Grammar Transfer in a Second Order
Recurrent Neural Network

Michiro Negishi
Department of Psychology
Rutgers University
101 Warren St. Smith Hall #301
Newark, NJ 07102
negishi@psychology.rutgers.edu

Stephen José Hanson
Psychology Department
Rutgers University
101 Warren St. Smith Hall #301
Newark, NJ 07102
jose@psychology.rutgers.edu

Abstract

It has been known that people, after being exposed to sentences generated by an artificial grammar, acquire implicit grammatical knowledge and are able to transfer the knowledge to inputs that are generated by a modified grammar. We show that a second order recurrent neural network is able to transfer grammatical knowledge from one language (generated by a Finite State Machine) to another language which differ both in vocabularies and syntax. Representation of the grammatical knowledge in the network is analyzed using linear discriminant analysis.

1 Introduction

In the field of artificial grammar learning, people are known to be able to transfer grammatical knowledge to a new language which consists of a new vocabulary [6]. Furthermore, this effect persists even when the new strings violate the syntactic rule slightly as long as they are similar to the old strings [1]. It has been shown in the past studies that recurrent neural networks also have the ability to generalize previously acquired knowledge to novel inputs. For instance, Dienes et al. ([2]) showed that a neural network can generalize abstract knowledge acquired in one domain to a new domain. They trained the network to predict the next input symbol in grammatical sequences in the first domain, and showed that the network was able to learn to predict grammatical sequences in the second domain more effectively than it would have learned them without the prior learning. During the training in the second domain, they had to freeze the weights of a part of the network to prevent catastrophic forgetting. They used this simulation paradigm to emulate and analyze domain transfer, effect of similarity between training and test sequences, and the effect of n-gram information in human data. Hanson et al. ([3]) also showed that a prior learning of a grammar facilitates the learning of a new grammar in the cases where either the syntax or the vocabulary was kept constant.

In this study we investigate grammar transfer by a neural network, where both syntax and vocabularies are different from the source grammar to the target grammar. Unlike Dienes et al.'s network, all weights in the network are allowed to change dur-
ing the learning of the target grammar, which allows us to investigate interference as well as transfer from the source grammar to the target grammar.

2 Simulation Design

2.1 The Grammar Transfer Task

In the following simulations, a neural network is trained with sentences that are generated by a Finite State Machine (FSM) and is tested whether the learning of sentences generated by another FSM is facilitated. Four pairs of FSMs used for the grammar transfer task are shown in Fig. 2. In each FSM diagram, symbols (e.g. A, B, C, ...) denote words, numbers represent states, a state number with an incoming arrow with no state numbers at the arrow foot (e.g. state 1 in the left FSM in Fig. 2A) signifies the initial state, and numbers in circles (e.g. state 3 in the left FSM in Fig. 2A) signify the accepting states. In each pair of diagrams, transfer was tested in both directions: from the left FSM to the right FSM, and to the opposite direction. Words in a sentence are generated by an FSM and presented to the network one word at a time. At each time, the next word is selected randomly from next possible words (or end of sentence where possible) at the current FSM state with the equal probability, and the FSM state is updated to the next state. The sentence length is limited to 20 words, excluding START.

The task for the network is to predict the correct termination of sentences. If the network is to predict that the sentence ends with the current input, the activity of the output node of the network has to be above a threshold value, otherwise the output has to be below another threshold value. Note that if a FSM is at an accepting state but can further transit to another state, the sentence may or may not end. Therefore, the prediction may succeed or fail. However, the network will eventually learn to yield higher values when the FSM is at an accepting state than when it is not. After the network learns each training sentence, it is tested with randomly generated 1000 sentences and the training session is completed only when the network makes correct end point judgments for all sentences. Then the network is trained with sentences generated by another FSM. The extent of transfer is measured by the reduction of the number of sentences required to train the network on an FSM after a prior learning of another FSM, compared to the number of sentences required to train the network on the current FSM from scratch.

2.2 The Network Architecture and the Learning Algorithm

The network is a second order recurrent neural network, with an added hidden layer that receives first order connections from the input layer (Fig. 1). The network has an input layer with seven nodes (A, B, C, ..., F, and START), an output layer with one node, an input hidden layer with four nodes, a state hidden layer with four nodes, and a feedback layer with four nodes. Recurrent neural networks are often used for modeling syntactic processing [3]. Second order networks are suited for processing languages generated by FSMs [4]. Learning is carried out by the weight update rule for recurrent networks developed by Williams and Zipser ([7]), extended to second order connections ([4]) where necessary. The learning rate and the momentum are 0.2 and 0.8, respectively. High and low thresholds are initialized to 0.20 and 0.17 respectively and are adapted after the network have processed the test sentences as follows. The high threshold is modified to the minimum value yielded for all end points in the test sentences minus a margin (0.01). The low threshold is modified to the high threshold minus another margin (0.02). These thresholds are used in the next training and test.