Sequence Learning: From Recognition and Prediction to Sequential Decision Making

Ron Sun, *University of Missouri, Columbia*
C. Lee Giles, *Pennsylvania State University*

Sequence learning is arguably the most prevalent form of human and animal learning. Sequences play a pivotal role in classical studies of instrumental conditioning,[1] in human skill learning,[2] and in human high-level problem solving and reasoning.[1]

So, it's logical that sequence learning is an important component of learning in many task domains of intelligent systems: inference, planning, reasoning, robotics, natural language processing, speech recognition, adaptive control, time series prediction, financial engineering, DNA sequencing, and so on. Naturally, the unique perspectives of these domains lead to different sequence-learning approaches. These approaches deal with somewhat differently formulated sequence-learning problems (for example, some with actions and some without) and with different aspects of sequence learning (for example, sequence prediction versus sequence recognition).

Despite the plethora of approaches, sequence learning is still difficult. We believe that the right approach to improving sequence learning is to first better understand the state of the art in the different disciplines related to this topic. This requires comparing, contrasting, and combining the existing techniques, approaches, and paradigms, to develop better, more powerful algorithms. Toward that end, we present here a brief tutorial on sequence learning.

Problem formulations and their interrelationships

With some necessary simplification, we can categorize various sequence-learning problems into four basic categories:

- *Sequence prediction* attempts to predict elements of a sequence on the basis of the preceding elements.
- *Sequence generation* attempts to generate elements of a sequence one by one in their natural order.
- *Sequence recognition* attempts to determine if a sequence is legitimate according to some criteria.
- *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) during the course of actions.[3,4]

We can more precisely formulate these four categories. First, we assume a deterministic world. To start, we examine the first three categories (in each case, $1 \leq i \leq j < \infty$):

- *Sequence prediction*: $s_i, s_{i+1}, \ldots, s_j \rightarrow s_{j+1}$. That is, given $s_i, s_{i+1}, \ldots, s_j$, we want to predict s_{j+1}. When $i = 1$, we make predictions based on all the previously seen elements of the sequence. When $i = j$, we make predictions based on only the immediately preceding element.
- *Sequence generation*: $s_i, s_{i+1}, \ldots, s_j \rightarrow s_{j+1}$. That is, given $s_i, s_{i+1}, \ldots, s_j$, we want to generate s_{j+1}. (As you can see, sequence prediction and generation are essentially the same.)
- *Sequence recognition*: $s_i, s_{i+1}, \ldots, s_j \rightarrow$ *yes or no*. That is, given $s_i, s_{i+1}, \ldots, s_j$, we want to determine whether this subsequence is legitimate. (Of course, we can also formulate the sequence recognition problem in other ways—for example, as a one-shot process, as opposed to the incremental process formulated here.)

We can turn sequence recognition into sequence generation or prediction by basing recognition on prediction. That is, $s_i, s_{i+1}, \ldots, s_j \rightarrow$ *yes* (a recognition problem), if and only if $s_i, s_{i+1}, \ldots, s_{j-1} \rightarrow s_j^p$ (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 might be in the form of Markov chains, hidden Markov models, recurrent neural networks, or a variety of other forms. The training might employ expectation maximization, gradient descent, or clustering. Such training might extract "central tendencies" from a set of exemplars.

The fourth category, sequential decision making (that is, sequence generation through actions) has three main variations:

- *Goal-oriented*: $s_i, s_{i+1}, \ldots, s_j; s_G \rightarrow a_j$. That is, given $s_i, s_{i+1}, \ldots, s_j$ and the goal state s_G, we want to choose an action a_j at time step j that will lead to s_G in the future.
- *Trajectory-oriented*: $s_i, s_{i+1}, \ldots, s_j; s_{j+1} \rightarrow a_j$. That is, given $s_i, s_{i+1}, \ldots, s_j$ and the desired next state s_{j+1}, we want to choose an action a_j at time step j that will lead to s_{j+1} in the next step.
- *Reinforcement-maximizing*: $s_i, s_{i+1}, \ldots, s_j \rightarrow a_j$. That is, given $s_i, s_{i+1}, \ldots, s_j$, we want to choose an action a_j at time step j that will lead to receiving maximum total reinforcement in the future. The calculation of total reinforcement can be in terms of discounted or undiscounted cumulative reinforcement, average reinforcement, or some other reinforcement functions.[4–6]

If the action selection is based on the immediately preceding element, a Markovian action policy is in force. If it involves some other preceding elements, a non-Markovian action policy is involved.[7,8]

From this viewpoint, sequence generation can be seen as a special case of sequential decision making.

So far, we have considered primarily closed-loop situations for the four categories. For open-loop situations in a deterministic world, we use the following formalizations (in each case, $1 \leq i \leq j < \infty$ and $k > 1$):

- *Sequence prediction*: $s_i, s_{i+1}, \ldots, s_j \rightarrow s_{j+1}, \ldots, s_{j+k}$. That is, given $s_i, s_{i+1}, \ldots, s_j$, we want to be able to predict $s_{j+1}, \ldots, s_{j+k}$.
- *Sequence generation*: $s_i, s_{i+1}, \ldots, s_j \rightarrow s_{j+1}, \ldots, s_{j+k}$. That is, given $s_i, s_{i+1}, \ldots, s_j$, we want to generate $s_{j+1}, \ldots, s_{j+k}$.
- *Sequential decision making*: $s_i, s_{i+1}, \ldots, s_j; G \rightarrow a_j, \ldots, a_{j+k}$. That is, given $s_i, s_{i+1}, \ldots, s_j$ and a goal state, a goal trajectory, or a reinforcement-maximizing goal, we want to choose actions a_j through a_{j+k} that might lead to that goal. In this case, the action selection is not based on preceding elements $s_{j+1}, \ldots, s_{j+k}$.

The formulation of sequence recognition (either one-shot or incremental) does not change in the case of open-loop settings, because sequence recognition has nothing to do with being closed- or open-loop.

To extend our discussion from a deterministic world to a stochastic one, we can specify a probability distribution in place of the deterministic prediction or generation. For prediction and generation, we calculate a distribution concerning an element in a sequence: $p(s_{j+1} \mid s_i \ldots s_j)$. Recognition can also be probabilistic: we can calculate the probability $p(s_{j+1})$ of whether a sequence is legitimate. Where action is involved, the transition to a new element (that is, a new state) after an action is performed can be stochastic (determined by a probability distribution): $p_{a_j}\left(s_j, s_{j+1}\right)$, where action a_j is performed in state s_j at step j, and the new state resulting from the action is s_{j+1}. We can also select an action from an action probability distribution $p(s_j, a_j)$. In this case, we might want to find the best stochastic action policy—that is, the optimal action probability distribution—in each state.

These four categories give rise to other sequence-learning tasks that require additional work. For example, we might want to segment a sequence to compress the description of sequences[9] or to deal better with temporal dependencies.[7,8] We might even form hierarchies of sequential segmentation or create modular structures during sequence learning to facilitate learning processes.[10,11] Sequence filtering is also common.

Characterizing sequence-learning models

We can characterize sequence-learning models along these dimensions:

- learning paradigms—for example, supervised, unsupervised, reinforcement-based, or knowledge-based;
- implementation paradigms—for example, neural networks, symbolic rules, or lookup tables;
- whether the world is deterministic or probabilistic (in terms of the determination of next elements);
- whether the world is Markovian or non-Markovian;
- whether the task is closed- or open-loop;
- whether action is involved;
- whether an action policy is deterministic or stochastic (if action is involved);
- applicable domains—for example, software agents, speech and language processing, navigation learning, or motor sequence learning.

Let us briefly examine several major approaches to sequence learning, while keeping in mind these dimensions.

Several neural network models deal with sequences; one example is *recurrent backpropagation networks*. These networks typically use hidden nodes as memory, which represents a condensed record of the previously encountered subsequence. The hidden nodes connect back to the input nodes and are used in subsequent time steps, so the network takes into account the previous states. Recurrent backpropagation networks are widely used for sequence prediction and generation.[12,13] Because they require supervised learning, they might not be suitable for typical sequential decision-making situations (where there is no teacher input and sparse reinforcement signals). Associative networks have also been proposed for sequence learning.[14,15]

A good solution for sequential decision making, as well as sequence prediction and generation, is the *temporal-difference method*.[6,16] Generally, this method involves an evaluation function, an action policy, or both. The evaluation function generates a value, $e(x)$, for a current state (input) x, which measures the goodness of x. The method chooses an action a according to a certain action policy, on the basis of the evaluation function $e(x)$. A performed action 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, we might also update the action policy to strengthen or weaken the tendency to perform a, according to the error in evaluating the state: $r + \gamma e(y) - e(x)$. That is, if the action improves the situation, increase the tendency to perform that action; otherwise, reduce the tendency. The learning process depends on the temporal difference in evaluating each state.

Q-learning, a variation of this method, merges the policy and the evaluation function into one function $Q(x, a)$, where x is the current state and a is an action.[16] Considerable research shows that Q-learning is as good as any other reinforcement-learning algorithm.[17,18] However, reinforcement learning is problematic when the input state space is continuous or otherwise large. In that case, reinforcement learning using table lookup is no longer applicable, and connectionist implementations are not guaranteed successful learning.[17,18]

Another method for sequential decision making is *explicit symbolic planning* as developed in AI. From an overall goal, this method generates a number of subgoals to accomplish in a certain order. Then, from each of these subgoals, a number of sub-

The Sequence-Learning Workshop

Sequence-learning research has been ongoing in such different disciplines as artificial intelligence, neural networks, cognitive science (concerning human sequence learning, for example, as in skill acquisition), and engineering. So, we need to examine the field in a cross-disciplinary way and consider all these different perspectives.

Accordingly, interdisciplinary sequence-learning gatherings have gone beyond narrowly focused meetings on only a specialized topic such as reinforcement learning or recurrent neural networks. They include researchers from all the different orientations and disciplines.

One such meeting, the Workshop on Neural, Symbolic, and Reinforcement Methods for Sequence Learning, cochaired by Ron Sun and Lee Giles, took place in Stockholm on 1 August 1999, preceding the International Joint Conference on AI. The workshop addressed these issues:

- Similarities and differences among different models, including problem formulation (ontological issues), mathematical comparisons, task appropriateness, and performance analysis and bounds.
- New and old models and their capabilities and limitations, including theory, implementation, performance, and empirical comparisons in various domains.
- Hybrid models, including their theories, approaches, and applications, including foundations for synthesis or hybridization.
- Successful sequence-learning applications and future extensions, including generalization and transfer of successful applications, and requirements for further enhancing their performance.

Researchers presented 17 papers (six invited and 11 submitted). Discussions covered mostly the first, second, and fourth bulleted items. Here are some highlights from the conference:

- Several talks dealt with extending and enhancing known hidden Markov model algorithms. Tim Oates and Paul Cohen discussed clustering of HMMs. Samuel Choi and his colleagues presented their "hidden-mode" Markov decision process models, which use multiple HMMs with transitions among them. Pierre Baldi and his colleagues presented BioHMMs, which incorporate input–output relations and bidirectional (forward and backward) predictions.
- Various extensions of reinforcement learning were also discussed—for example, by Choi and his colleagues and by Sun and Chad Sessions.
- Alessandro Sperduti compared informally a variety of different models and illustrated their strengths and weaknesses.
- Paolo Frasconi and Sperduti talked about different forms of recurrent backpropagation networks.
- Many sequence-learning applications were reported. For example, Baldi and his colleagues discussed protein secondary structure prediction. Paola Sebastiani and her colleagues focused on robot sensory processing. Raju Bapi and Kenji Doya analyzed motor sequence learning. And Mohammad Zaki dealt with data mining in large databases of sequential data.
- Manuela Veloso discussed, in depth, sequence learning in mobile robot teams.

subgoals are generated, and so on, until each of these goals can be accomplished in one step. Subgoal generation is based on domain knowledge. Alternatively, the method can incrementally establish a partial ordering of primitive actions on the basis of domain knowledge. Then, it produces a total ordering of actions on the basis of partial ordering; this total ordering constitutes a plan for action sequencing. However, this approach has these shortcomings:

- It requires substantial prior knowledge, which is not always readily available.
- Its computational complexity is high.
- It might be unnatural for describing some simple reactive sequential behavior, as advocates of situated action have pointed out.[19]

Inductive logic programming techniques[20] can be used to learn symbolic knowledge from exemplars; this partially remedies one problem of symbolic planning (that is, the problem of prior knowledge).

Hidden Markov models learn underlying state transition probability distributions from which overt observation data are generated by a certain process (for example, the speech process). So, HMMs can handle sequence generation.[21] They can also help accomplish sequence recognition by computing probabilities of generating sequences from underlying models.[21]

Evolutionary computation is a "weak" method for knowledge-lean heuristic search. EC updates its knowledge (mostly) on the basis of the acquired experience of an entire generation (each member of which goes through many training trials) at the end of all its trials. Its generality makes it useful for tackling all types of sequence learning.[22,23]

Many possibilities exist for combining different techniques to form hybrid models—for example, combining symbolic rules and neural networks for implementing reinforcement learning[24] or combining symbolic planning and reinforcement learning to produce action sequences.[25,26] Researchers have also proposed combining EC and recurrent neural networks.

Further issues

One significant issue in sequence learning is temporal (non-Markovian) dependencies. A current situation might depend on what has happened before and sometimes on what happened a long time ago. Many existing models have difficulties handling such dependencies. For example, dealing with long-range dependencies is hard for many recurrent neural network models[12] and even harder for reinforcement learning. Many heuristic methods might help facilitate learning of temporal dependencies somewhat,[7,8] but they also break down in cases of long-range dependencies.

Another issue is hierarchical structuring of sequences. Many real-world problems give rise to sequences that have clearly hierarchical structures: a sequence consists of subsequences that consist of sub-subsequences, and so on. The difficulty lies in how we identify and deal with these subsequences. This issue is somewhat related to that of temporal dependencies, because hierarchical structures are often determined on the basis of temporal dependencies. Learning hierarchical structures might help reduce or even eliminate temporal dependencies. It might also help compress the description of sequences. For learning hierarchical sequences with no action involved, some segmentation methods exist.[9] For learning action sequences, Ron Sun and Chad Sessions have developed reinforcement-learning methods that show potential for developing hierarchical action sequences.[8]

Yet another issue concerns complex or chaotic sequences often encountered in real-world applications such as time series prediction or neurobiological modeling.[27] This area has seen substantial research, which, unfortunately, has not generated enough understanding. Much more work is needed, and we should expect to see further progress. This is also true for hybridization of existing sequence-learning techniques.

The importance of sequential behavior and sequence learning in intelligence and cognition cannot be overestimated. Earlier AI and machine learning research mistakenly downplayed the role of sequential behavior and sequence learning. Recent research has corrected this to a large extent. We hope this introductory article brings this problem's importance to the attention of scientific and engineering communities. We have tried to present a unified, coherent view of a variety of sequence-learning formulations, paradigms, and approaches. This is the first step toward establishing a more unified and coherent conceptual framework for studying different types of sequence learning.

For further details on sequence learning, you might find our book *Sequence Learning: Paradigms, Algorithms, and Applications* useful.[28] See the book information page at www.cecs.missouri.edu/~rsun/book5-ann.html. In addition, the sidebar briefly describes a workshop we organized that attempted to provide a broad, cross-disciplinary look at sequence learning.

Ron Sun is an associate professor of computer engineering and computer science at the University of Missouri, Columbia. His research interest centers around intelligence and cognition, especially in the areas of commonsense reasoning, human and machine learning, and hybrid connectionist models. He is the coeditor in chief of *Cognitive Systems Research* and serves on the editorial boards of *Connection Science*, *Applied Intelligence*, and *Neural Computing Surveys*. He has been on the program committees of the National Conference on Artificial Intelligence, International Joint Conference on Neural Networks, and International Two-Stream Conference on Expert Systems and Neural Networks. He is a member of the AAAI and Cognitive Science Society and a senior member of the IEEE. He received his PhD from Brandeis University. Contact him at the CECS Dept., Univ. of Missouri, Columbia, MO 65211; rsun@cecs.missouri.edu; www.cecs.missouri.edu/~rsun.

C. Lee Giles is the David Reese Professor at the School of Information Sciences and Technology at Pennsylvania State University. His research interests are in basic applied research in intelligent information-processing systems. He has served or is serving on the editorial boards of *IEEE Intelligent Systems*, *IEEE Transactions on Knowledge and Data Engineering*, *IEEE Transactions on Neural Networks*, the *Journal of Computational Intelligence in Finance*, *Neural Networks*, and *Neural Computation*. He is a fellow of the IEEE and a member of the AAAI, ACM, International Neural Network Society, and American Association for the Advancement of Science. He received his PhD from the University of Arizona. Contact him at IST, Pennsylvania State Univ., University Park, PA 16801; giles@ist.psu.edu.

References

1. J. Anderson, *Learning and Memory*, John Wiley & Sons, New York, 1995.
2. R. Sun, E. Merrill, and T. Peterson, "From Implicit Skills to Explicit Knowledge: A Bottom-Up Model of Skill Learning," *Cognitive Science*, vol. 25, no. 2, 2001, pp. 203–244.
3. R. Bellman, *Dynamic Programming*, Princeton Univ. Press, Princeton, N.J., 1957.
4. D. Bertsekas and J. Tsitsiklis, *Neuro-Dynamic Programming*, Athena Scientific, Belmont, Mass., 1996.
5. L. Kaelbling, M. Littman, and A. Moore, "Reinforcement Learning: A Survey," *J. Artificial Intelligence Research*, vol. 4, 1996, pp. 237–285.
6. R. Sutton and A. Barto, *Reinforcement Learning*, MIT Press, Cambridge, Mass., 1997.
7. A. McCallum, "Learning to Use Selective Attention and Short-Term Memory in Sequential Tasks," *Proc. Conf. Simulation of Adaptive Behavior*, MIT Press, Cambridge, Mass.,1996, pp. 315–324.
8. R. Sun and C. Sessions, "Self Segmentation of Sequences," *IEEE Trans. System, Man, and Cybernetics*, vol. 30, no. 3, 2000, pp.403–418.
9. C. Nevill-Manning and I. Witten, "Identifying Hierarchical Structure in Sequences: A Linear-Time Algorithm," *J. Artificial Intelligence Research*, vol. 7, 1997, pp. 67–82.
10. R. Sun and T. Peterson, "Multi-agent Reinforcement Learning: Weighting and Partitioning," *Neural Networks*, vol. 12, nos. 4–5, 1999, pp. 127–153.
11. C. Tham, "Reinforcement Learning of Multiple Tasks Using a Hierarchical CMAC Architecture," *Robotics and Autonomous Systems*, vol. 15, 1995, pp. 247–274.
12. C.L. Giles and M. Gori, eds., *Adaptive Processing of Sequences and Data Structures*, Springer-Verlag, Heidelberg, Germany, 1998.
13. C.L. Giles, B.G. Horne, and T. Lin, "Learning a Class of Large Finite State Machines with a Recurrent Neural Network," *Neural Networks*, vol. 8 no. 9, 1995, pp. 1359–1365.
14. I. Guyon et al., "Storage and Retrieval of Complex Sequences in Neural Networks," *Physical Rev. A*, vol. 38, 1988, pp. 6365–6372.
15. D. Kleinfeld and H. Sompolinsky, "Associative Neural Network Models for the Generation of Temporal Patterns," *Biophysics J.*, vol. 54, 1988, pp. 1039–1051.
16. C. Watkins, *Learning with Delayed Rewards*, PhD thesis, Cambridge Univ., Cambridge, UK, 1989.
17. L. Lin, "Self-Improving Reactive Agents Based on Reinforcement Learning, Planning, and Teaching," *Machine Learning*, vol. 8, 1992, pp. 293–321.
18. T. Tesauro, "Practical Issues in Temporal Difference Learning," *Machine Learning*, vol. 8, 1992, pp. 257–277.
19. P. Agre and D. Chapman, "What Are Plans For?" *Designing Autonomous Agents*, P. Maes, ed., Elsevier, New York, 1990.
20. N. Lavrac and S. Dzeroski, *Inductive Logic Programming*, Ellis Horwood, New York, 1994.
21. L.E. Baum, "An Inequality and Associated Maximization Technique in Statistical Estimation for Probabilistic Functions of a Markov Process," *Inequalities*, vol. 3, 1972, pp. 1–8.
22. J. Grefenstette, "The Evolution of Strategies for Multiagent Environments," *Adaptive Behavior*. vol. 1, no. 1, 1992, pp. 65–90.
23. L. Meeden, "An Incremental Approach to Developing Intelligent Neural Network Controllers for Robots," *IEEE Trans. Systems, Man, and Cybernetics, Part B: Cybernetics*, vol. 26, no. 3, 1996, pp. 473–485.
24. R. Sun and T. Peterson, "Autonomous Learning of Sequential Tasks: Experiments and Analyses," *IEEE Trans. Neural Networks*, vol. 9, no. 6, Nov. 1998, pp. 1217–1234.
25. R. Dearden and C. Boutilier, "Abstraction and Approximate Decision Theoretic Planning," *Artificial Intelligence*, vol. 89, 1997, pp. 219–283.
26. R. Sun and C. Sessions, "Learning Plans without A Priori Knowledge," *Proc. 1998 IEEE World Congress Computational Intelligence and Int'l Joint Conf. Neural Networks* (WCCI-IJCNN '98), vol.1, IEEE Press, Piscataway, N.J., 1998, pp. 1–6.
27. L. Wang and D. Alkon, eds., *Artificial Neural Networks: Oscillations, Chaos, and Sequence Processing*, IEEE CS Press, Los Alamitos, Calif., 1993.
28. R. Sun and L. Giles, *Sequence Learning: Paradigms, Algorithms, and Applications*, Springer-Verlag, Heidelberg, Germany, 2001.