and advances beyond existing work can be made (see, e.g., Wang and Alkon 1993, Giles and Gori 1998).

Given the variety of different approaches for sequence learning, we may compare, and combine, some of these approaches. In this volume, the chapter by Sona and Sperduti and, to a lesser extent, the chapter by Chappelier et al compare (informally) a variety of different models for sequence learning, and illustrate their respective strengths and weaknesses. Clearly, these different techniques are suitable for somewhat different learning situations, and a clear and formal understanding of this aspect is yet to be achieved. In terms of combining different techniques to form some sorts of hybrid models, there are clearly many possibilities, for example, combining symbolic rules and neural networks for implementing reinforcement learning (Sun and Peterson 1998), or combining symbolic planning and reinforcement learning to produce action sequences (Dearden and Boutilier 1997, Sun and Sessions 1998). There are also proposals for combining EC and recurrent neural networks for sequence learning. Hybridization is definitely an issue worthy of further consideration.

4 Applications

A number of applications of sequence learning are reported in this volume. For example, Baldi et al discussed protein secondary structure prediction. Sabastiani et al focused on robot sensory processing. Bapi and Doya analyzed motor sequence learning. Tesauro explored automatic pricing in e-commerce. Zaki dealt with data mining in large databases of sequential data.

There are, of course, many other areas of applications for sequence learning that are not covered by the present volume due to space limitation. In fact, sequence learning is arguably the most prevalent form of human and animal learning. In classical studies of instrumental conditioning, sequencing is central (Anderson 1995). In human skill learning, sequentiality is often the most important aspect (Willingham et al 1989, Sun et al 2000; see also the chapter by Lebiere and Wallach in this volume). In human high-level problem solving and reasoning, sequences are also essential (Anderson 1995). Therefore, when building intelligent systems to emulate human intelligence and cognition, we must pay serious attention to sequences, including sequence learning.

5 Further Issues

One difficult issue in sequence learning is temporal (non-Markovian) dependencies, especially long-range dependencies. A current situation may be dependent on what happened before, and sometimes on what happened a long time ago. Many existing models have difficulties dealing with such temporal dependencies. For example, it is well known that many recurrent neural networks models cannot deal well with long-range dependencies (see the chapter by Chappelier et al). Dealing with long-range dependencies is even harder for reinforcement learning. Many heuristic methods (e.g., McCallum 1996, Sun and Sessions 2000) may help to facilitate learning of temporal dependencies to some extent, but they break down in cases of long-range dependencies. The methods described in the chapters by Zaki and by Schmidhuber may have some potential in dealing with this issue.

Another issue in sequence learning is hierarchical structuring of sequences. Many real-world problems give rise to sequences that have clear hierarchical structures: a sequence is made up of subsequences and they in turn are made up of subsubsequences, and so on. The difficulty lies in how we automatically identify these subsequences and deal with them accordingly. This issue is somewhat related to the issue of temporal dependencies, as hierarchical structures are often determined based on temporal dependencies. Learning hierarchical structures may help to reduce or even eliminate temporal dependencies. It may also help to compress the description of sequences. For learning hierarchical sequences with no action involved, there are some existing methods for segmentation (e.g., Nevill-Manning and Witten 1997). For learning hierarchical action sequences, the reinforcement learning methods developed in the chapter by Sun and Sessions showed some potential.

Yet another issue concerns complex or chaotic sequences often encountered in real-world applications such as time series prediction or neurobiological modeling (Wang and Alkon 1993). There has been a substantial amount of work in this area. However, not enough understanding has been achieved. Much more work is needed and further progress is expected in this area. The same can be said about hybridization of existing sequence learning techniques as mentioned earlier.

6 Concluding Remarks

The importance of sequential behavior and sequence learning in intelligence and cognition cannot be over-estimated. The mistake of earlier AI and Machine Learning has been downplaying the role of sequential behavior and sequence learning (which has been corrected to some large extent recently). We hope that this book serves to emphasize, and bring to the attention of the scientific and engineering communities, the importance of this problem. In this introduction, I tried to present a unified view of a variety of sequence learning formulations, paradigms, and approaches, to give some coherence to this topic. This book, I hope, is the first step toward establishing a more unified, more coherent conceptual framework for the study of sequence learning of various types.

References

P. Agre and D. Chapman, (1990). What are plans for? In: P. Maes, ed. *Designing Autonomous Agents.* Elsevier, New York.

J. Anderson, (1995). *Learning and Memory.* John Wiley and Sons, New York.

L. E. Baum, (1972). An inequality and associated maximization technique in statistical estimation for probabilistic functions of a Markov process. *Inequalities*, 3, 1-8.

R. Bellman, (1957). *Dynamic Programming.* Princeton University Press, Princeton, NJ.

D. Bertsekas and J. Tsitsiklis, (1996). *Neuro-Dynamic Programming.* Athena Scientific, Belmont, MA.

R. Dearden and Boutilier, (1997). Abstraction and approximate decision theoretic planning. *Artificial Intelligence.* 89, 219-283.

P. Frasconi, M. Gori, and G. Soda, (1995). Recurrent neural networks and prior knowledge for sequence processing. *Knowledge Based Systems.* 8, 6, 313-332.

C.L. Giles and M. Gori, (eds.) (1998). *Adaptive Processing of Sequences and Data Structures.* Springer, Heidelberg, Germany.

C.L. Giles, B.G. Horne, and T. Lin, (1995). Learning a class of large finite state machines with a recurrent neural network. *Neural Networks*, 8 (9), 1359-1365.

J. Grefenstette, The evolution of strategies for multiagent environments. *Adaptive Behavior.* 1(1). 65-90.

I. Guyon et al, (1988). Storage and retrieval of complex sequences in neural networks. *Physical Review A*, 38, 6365-6372.

L. Kaelbling, M. Littman, and A. Moore, (1996). Reinforcement learning: a survey. *Journal of Artificial Intelligence Research*, 4, 237-285.

D. Kleinfeld and H. Sompolinsky, (1988). Associative neural network models for the generation of temporal patterns. *Biophysics Journal*, 54, 1039-1051.

N. Lavrac and S. Dzeroski, (1994). *Inductive Logic Programming.* Ellis Horword, New York.

L. Lin, (1992). Self-improving reactive agents based on reinforcement learning, planning, and teaching. *Machine Learning.* 8, pp.293-321.

A. McCallum, (1996). Learning to use selective attention and short-term memory in sequential tasks. *Proc. Conference on Simulation of Adaptive Behavior.* 315-324. MIT Press, Cambridge, MA.

L. Meeden, (1995). An incremental approach to developing intelligent neural network controllers for robots. *Adaptive Behavior.*

C. Nevill-Manning and I. Witten, (1997). Identifying hierarchical structure in sequences: a linear-time algorithm. *Journal of Artificial Intelligence Research*, 7, 67-82.

S. Singh, (1994). *Learning to Solve Markovian Decision Processes.* Ph.D Thesis, University of Massachusetts, Amherst, MA.

R. Sun, E. Merrill, and T. Peterson, (2000). From implicit skills to explicit knowledge: a bottom-up model of skill learning. *Cognitive Science*, in press.

R. Sun and T. Peterson, (1998). Autonomous learning of sequential tasks: experiments and analyses. *IEEE Transactions on Neural Networks*, 9, 6, pp.1217-1234.

R. Sun and T. Peterson, (1999). Multi-agent reinforcement learning: weighting and partitioning. *Neural Networks*, 12, 4-5. pp.127-153.

R. Sun and C. Sessions, (1998). Learning plans without a priori knowledge. *Proceedings of WCCI-IJCNN'98*, vol.1, 1-6. IEEE Press, Piscataway, NJ. A longer version to appear in: *Adaptive Behavior.*

R. Sun and C. Sessions, (2000). Self-segmentation of sequences: automatic formation of hierarchies of sequential behaviors. *IEEE Transactions on Systems, Man, and Cybernetics: Part B Cybernetics*, Vol.30, No.3, 403-418.

R. Sutton and A. Barto, (1997). *Reinforcement Learning.* MIT Press, Cambridge, MA.

T. Tesauro, (1992). Practical issues in temporal difference learning. *Machine Learning.* 8, 257-277.

C. Tham, (1995). Reinforcement learning of multiple tasks using a hierarchical CMAC architecture. *Robotics and Autonomous Systems.* 15, 247-274.

S. Thrun and A. Schwartz, (1995). Finding structure in reinforcement learning. *Neural Information Processing Systems 7*, MIT Press, Cambridge, MA.

L. Wang and D. Alkon, (eds.) (1993). *Artificial Neural Networks: Oscillations, Chaos, and Sequence Processing.* IEEE Computer Society Press, Los Alamitos, CA.

C. Watkins, (1989). *Learning with Delayed Rewards.* Ph.D Thesis, Cambridge University, Cambridge, UK.

S. Wermter et al, (ed.) (1996). *Connectionist and Statistical Approaches to Natural Language Processing.* Springer, Heidelberg, Germany.

D. Willingham, M. Nissen, and P. Bullemer, (1989). On the development of procedural knowledge. *Journal of Experimental Psychology: Learning, Memory, and Cognition.* 15, 1047-1060.