Rule Induction and Interference in the Absence of Feedback: A Classifier System Model

Lewis G. Roussel, Robert C. Mathews, and Barry B. Druhan

Louisiana State University

Department of Psychology

ABSTRACT

This study extends the work of Druhan et al. (1989) and Mathews et al. (1989b) by applying their computational model of implicit learning to the task of learning artificial grammars (AG) without feedback. The ability of two induction algorithms, the forgetting algorithm which learns by inducing new rules from presented exemplars and the genetic algorithm which heuristically explores the space of possible rules, to induce the grammar rules through experience with exemplars of the grammar is evaluated and compared with data collected from human subjects performing the same AG task. The computational model, based on Holland et al.'s (1986) induction theory represents knowledge about the grammar as a set of partially valid condition-action rules that compete for control of response selection. The induction algorithms induce new rules that enter into competition with existing rules. The strengths of rules are modified by internally generated feedback. Strength accrues to those rules that best represent the structure present in the presented exemplars. We hypothesized that the forgetting algorithm would successfully learn to discriminate valid from invalid exemplars when the set of exemplars was high in family resemblance. We also proposed that the genetic algorithm would perform better than chance but not as well as the forgetting algorithm. Results supported those hypotheses. Interestingly, the Mathews et al. (1989a) subjects performed no better than chance on the same AG learning task. We concluded that this discrepancy between the simulation results and the human data is caused by interference from unconstrained hypotheses generation of our human subjects. Support for this conclusion is two-fold: (1) subjects are able to learn the AG when the task is designed so that hypothesis generation is inhibited, and (2) informal inspection of verbal protocols from human subjects indicates they are generating and maintaining hypotheses of little or no validity.

INTRODUCTION

Some models have been proposed to account for learning in situations where external feedback is either missing or unreliable (eg., Fried & Holyoak, 1984; Billman & Heit, 1988). Generally, these models work by detecting environmental regularity in the form of either frequency cues or feature covariation. Models of implicit learning, which also operate by detecting and encoding structural features in the input, would seem to be perfectly capable of learning in such an environment. A model (THIYOS) based on a classifier system has been found useful for modeling implicit learning of artificial grammars when external feedback is present (Mathews, Druhan, & Roussel, 1989). We are encouraged by this result because the induction model of Holland, Holyoak, Nisbett, and Thagard (1986) from which THIYOS is derived is claimed to be relevant to a wide range of task domains, from conditioning to stereotyping. This paper will present evidence that THIYOS can learn in the absence of external feedback.

Recently Mathews et al. (1989b) applied three different induction algorithms to learning artificial grammars (AG) with THIYOS. The AG task consisted of a series of trials in which THIYOS attempted to select the valid string from a list of five alternatives and was given feedback about the correct choice after each trial. On each trial four of the strings contained from one to four violations (incorrect letters). The simulation data suggested that the effectiveness of the algorithm depends on how data driven the task is. One of the algorithms, termed "forgetting", was shown to be superior for learning an AG in this multiple choice task because it best exploits clues to the underlying rule structure provided by new exemplars across trials.

Although the Mathews et al. (1989b) data were collected under conditions of continues feedback about correctness of choices, we hypothesized that the forgetting algorithm would operate well in such a data driven task even without feedback. Given that all of the choices contain grammatical features (e.g., valid bigrams or trigrams), even when wrong choices are selected valid rules could be generated by the forgetting algorithm. Further, valid rules should win out across trials over less valid rules because valid features will apply more often in the entire population of choices. To test this hypothesis, THIYOS was modified to operate without feedback and then applied to the task of learning the same two artificial grammars studied previously by Mathews et al. (1989b). After a brief description of the learning task and THIYOS, the experiment and results will be discussed along with some comparisons of simulation data with data collected from human subjects who also tried to learn the grammars without feedback.

THE LEARNING TASK

On each trial a set of five letter-strings were presented. The strings on each trial consisted of one completely valid string, one string with one letter that could not occur in a particular position, one string with two wrong letters, one with three, and one with four invalid letters. The object was to select which of the five choices was the correct string. No feedback about the correctness of the choice was given. This procedure continued for 200 trials and was repeated three times with three sets of stimulus items. Data from human subjects learning the grammar under identical conditions were collected in a previous study (Mathews, Buss, Stanley, Blanchard-Fields, Cho, and Druhan, 1989). Subjects in that study participated in three sessions distributed over a three week period. Each session consisted of 200 trials. The stimulus items used in the simulations were identical to those in the Mathews, et al. study. The initial block of ten trials each week contained all new items and then each successive block of trials contained five old items repeated from previous trials that week and five new items. In sessions 2 and 3, most items (old or new) occurred in previous sessions.

Both grammars used in the Mathews, et al. (1989a) study were used in this study. The finite state grammar is illustrated in figure 1a. A valid string is any sequence of letters generated by following a pattern of arrows from left to right. Strings generated by this grammar tend to share many features, that is they have a high level of family resemblance among exemplars. The biconditional grammar is illustrated in figure 1b. It contains three letter association rules that determine what pairs of letters must occur in corresponding positions in the first and second halves of the string. As illustrated in figure 1b, the three association rules are S goes with V, C goes with P, and T goes with X. The first four letters in a valid string can be any combination of the six letters, but once they are selected, they completely determine what the second set of four letters must be (see figure 1b). Exemplars of the biconditional grammar do not share any common sequences of initial or final letter patterns and, therefore, they have a lower level of family resemblance among exemplars.

THE MODEL

The computational model described here is essentially a classifier system model with certain modifications that make it amenable to modeling artificial grammars in general and the particular task specifically (see Holland et al., 1986). Classifier systems are a

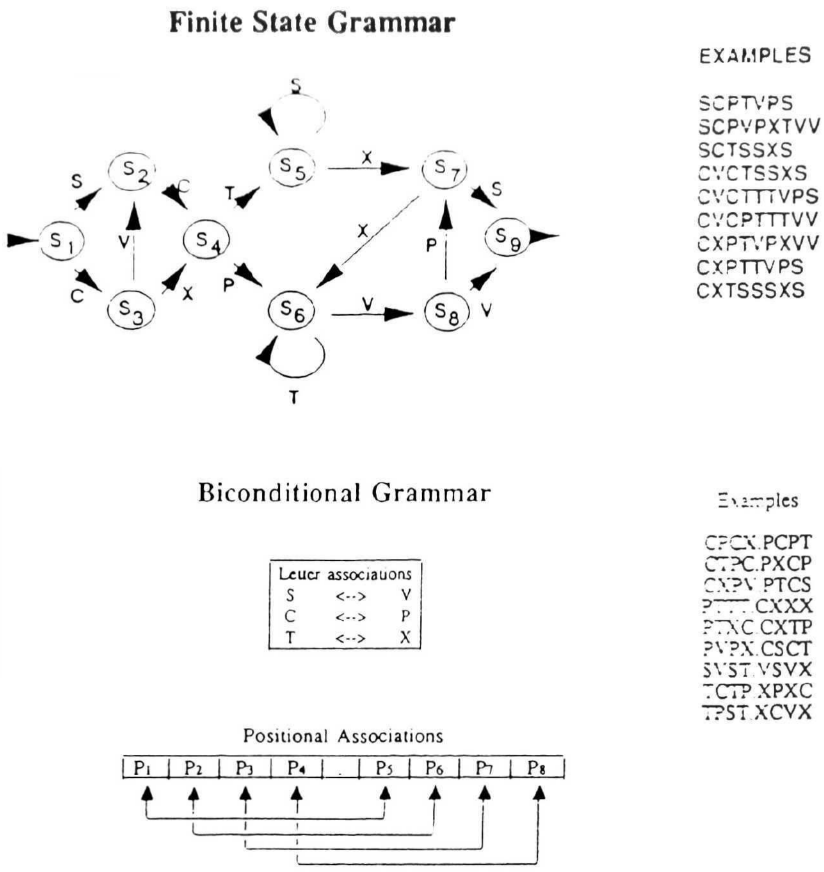

Figure 1a and 1b.

type of production system model with some specific processing assumptions. First, the productions, condition-action pairs, are composed of strings of equal length. Each condition-action pair is called simply a "classifier". Each element can be thought of as representing a unique feature of the object or event being described by the classifier. Complex objects or events can be coded by adding conditions to the condition side of the classifier--each classifier has only one action.

Rules are represented as condition-action pairs, in which both the conditions and the actions are fixed length strings of letters, numbers, "#"s, and "_"s. The exemplars of the grammar to be learned are represented in a similar fashion such that the lengths of the string, action string, and the exemplar string are all equal. In order to determine whether a rule applies, its condition side is matched position by position against the exemplar string. The "#"s are a sort of wildcard character that will match anything. In addition, the "#"s act as variables in that they can pass information through from a message to an action. Consider the following example: the rule "if the string begins with SCT then choose it" would be represented in THIYOS as:

"##SCT########0###00 I 02CHOOSE_____######".
condition action

The five alternative strings are placed on the message list in a similar format. For example, the above rule would match an exemplar on the message list such as: "01SCTVPXVV___#10###". Numbers at the beginning of the strings are tags which differentiate strings coming from the input interface from those going to the output interface. In the exemplar string, the "1" and "0" in the 14th and 15th positions indicate that it is choice number 1 for the given trial, and that it has zero violations. Since the corresponding positions in the condition and action side of the classifier contain "#"'s, the "1" and "0" are passed through from the exemplar to the action. Since the action of this classifier is tagged for the output interface, it would tell the system to choose letter string number 1. The execution cycle performs one trial per cycle by iterating through the following steps:

1) Read in the five alternative exemplars from the input interface, and place them on the message list.

2) Compare the condition sides of all rules to each message on the message list and record all matches.

3) Calculate a bid for each classifier on the interim list using the parameters of strength, specificity, and support.

4) Select the w highest bidders and allow these classifiers to post their messages on an interim message list. (The size of the set (w) reflects the model's assumptions about working memory limitations.)

5) Recalculate the bids for all classifiers on the interim list and post their messages on a new message list. This rebidding is necessary because some classifiers lose the support of classifiers that failed to make it to the interim list.

6) All rules on the new message list loose a portion of their strength as payout for the privilege of posting their messages.

7) Process the contents of the new message list through an output interface which strips off messages tagged for output. The highest bidder and all rules whose messages agree with the highest bidder are rewarded by incrementing their strength.

8) Depending on which learning algorithm is being applied, a new rule(s) is created by that algorithm and added to the system.

9) Replace the old message list with the new one.

10) Reduce the strength of all rules in the system. This step simulates the forgetting of rules which seldom or never compete and apply.

11) Return to step 1.

This process continues until all trials have been completed.

Within the Induction framework of Holland, et al. (1986) an algorithm termed the "bucket brigade" is used to allocate activation or strength to good rules in the system across different time steps based on strength, specificity, and support. A similar mechanism is used here with some necessary modifications. In the current model each trial is processed in discrete fashion. No chaining between rules takes place (e.g., see Holland et al., 1986). Therefore the calculation of support was modified to represent the level of agreement among rules on a single time-step rather than the relevance of a rule to the context of the previous time-step (see Druhan & Mathews, 1989, for a more detailed discussion). Strength is a numerical measure of the level of success a rule has had in representing the environment. Specificity is a measure of how complete that representation is (Holland et al., 1986). The learning algorithms

We examined two algorithms for creating rules. The first, the genetic algorithm, is inspired from genetics and involves mutation and crossover of different parts of existing rules. The second, the forgetting algorithm, makes new rules from presented exemplars by retaining a subset of features of that item and creating a new rule that says to select strings with those features.

The genetic algorithm. The genetic algorithm uses two methods or "operators" for altering existing rules to make new rules. The first, crossover, splits pairs of rules into two parts and recombines the parts to make new rules. Candidates for crossover are selected from those rules posting messages on the current trial. These candidates are copies, ranked by strength and paired off so that the strongest two go together, and the next stronges pair, etc.. Then a random split point is selected in each pair of classifiers and the copies are split and recombined by exchanging their initial and final portions. Thus two new rules are created from each pair of rules and the original or parent rules are left intact in the set of rules.

The second genetic operator, mutation, creates a copy of a randomly selected original rule and randomly changes one character, making a new rule. The mutation rate is set very low in this simulation; each existing rule in the rule set mutates with a probability of 0.001. The purpose of mutation is to keep some "new blood" in the rule system. Without mutation, crossover might tend to inbreed certain types of rules and effectively lock out discovery of radical different rules.

The genetic algorithm is best characterized as top-down or conceptually driven. Its search through the space of possible rules is guided mainly by the knowledge contained in its existing rule set. Input serves only to qualify the goodness of that knowledge by strengthening those rules created by the algorithm that have posted their messages. Therefore we hypothesized that, in the absence of external feedback, the genetic algorithm would eventually succeed in inducing a representation of the grammar but would proceed slowly.

The forgetting algorithm. The forgetting algorithm extracts features from the choice selected by the simulation on the current trial and makes it a new rule. In this algorithm features of the choice are included in the new rule probabilistically. A serial position curve is used to set the probabilities of incorporating features into the new rule.

One interesting aspect of the forgetting algorithm concerns the tradeoff between remembering too much versus too little of an exemplar. If all the features of the exemplar were incorporated into the new rule it could only be applied to that one exemplar. On the other hand remembering too little about a past exemplar and creating a new rule based on only one feature of the exemplar would tend to produce rules that have little or no validity. Thus, it is better to forget part of the exemplars one sees in creating new rules--hence the name forgetting algorithm.

The forgetting algorithm is data driven in that it induces rules based solely on the input. Because all items to choose from are either valid or distortions of valid items, all share features in common with valid strings across trials and, consequently, whatever structure that exists in the input will be captured in the set of induced rules. This suggests that "forgetting" should perform well in the present task even though external feedback is absent.

THE SIMULATION

The two algorithms were applied to the two grammars, finite state and biconditional. The genetic algorithm needs rules to "mate" and produce "offspring" so a set of initial rules was constructed and entered into THIYOS at the start of each run. This initial set of rules represents the basic knowledge that any subject would bring to the task. This knowledge consists of the set of all possible rules for string selection based on single letters in a specific position (e.g., "choose strings that have an "S" in position 3"). Runs of the forgetting algorithm were initiated with the same set of initial rules.

RESULTS

The dependent variable is the number of errors (incorrect strings) selected in trials 11 through 200 of each session (trials 1 through 10 were not included in the analyses because the distribution of old versus new items in those trials differed from all other trials). Since each trial consisted of five choices, there is a 0.80 probability of committing an error by chance. Therefore, chance performance is 152 errors per session.

The mean performance for each condition is plotted in figure 2. In an analysis of variance (ANOVA) of the error data the overall group (algorithm) effect was significant for both grammars with $F(1,18)=56.26$, $p<0.0001$ and $F(1,18)=11.23$, $p<0.0036$ for the finite state grammar and the biconditional respectively. For the finite state grammar the forgetting algorithm performed better than genetic. Both were significantly better than chance by session three. For the biconditional grammar, the same pattern, forgetting better than genetic, was obtained. Again, both were significantly better than chance by

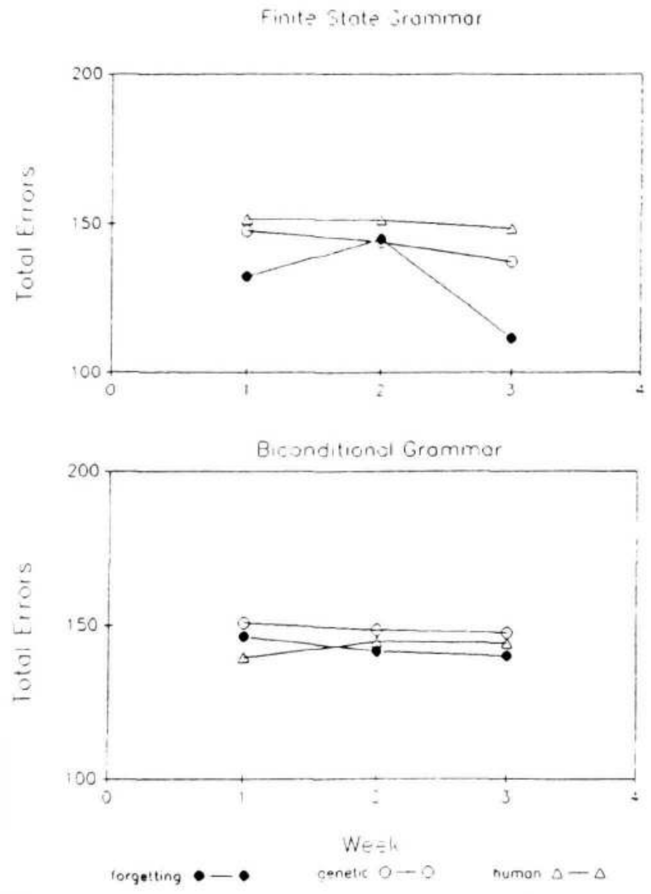

Figure 2.

session three.

Data from Mathews, et al. (1989a, experiment 1) were included in a second analysis to determine how well the simulation fit the human data. Subjects had been trained with the same stimulus items used in the simulation. These subjects had performed the same multiple choice task under no-feedback conditions. The human subjects performed at chance for all three sessions and for both grammars (see figure 2).

DISCUSSION

Two general conclusions can be drawn. First, THIYOS combined with either of the two learning algorithms studied was able to perform better than chance in the grammar learning task even though the simulation received no feedback. This supports our hypothesis that the forgetting algorithm should be able to operate in the absence of feedback. The genetic algorithm was also successful but always performed less well than forgetting.

The second conclusion to be drawn concerns the performance of our human subjects: they were unable to learn about these artificial grammars when feedback was absent. This is a puzzle about which one can intelligently speculate in the light of the results from other artificial grammar (AG) learning experiments.

A tentative explanation for this discrepancy between the simulation results and data from human subjects would be that the performance of human subjects is hampered because their explicit, conscious strategies interfere with implicit learning. That interference is responsible for the poor performance of the subjects, is supported by several studies. Reber and his colleagues (e.g., Reber, 1976; Reber, Kassin, Cantor, & Lewis, 1980) demonstrated that subjects performed poorly in an AG learning task when given instructions to search for rules. He attributed this interference to explicit processing mechanisms. In the Reber et al. (1980) study, presenting the stimuli in a structured display that revealed the family resemblance of the items resulted in better performance with explicit processing whereas presenting the items in a random fashion (unstructured condition) resulted in better performance with implicit processing. The present multiple choice task is similar to the unstructured condition because the five choices on a particular trial are selected randomly and share little family resemblance. Therefore it is reasonable to expect that explicit processes could interfere with learning in the present task.

On going work in our laboratory using a different paradigm for studying implicit learning indicates that subjects can acquire knowledge about the Mathews et al. finite state grammar without feedback during training. The paradigm is based on the Hebb effect. Subjects were presented letter strings, one per trial, and asked to hold each in memory while performing a distractor task. After the distractor task they attempted to recall as many letters of the string as possible. The subjects were not told that every third string was a valid string while all others were random strings. No feedback was given. Results indicate that subjects improved in recall of valid strings relative to recall of random strings across trials (showing a Hebb effect on valid strings) and they subsequently performed above chance on a string discrimination task. Thus humans can learn the finite state grammar without feedback when they are not explicitly forming hypotheses during learning.

Finally, when feedback is absent, there is little to guide the hypothesis generating processes toward solution. More importantly, without feedback, the hypothesis generating mechanism is virtually unrestrained. Completely wrong hypotheses can be generated and strengthened across trials. Verbal protocols of our human subjects taken after the experiment support this conclusion. Typically, subjects generated invalid rules such as "pick strings that look like words" or pick strings that have letters lower in the alphabet". We are currently testing the notion that such invalid rules can block the implicit learning mechanism by introducting one or more of their bad rules in a THIYOS simulation. We predict that their rules may gain strength and prevent the generation of valid rules.

If such simple mechanisms as the forgetting algorithm are capable of exploiting structured stimulus domains without error correcting feedback, we should expect that living systems would have evolved similar capacities. The fact that the process was completely stopped by unrestrained hypothesis testing in our human subjects is quite interesting and it adds further impetus to the postulation of two distinct learning mechanisms.

REFERENCES

Billman, D, & Heitman, E. (1988). Observational learning from internal feedback: A simulation of an adaptive learning method. *Cognitive Science, 12*, 587-625.

Druhan, B. B., & Mathews, R. C. (1989). THIYOS: A classifier system model of implicit knowledge of artificial grammars. *Paper presented at the annual meeting of the Cognitive Science Society, Ann Arbor.*

Fried, L. S., & Holyoak, K. J. (1984). Induction of category distributions: A framework for classification learning. *Journal of Experimental Psychology: Learning, Memory, and Cognition, 10*, 234-257.

Holland, J. H., Holyoak, K. J., Nisbett, R. E., & Thagard, P. R. (1986). *Induction: Processes of inference, learning and discovery.* Cambridge, MA: The MIT Press.

Mathews, R. C., Buss, R. R., Stanley, W. B., Blanchard-Fields, F., Cho, J. R., & Druhan, B. (1989). Role of implicit and explicit processes in learning from examples: A synergistic effect. *Journal of Experimental Psychology: Learning, Memory, and Cognition, 15*, 1083-1100.

Mathews, R. C., Druhan, B. B., & Roussel, L. G. (1989). Forgetting is learning: Evaluation of three induction algorithms for learning artificial grammars. *Paper presented at the annual meeting of The Psychonomic Society, Atlanta.*

Reber, A. S. (1976). Implicit learning of synthetic languages: The role of instructional set. *Journal of Experimental Psychology: Learning, Memory, and Cognition, 2*, 88-94.

Reber, A. S., Kassin, S. M., Lewis, S., & Cantor, G. (1980). On the relationship between implicit and explicit modes in the learning of a complex rule structure. *Journal of Experimental Psychology: Human Learning and Memory, 6*, 492-502.

ACKNOWLEDGEMENTS

This research was supported in part by National Science Foundation Grant #BNS-8509493 to Robert C. Mathews, Ray R. Buss, and William B. Stanley, and in part by the Louisiana Board of Regents Grant #86-LBR(21)- 021-10 through the Louisiana Quality Education Support Fund to Robert C. Mathews.