the input/output sequences span long intervals. We show why gradient based learning algorithms face an increasingly difficult problem as the duration of the dependencies to be captured increases. These results expose a trade-off between efficient learning by gradient descent and latching on information for long periods. Based on an understanding of this problem, alternatives to standard gradient descent are considered.

I. Introduction

WE ARE INTERESTED IN training recurrent neural networks to map input sequences to output sequences, for applications in sequence recognition, production, or time-series prediction. All of the above applications require a system that will store and update context information; *i.e.*, information computed from the past inputs and useful to produce desired outputs. Recurrent neural networks are well suited for those tasks because they have an internal state that can represent context information. The cycles in the graph of a recurrent network allow it to keep information about past inputs for an amount of time that is not fixed *a priori*, but rather depends on its weights and on the input data. In contrast, static networks (i.e., with no recurrent connection), even if they include delays (such as time delay neural networks [15]), have a finite impulse response and can't store a bit of information for an indefinite time. A recurrent network whose inputs are not fixed but rather constitute an input sequence can be used to transform an input sequence into an output sequence while taking into account contextual information in a flexible way. We restrict our attention here to discrete-time systems.

Learning algorithms used for recurrent networks are usually based on computing the gradient of a cost function with respect to the weights of the network [22], [21]. For example, the back-propagation through time algorithm [22] is a generalization of back-propagation for static networks in which one stores networks in which dynamic neurons—with a single feedback to themselves—have only incoming connections from the input layer. It is local in time like the forward propagation algorithms and it requires computation only proportional to the number of weights, like the back-propagation through time algorithm. Unfortunately, the networks it can deal with have limited storage capabilities for dealing with general sequences [7], thus limiting their representational power.

A task displays long-term dependencies *if* prediction of the desired output at time t depends on input presented at an earlier time $\tau \ll t$. Although recurrent networks can in many instances outperform static networks [4], they appear more difficult to train optimally. Earlier experiments indicated that their parameters settle in sub-optimal solutions that take into account short-term dependencies but not long-term dependencies [5]. Similar results were obtained by Mozer [19]. It was found that back-propagation was not sufficiently powerful to discover contingencies spanning long temporal intervals. In this paper, we present experimental and theoretical results in order to further the understanding of this problem.

For comparison and evaluation purposes, we now list three basic requirements for a parametric dynamical system that can learn to store relevant state information. We require the following:

1) That the system be able to store information for an arbitrary duration.
2) That the system be resistant to noise (i.e., fluctuations of the inputs that are random or irrelevant to predicting a correct output).
3) That the system parameters be trainable (in reasonable time).

Throughout this paper, the long-term storage of definite bits of information into the state variables of the dynamic system