[Category: Genetic Programming] Genetic Programming for Grammar Induction

Emin Erkan Korkmaz
Department of Computer Engineering
Middle East Technical University
Ankara-Turkey
korkmaz@ceng.metu.edu.tr

Göktürk Üçoluk
Department of Computer Engineering
Middle East Technical University
Ankara-Turkey
ucoluk@ceng.metu.edu.tr

Abstract

There has been a big interest in inducing classes of grammars in the area of machine learning. Various attempts have been carried out for automatically inferring different grammar classes. The symbolic nature of the grammar induction problem makes it suitable for the GP-approach. However the straightforward application of the GP method on **Context Free Grammar Induction** problem fails to generate a satisfactory solution. The interdependency among subparts of a CFG is high and it seems to be the reason that prevents the GP method from finding out effective building blocks during the search. In this paper a new approach is presented where the aim is to formalize a control module for the genetic search which can use the interdependency information existing in CFGs and hence can direct the search only among well-fit grammars in the search space.

1 Introduction

GP is an induction method used to search over a huge state space consisting of structured representations that are trees. Therefore GP is appropriate for solving symbolic tasks like finding out a function or composition of functions for a generalization over an example set.

Induction of context-free grammars can be visualized as a symbolic task too. The left-hand side of a rewrite rule in a grammar can be treated as a function which is composed of the right-hand side elements of the same rule. Thus, it is possible to represent context-free grammars as structured trees and transfer the problem of inducing a context-free grammar to a search problem among the possible tree representations that can be formed using the terminal and nonterminal symbols of a context-free grammar. When the GP method is used in a straightforward manner for the CFG-induction problem, although it is possible to achieve a certain degree of convergence, the output grammar induced is far away from a reasonable abstraction over the training sentences.

It can be claimed that the limitations for the convergence appear to be due to the nature of the problem at hand. The interdependency among subparts of a context-free grammar is high. That is, the contribution of a part of a grammar to the fitness function heavily depends on some other subparts. Hence the risk of destroying the interdependent overall structure and as a result a dramatic fall of the fitness value after a genetic operation is quite high.

On the other hand it can be claimed that the above problem is based on the limitations of the tree representation used for the genetic search. The tree abstractions are capable of representing subparts of a problem and how these subparts are connected to each other. However the representations used, fail to capture the global information based on the interdependency relation of the subparts of a grammar, hence it becomes possible to destroy this interdependent global structure throughout a genetic operation which results those dramatic changes in the fitness values and disturbs the convergence. In the following sections, a new approach which can formalize and process this global information related to the overall structure of a grammar, is presented. This new attempt tries to concentrate the search around the localization of well-fit elements of the search space by means of a classification algorithm and tries to increase the efficiency by assigning higher chances to creation of phenotypes that are in the vicinity of those localizations.

In the following section a short summary of the previous attempts on grammar induction problem is presented. In section 3 the straightforward application of GP method together with the initial results is described. Then in section 4 an analysis of the underlying facts that prevent the GP-method from a satisfactory convergence is provided. In section 5 the solution is proposed; in section 6 the results obtained are presented. The last sections conclude about the obtained results.

2 Previous Work

Although statistical approaches form the main stream among the researchers working on automatic grammar induction, several attempts have been carried out to attack the problem with evolutionary techniques also.

For instance in [8] a method using genetic algorithms for context-free grammar induction has been proposed. The authors claim that although statistical methods offer a possible solution to the problem, it is quite difficult to escape from problems like the “zero-frequency” problem using this approach. Hence, they propose an evolutionary approach where each chromosome in the population represents a context-free grammar. The fitness is measured by the ability of this grammar to parse a set of sample strings. The crossover operation used for the process combines subsets of the rules of two different grammars, so the operation does not create new rules in the population, but just makes new combinations of the existing rules appearing in different chromosomes in the population. On the other hand mutation is allowed to modify any symbol in the grammar and this is the operation used to introduce new rules that may have not appeared in earlier generations. The proposed approach is able to infer simple natural language grammars over a small set of training examples. The example grammar given as an output of the experiments carried out, covers noun phrases containing a single determiner and a set of nouns and the verb phrases containing a single verb and the noun phrase recursively.

The work proposed by [7] is very similar to [8]. However it is stressed that straight forward application of genetic algorithms are not very effective in grammar induction. Based on this fact this work proposes modifications to increase the success of the method are proposed. An example among the proposed modifications is to enlarge the mutation operation where the operation mixes the sub-components of a grammar as well as certain symbols of the grammar are modified by the operation.

3 Genetic Programming for Grammar Induction

As stated in section 1, it is possible to represent a CFG as a structured tree. This representation makes it possible to perform a genetic search among the candidate grammars in the form of structured trees. Note that, this representation allows the search to proceed by modifying the rules of a grammar both during mutation and crossover.

3.1 Fitness Function

The initial fitness function that has been tried for the search process was simply checking the number of training sentences that the grammar can parse successfully. Obviously this initial try for the fitness function was away from success. As the majority of the grammars that appeared during the search were not able to parse even a single sentence, the population was filled with invalid elements during the search process. Hence the convergence was disturbed.

The idea used to overcome the problem has been assigning a fitness value to a grammar even if the grammar is not able to parse even a single training sentence successfully.

In order to determine the fitness of such a ill-formed grammar, a parse tree is formed while trying to parse the training sentence with the parser formed by the candidate grammar. If the sentence cannot be parsed, it is not possible to obtain a complete parse tree. However, even if the parsing process stucks at some point , it is possible to obtain an incomplete parse tree of the sentence. This parse tree would denote the structure that has been formed up to the point where the parsing was not able to proceed further. Depending on the structure of the grammar, more than one such incomplete parse trees can be formed for a single sentence. So these incomplete parse trees that appear during the attempt to parse a sentence are analyzed and the maximum number of words, namely, terminal symbols that appear on the parse trees throughout the parsing process, is determined. This maximum number denotes the success of the grammar in parsing that sentence. If this maximum is equal to the length of the sentence, it would mean that the sentence has been parsed successfully. However even if the sentence cannot be parsed totally, this maximum number would be a measure of how close the grammar was to parse the sentence successfully.

Using the idea presented above, the fitness function is set as:

$$F(G) = \sum_{i=1}^{\#ofsentences} (LENGTH(S_i)/MAXPARSE(G, S_i)) - 1 \quad (1)$$

The genetic programming search tries to minimize the fitness value above for the context free grammars. In equation 1, $MAXPARSE(G, S_i)$ denotes the maximum number of terminal elements that appear on the parse trees during the attempt to parse the sentence S_i with grammar G. When the sentence is parsed successfully $MAXPARSE(G, S_i)$ equals the sentence length, hence $F(G)$ is zero.

3.2 The initial results

The representation described in the previous section has been tried on a group of training examples formed by considering a subpart of the English language. The training set consisted of about 20 simple English sentences consisting of an NP and a VP structure where the verb existing in the VP can be either transitive or intransitive. Considering the simplicity of the elements of the training set it can be claimed that the aimed output grammar would be a *toy grammar* on the English language. Though the training set was based on only a few simple structures that can exist in English language, the abstraction obtained was not satisfactory.

With the use of the new fitness function presented in the previous section, a certain degree of convergence has been obtained. For instance consider the following fragment of a grammar induced where X_2 and X_3 are non-terminal symbols and N and D are terminals symbols corresponding to nouns and determiners:

- $X_2 \rightarrow X_3N$
- $X_3 \rightarrow X_2$
- $X_3 \rightarrow D$

This fragment induced, can be considered as a correct abstraction over the NP clauses existing in the elements in the training set, since the NPs in the training set only consisted of a certain number of Ns preceded by a determiner. If we consider the fragment induced, using the recursion on X_2 any number of Ns can be produced and this sequence has to be preceded by a determiner.

However, when the overall success of the search process is considered, it is observed that the induced grammars usually have irrelevant rules and they over-generate on the training set. It can be claimed that the straightforward application of the GP method on CFG induction problem fails to produce a reasonable abstraction and the attempts to improve the fitness function does not seem to be helpful beyond a small degree.

4 The Nature of the Problems in Grammar Induction

The interdependency among the subparts of grammar can be considered as the major factor preventing convergence. Attempts to formalize more efficient fitness functions are not expected to be helpful, since a lot of the problem seems to be related to the basic aspects of the genetic approach. This observation leads to the idea that the performance of the approach can only be increased by an outside guidance. In this research this guidance is considered to be a *separate control module* that can direct the GP search.

The problem with applications with high interdependency is that, it becomes so probable for the population to be captured with the invalid elements during the search. The reason causing this unwarranted situation is related to the basic assumption underlying the GP method. GP approach assumes that combining efficient building blocks would lead to better solutions. However forming and processing such building blocks becomes impossible for problems with high interdependency. After all, genetic search is also based on the idea of stepping from one structure to another that is similar to the previous one, by modifying a subpart of the structure throughout a genetic operation. The implicit assumption of this approach is that similar structures would have compatible fitness values and it would be possible to proceed towards better solutions throughout the search. However with the tree representation it becomes possible to destroy a lot with even a minor modification on the chromosome. Therefore very similar structures in the domain might have totally different fitness values.

This is the case for the grammar induction problem too. It is possible to analyze the case using an example: Consider the grammar given below with a terminal element set consisting of N and V and a non-terminal element set consisting of S, NP and VP. This grammar would cover recursive NP structures and simple transitive VP structures consisting of a verb and an NP in English language.

- $S- > NP, VP$
- $NP- > N, NP$
- $NP- > N$
- $VP- > V, NP$

This candidate grammar would be a valid element in our domain set and it is expected to have a high fitness value since it can parse quite a number of sentences. However the interdependency between the subparts makes it too risky to end up with an invalid grammar if we apply a genetic operation on one of these elements. For instance the terminal element "V" is one of the critical subparts in the grammar. The existence of this element is a factor that contributes highly to the fitness of the element.

For instance, if we replace the terminal element "V" with the terminal element "N" the grammar cannot parse any sentence at all, since any sentence should have a verb in it. The two grammars seem to be very similar and for the genetic search to proceed successfully the fitness values in such cases should be compatible with each other. The absence of such a regularity makes it impossible to determine if we are going to be falling off the edge or if we are stepping up to another confident structure when we perform a genetic operation.

The fitness function presented in section 3.1 can be considered as an attempt to overcome the problem, however this attempt can be considered only as a partial solution. The fitness function used enables the search to keep track of the building blocks more efficiently, since it is possible to assign fitness values to incomplete parse attempts. On the other side, it is still probable to destroy the overall structure of a chromosome throughout a genetic operation, therefore still the problem of having dramatic changes in the fitness value exists with such an improved fitness function too.

5 Solution Proposed:

5.1 Search for an answer in Cognitive Science

The organization of the conceptual system is an ongoing debate in the area of cognitive science. The classical theory on category formation states that things are placed into same categories on the basis of what they have in common and concepts are *atomistic*, that is they can be broken down into smaller building blocks. However the classical view is not shared by all of the cognitive scientists and there are researchers claiming that the classical theory is capable of explaining only a small part of the whole story and concept formation is based on more complex processes rather than simple building blocks. The new approach is called *prototype theory* and visualizes concepts as atomic structures. The new approach focuses on the overall structures of the concepts that goes beyond putting together building blocks [2].

It can be claimed that this theoretical debate taking place in Cognitive science could have certain impacts on the formalization of the control module needed to direct the genetic search. A parallelism can be formed between the debate taking place in Cognitive science and the limitations of the genetic approach. The tree representation used for chromosomes heavily depends on the assumption that a solution which can be considered as a concept, is formed by bringing in different building blocks. The tree representation holds only these various building blocks and how they are connected to each other. And similar to the discussions taking place in cognitive science, the representation used can help to solve certain amount of problems, however this seems to be only a small part of the whole story and whenever the interdependency among the subparts of a problem increases, that is to say the total meaning goes beyond putting the building blocks together, the genetic approach fails together with the classical theory of concept formation.

5.2 What could be the solution?

Considering the limitations of the GP formalism, it can be claimed that a control module is needed to guide the genetic search which should be able to form and analyze the global information due to the interdependency among the subparts of the problem.

In order to formalize the control module, again the debate going on in cognitive science could be helpful. The new approach in cognitive science considers *atomic prototypes of concepts* to be able to explain the organization of the conceptual system. Being a member of a concept or category is defined in terms of the *distance* to these prototypes. Using the same approach, if prototypes for the valid elements in the domain can be built based on the interdependency information existing in the chromosomes, then it might be possible to direct the search only around well-formed chromosomes.

The key point for this re-representation of the chromosomes is related in fact to the redefinition of the *similarity* between the chromosomes. The aim is obviously to visualize the domain from a different angle so that the chromosomes with compatible fitness values would be showing up in places close to each other.

5.3 Vectorial Representation

The solution we propose is to transform the chromosomes to single points of an n-dimensional space which are then subject to the control module. It is aimed that the control module would use these atomic representations and would try to determine prototypes for the valid and invalid elements in the domain. Then, these prototypes can be fed in the genetic search and the genetic search can use them to determine the consequences of a genetic operation *beforehand* and perform the right genetic operations that would keep the search only among the sensible elements in the search space.

Here it should be noted that this transformation is not aimed to be a one-to-one correspondence between the chromosomes and points in the n-dimensional space. Rather it is aimed to capture the overall information that exists implicitly in the chromosome structure.

The transformation process for a chromosome can be divided into two phases. First, the terminal elements used for the genetic search have to be transformed into the new space. In order to perform this transformation base vectors have been used for each terminal set, since the elements in the terminal set can be considered as entities having the same characteristics.

The dimension of the space has been determined as the number of elements in the terminal set. For instance if the terminal is $T = \{D, N, P, V\}$ then the base vectors corresponding to the elements would be:

$D = [0, 0, 0, 1]$

$N = [0, 0, 1, 0]$

$P = [0, 1, 0, 0]$

$V = [1, 0, 0, 0]$

The usage of these base vectors for the terminal set elements allows to place the set members into the space with the same distances among each other.

On the other hand, in order to form the vectors corresponding to a nonterminal element there are various methods that can be used. The most trivial one would be adding the vectors of the arguments of this nonterminal element. For instance if $X(X_1, X_2)$ is a subpart of the tree, then the vector that would correspond to X might be determined as:

$$V_X = V_{X_1} + V_{X_2} \tag{2}$$

By using such a bottom up construction it is possible to determine a single vector for the whole chromosome. Note that it is possible to have more than one chromosomes mapped to the same point in the space since the vector addition is commutative and associative. However this is not in contradiction with our purpose, since it is aimed to extract information regarding to the overall structure of a chromosome with this mapping. For instance with this addition operation it is possible to gather the information about how many of which terminal elements are used in a chromosome.

5.4 The interaction between the Control Module and GP

The general flow of the interaction between the two modules is formalized as follows:

- Genetic Search : start the search and for each chromosome that appears in a population, form the corresponding vector representation and send the vector together with the fitness value to the control module.
- Control Module: after collecting a certain amount of vector values and corresponding fitnesses, run a classification algorithm on the data and try to form prototypes for the valid and invalid chromosomes based on the fitness values.
- Genetic Search: after the prototypes are formed by the control module, for each genetic operation to be performed, send more than one alternative to the control module. (For instance for the crossover operation different alternatives would mean different crossover points on the parents.) Then, receive from the control module the distances between the offsprings that would be produced by the alternative operations and the prototypes induced and then choose the alternative with offsprings closest to the valid prototypes in the n-dimensional space.

6 Implementation & Results

In the implementation, the control module used collects vector and fitness values for a period of thirty generations and after each thirty generations, the control module uses the *"C4.5 Decision Tree Induction System"* for building the prototypes. It is aimed to induce two different kinds of prototypes. The first one is assumed to be a prototype for chromosomes with low fitness values and the second one for chromosomes with average or high fitness values. Therefore the positive examples for the induction system are set as vectors with a corresponding average or high fitness value and the negative examples are determined as vectors with fitness values below the average.

The control module produces the first prototypes at generation thirty. Then, before each genetic operation is carried out, the genetic engine sends the alternative operations and gets the feedback from the control module about the the offsprings that would be produced. The alternative that would reproduce offsprings in the positive class is chosen.

Different trials have been carried out with varying random seeds in order to visualize the behavior of the genetic search with and without the control module. During the runs with the control module, for each genetic operation two different alternatives are sent to the control module and the better one is chosen according to the feedback obtained from the control module.

In order to be able to compare the results obtained with and without the control module the genetic parameters are kept constant through all the different trials. Below is the list of genetic parameters used in all trials.

- Population size = 100
- max depth for new trees = 3
- max depth after crossover = 4
- max mutant depth = 2
- crossover at function point fraction = 0.1
- crossover at any point fraction = 0.7
- Reproduction fraction = 0.1
- Mutation fraction = 0.1

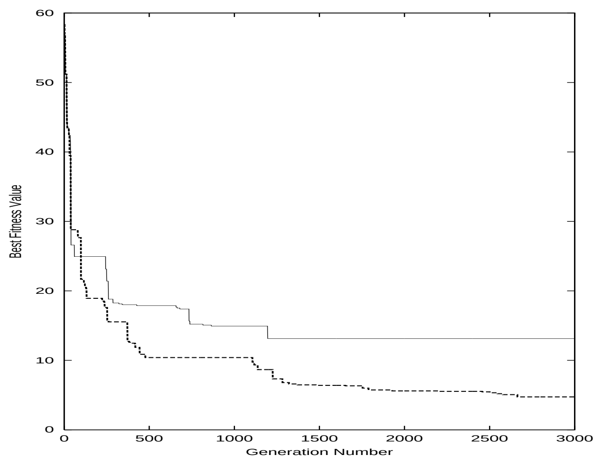

Figure 1: Comparison of controlled search and normal run.

In figures 1, 2, 3, 4 and 5, five different comparisons are presented between the normal run and the controlled search. The change of the fitness value for both the normal run and the controlled search are presented together. Here the dashed lines in each graph denote the change of the fitness function during the controlled search. For each comparison a different random seed

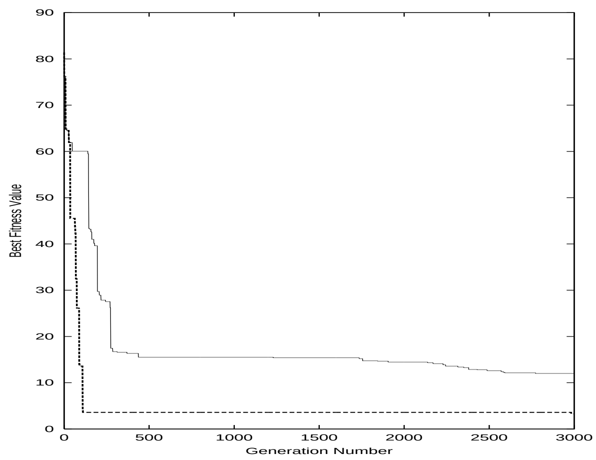

Figure 2: Comparison of controlled search and normal run.

is used in order to determine if the progress obtained is within the borders of normal variation of the genetic search or not.

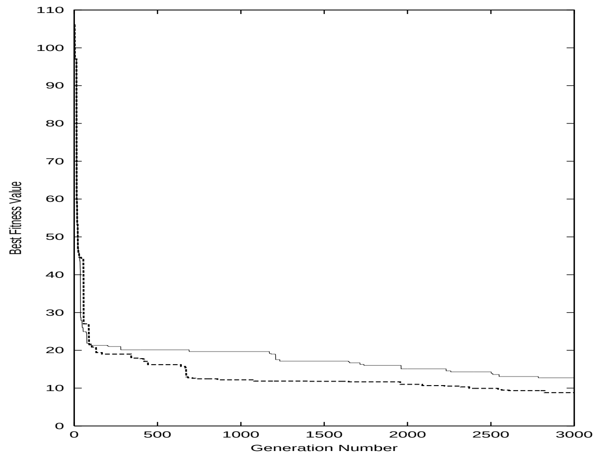

Figure 3: Comparison of controlled search and normal run.

The controlled search has a better performance compared to the straightforward application of GP on the CFG-induction problem. In the first three figures the controlled search clearly outperforms the normal run. However it is not possible to claim that the controlled search is always able to escape from the local minimas that the normal run is suffering from. The performance in figures 4 and 5 denotes that the normal run can follow the controlled search with some phase difference in some trials. In these figures the normal run can reach the performance of the controlled search after a certain amount of generations and then the two methods seem to be stuck at the same local minima.

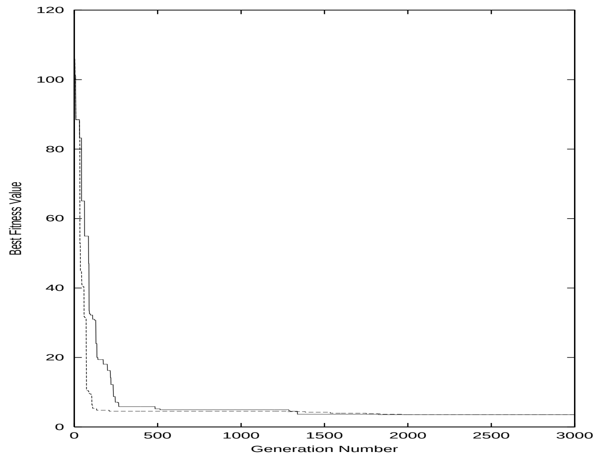

Figure 4: Comparison of controlled search and normal run.

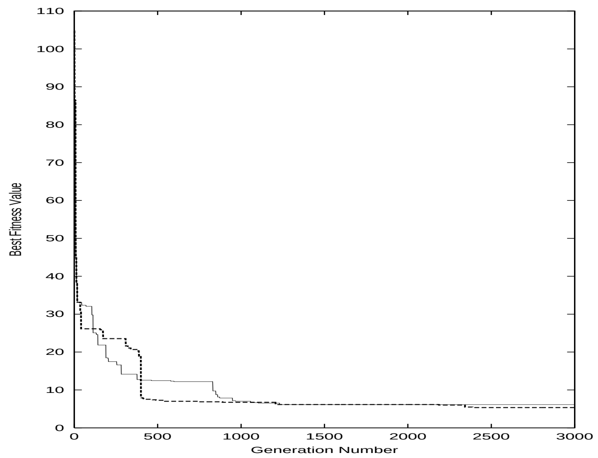

Figure 5: Comparison of controlled search and normal run.

7 Conclusion

In the previous section the initial attempts to formalize the control module and the initial results obtained by this new approach are presented. The progress obtained compared to the classical approach seems to be very motivating.

The current results denote that using the new controlled search provides more efficient searches on the problem domain. The classical approach is able to reach the fitness values that are produced with the new approach with a phase difference. This denotes an increase in the efficiency. Here the processing time consumed by forming the vectors and checking memberships brings in an overhead to the search. However it should be noted that all of the operations related to vector algebra are either linear or logarithmic. It is obvious that such operations would be much more efficient rather then calculating the fitness value of a chromosome for the problem of grammar induction where you have to extract the grammar from the chromosome and test if the grammar parses the sentences in the training set one by one.

The transformation used to map the chromosomes into the space is still trivial. The future work should focus on advanced ways to carry out the transformation where more insights about the global information produced by the interdependency would be obtained.

References

[1] Ricardo Aler and Daniel Borrajo and Pedro Isasi. *Genetic Programming and Deductive-Inductive Learning: A Multi strategy Approach* Proceedings of the Fifteenth International Conference on Machine Learning, ICML'98, 1998, Springer-Verlag.

[2] Lakoff George, *Women, Fire and Dangerous Things*, The University of Chicago Press, 1990.

[3] Juille, H. and Pollack, J. B. (1998). A sampling-based Heuristic for tree search to grammar induction. AAAI-98.

[4] Keller, B. and Lutz, R. (1997). Evolving stochastic context-free grammars from examples using a minimum description length principle. The Workshop on Automata, Inductive Grammatical Inference and Language Acquisition. ICML-97.

[5] Mitchell, Melanie. *An Introduction to Genetic Algorithms.* MIT press ,1996

[6] T. Mitchell and S. Thrun. *Learning Analytically and Inductively.* In Mind Matters: A Tribute to Allen Newell, Steier and Mitchell (eds.), Erlbaum, 1995.

[7] Simon Lucas, *Structuring Chromosomes for Context-Free Grammar Evolution*, Proceedings of first IEEE International Conference on Evolutionary Computation, pp 130-135, June, 1994.

[8] Smith, T.C. and Witten, I.H. (1995) *A genetic algorithm for the induction of natural language grammars*, Proc IJCAI-95 Workshop on New Approaches to Learning for Natural Language Processing, 17-24, Montreal, Canada.