

Graded State Machines: The Representation of Temporal Contingencies in Simple Recurrent Networks

DAVID SERVAN-SCHREIBER, AXEL CLEEREMANS AND JAMES L. MCCLELLAND
School of Computer Science and Department of Psychology, Carnegie Mellon University

Abstract. We explore a network architecture introduced by Elman (1990) for predicting successive elements of a sequence. The network uses the pattern of activation over a set of hidden units from time-step $t-1$, together with element t , to predict element $t + 1$. When the network is trained with strings from a particular finite-state grammar, it can learn to be a perfect finite-state recognizer for the grammar. When the net has a minimal number of hidden units, patterns on the hidden units come to correspond to the nodes of the grammar; however, this correspondence is not necessary for the network to act as a perfect finite-state recognizer. Next, we provide a detailed analysis of how the network acquires its internal representations. We show that the network progressively encodes more and more temporal context by means of a probability analysis. Finally, we explore the conditions under which the network can carry information about distant sequential contingencies across intervening elements to distant elements. Such information is maintained with relative ease if it is relevant at each intermediate step; it tends to be lost when intervening elements do not depend on it. At first glance this may suggest that such networks are not relevant to natural language, in which dependencies may span indefinite distances. However, embeddings in natural language are not completely independent of earlier information. The final simulation shows that long distance sequential contingencies can be encoded by the network even if only subtle statistical properties of embedded strings depend on the early information. The network encodes long-distance dependencies by *shading* internal representations that are responsible for processing common embeddings in otherwise different sequences. This ability to represent simultaneously similarities and differences between several sequences relies on the graded nature of representations used by the network, which contrast with the finite states of traditional automata. For this reason, the network and other similar architectures may be called *Graded State Machines*.

Keywords. Graded state machines, finite state automata, recurrent networks, temporal contingencies, prediction task

1. Introduction

As language abundantly illustrates, the meaning of individual events in a stream—such as words in a sentence—is often determined by preceding events in the sequence, which provide a context. The word 'ball' is interpreted differently in "The countess threw the ball" and in "The pitcher threw the ball." Similarly, goal-directed behavior and planning are characterized by coordination of behaviors over long sequences of input-output pairings, again implying that goals and plans act as a context for the interpretation and generation of individual events.

The similarity-based style of processing in connectionist models provides natural primitives to implement the role of context in the selection of meaning and actions. However, most connectionist models of sequence processing present all cues of a sequence in parallel and

often assume a fixed length for the sequence (e.g., Cottrell, 1985; Fianty, 1985; Selman, 1985; Sejnowski & Rosenberg, 1987; Hanson and Kegl, 1987). Typically, these models use a pool of input units for the event present at time t , another pool for event $t + 1$, and so on, in what is often called a 'moving window' paradigm. As Elman (1990) points out, such implementations are not psychologically satisfying, and they are also computationally wasteful since some unused pools of units must be kept available for the rare occasions when the longest sequences are presented.

Some connectionist architectures have specifically addressed the problem of learning and representing the information contained in sequences in more elegant ways. Jordan (1986) described a network in which the output associated to each state was fed back and blended with the input representing the next state over a set of 'state units' (Figure 1).

After several steps of processing, the pattern present on the input units is characteristic of the particular sequence of states that the network has traversed. With sequences of increasing length, the network has more difficulty discriminating on the basis of the first cues presented, but the architecture does not rigidly constrain the length of input sequences. However, while such a network learns *how to use* the representation of successive states, it does not discover a representation for the sequence.

Elman (1990) has introduced an architecture—which we call a simple recurrent network (SRN)—that has the potential to master an infinite corpus of sequences with the limited means of a learning procedure that is *completely local in time* (Figure 2). In the SRN, the hidden unit layer is allowed to feed back on itself, so that the intermediate results of processing at time $t - 1$ can influence the intermediate results of processing at time t . In practice, the simple recurrent network is implemented by copying the pattern of activation on the hidden units onto a set of 'context units' which feed into the hidden layer along with the input units. These context units are comparable to Jordan's state units.

In Elman's simple recurrent networks, the set of context units provides the system with memory in the form of a trace of processing at the previous time slice. As Rumelhart, Hinton and Williams (1986) have pointed out, the pattern of activation on the hidden units

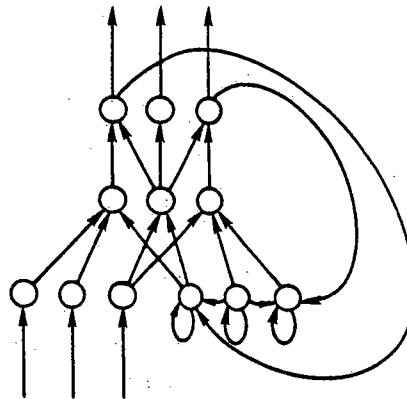


Figure 1. The Jordan (1986) Sequential Network.

