Discovering Phonotactic Finite-State Automata by Genetic Search

Anja Belz
School of Cognitive and Computing Sciences
University of Sussex
Brighton BN1 9QH, UK
email: anjab@cogs.susx.ac.uk

Abstract

This paper presents a genetic algorithm based approach to the automatic discovery of finite-state automata (FSAs) from positive data. FSAs are commonly used in computational phonology, but – given the limited learnability of FSAs from arbitrary language subsets – are usually constructed manually. The approach presented here offers a practical automatic method that helps reduce the cost of manual FSA construction.

1 Background

Finite-state techniques have always been central to computational phonology. Definite FSAs are commonly used to encode phonotactic constraints in a variety of NLP contexts. Such FSAs are usually constructed in a time-consuming manual process.

Following notational convention, an FSA is a quintuple (S, I, δ, s_0, F), in which S is a finite nonempty set of states, I is a finite nonempty alphabet, δ is the state transition function, s_0 is the initial state, and F is a nonempty set of final states in S. The language L accepted by the finite automaton A, denoted $L(A)$, is $\{x|\delta(s_0, x) \in F\}$.

Generally, the problem considered here is that of identifying a language L from a fixed finite sample $D = (D^+, D^-)$, where $D^+ \subseteq L$ and $D^- \cap L = \emptyset$ (D^- may be empty). If D^- is empty, and D^+ is structurally complete with regard to L, the problem is not complex, and there exist a number of algorithms based on first constructing a canonical automaton from the data set which is then reduced or merged in various ways. If D^+ is an arbitrary strict subset of L, the problem is less clearly defined. Since any finite sample is consistent with an infinite number of languages, L cannot be identified uniquely from D^+. "...the best we can hope to do is to infer a grammar that will describe the strings in D^+ and predict other strings that in some sense are of the same nature as those contained in D^+." (Fu and Booth, 1986, p. 345)

To constrain the set of possible languages L, the inferred grammar is typically required to be as small as possible. However, the problem of finding a minimal grammar consistent with a given sample D was shown to be NP-hard by Gold (1978). Nonapproximability results of varying strength have been added since. In the special case where D contains all strings of symbols over a finite alphabet I of length shorter than k, a polynomial-time algorithm can be found (Trakhtenbrot and Barzdin, 1973), but if even a small fraction of examples is missing, the problem is again NP-hard (Angluin, 1978).

2 Task Description

Given a known finite alphabet of symbols I, a target finite-state language L, and a data sample $D^+ \subseteq L \subseteq I^*$, the task is to find an FSA A, such that $L(A)$ is *consistent* with D^+, $L(A)$ is a superset of D^+ encoding *generalisation* over the structural regularities of D^+, and the size of S is as *small* as possible. Where the target language is known in advance, the degree of language and size approximation can be measured, and its adequacy relative to training set size and representativeness can be described. In the case of inference of automata that encode (part of) a phonological grammar, language approximation and its degree of adequacy can be described relative to a set of theoretical linguistic assumptions that describes a target grammar.

3 Search Method

By direct analogy with natural evolution, genetic algorithms (GAs) work with a population

of individuals each of which represents a candidate solution to the given problem. These individuals are assigned a fitness score and on its basis are selected to 'mate', and produce the next generation. This process is typically iterated until the population has converged, i.e. when individuals have reached a degree of similarity beyond which further improvement becomes impossible. Characteristics that form part of good solutions are passed on through the generations and begin to combine in the offspring to approach global optima, an effect that has been explained in terms of the *building block hypothesis* (Goldberg, 1989). Unlike other search methods, GAs sample different areas of the search space simultaneously, and are therefore able to escape local optima and to avoid areas of low fitness.

The main issues in GA design are *encoding* the candidate solutions (individuals) as data structures for the GA to work on, defining a *fitness function* that accurately expresses the goodness of candidate solutions, and designing *genetic operators* that combine and alter the genetic material of good solutions (parents) to produce new, better solutions (offspring).

In the present GA[1], the state-transition matrices of FSAs are directly converted into genotypes. Mutation randomly adds or deletes one transition in each FSA and a variant of uniform crossover tends to preserve the general structure of the fitter parent, while adding some substructures from the weaker parent (offspring can be larger or smaller than both parents). Fitness is evaluated according to three fitness criteria. The first two follow directly from the task description: (1) size of S (*smallness*), and (2) ability to parse strings in D^+ (*consistency*), where ability to partially parse strings is also rewarded. Used on their own, however, these criteria lead search in the direction of universal automata that produce all strings $x \in I^*$ up to the length of the longest string in D^+. To avoid this, (3) an overgeneration criterion is added that requires automata to achieve a given degree of generalisation, such that the size of $L(A)$ is equal to the size of the target language (where the target language is not known, its size is estimated). Transitions that are not required to parse any string in D^+ are eliminated. Fitness criteria 1-3 are weighted (reflecting their relative importance to fitness evaluation). These weights can be manipulated to directly affect the structural and functional properties of automata.

[1]Developed jointly with Berkan Eskikaya, COGS, University of Sussex, and described in more detail in (Belz and Eskikaya, 1998)

S^+ size	Succ. (Avg)	Target found (Avg)	Convergence at gen.	Accurate (Target) automata
8	0	-	201	0
16	3(1.4)	33(49)	211	25(21)
32	6(3.7)	40(67)	172	58(49)
48	10(5.2)	46(83)	171	99(92)
64	8(5.4)	45(68)	191	79(73)
80	9(6.0)	37(73)	142	79(79)
96	9(6.2)	35(75)	114	80(76)
112	9(6.1)	46(73)	158	80(80)
128	9(6.4)	36(77)	135	89(89)

Table 1: Toy data set results.

4 Results

Toy Data Set The data used in the first set of tests was generated with the grammar $S \rightarrow cA$, $A \rightarrow c_1v_1B$, $A \rightarrow c_2v_2B$, $B \rightarrow c$. Here, c abbreviates a set of 4 consonants, c_1 2 sharped consonants, c_2 2 non-sharped consonants, v_1 2 front vowels, and v_2 2 non-front vowels. The grammar (generating a total of 128 strings) is a simple version of a phonotactic constraint in Russian, where non-sharped consonants cannot be followed by front vowels, (e.g. Halle (1971)).

Tests were carried out for different-size randomly selected language subsets. Different combinations of weights for the three fitness criteria were tested. Table 1 gives results for the best weight combination averaged over 10 runs differing only in random population initialisation (and results averaged over all other weight combinations in brackets). The first column indicates how many successful runs there were out of 10 for the best weight combination (and the average for all weight combinations in brackets). A run was deemed successful if the target automaton was found before convergence. The last column shows how many accurate automata were in final populations, and in brackets how many of these also matched the target FSA in size.

	Train.	Test.	General.	States
Best	100%	61%	100	7
Avg	94%	49%	100.3	7.1

Table 2: Results for Russian noun data set.

The general effects of reducing the size of D^+ are that successful runs become less likely, and that the weight assigned to the degree of overgeneration becomes more important and tends to have to be increased. The larger the data set, the more similar the results that can be achieved with different weights.

Russian Noun Data Set The data used in the second series of tests were bisyllabic feminine Russian nouns ending in `-a`. The alphabet consisted of 36 phonemic symbols[2]. The training set contained 100 strings, and a related set of 100 strings was used as a test set.

Results are shown in Table 2. The target degree of overgeneration was set to 100 times the size of the data set. Tests were carried out for different weights assigned to the overgeneration criterion. Results are given for the best automaton found in 10 runs for the best weight settings, and for the corresponding averages for all 10 runs.

Figure 1 shows the fittest automaton from Table 1. Phonemes are grouped together (as label sets on arcs) in several linguistically useful ways. Vowels and consonants are separated completely. Vowels are separated into the set of those that can be preceded by sharped consonants (capitalised symbols) and those that cannot. Correspondingly, sharped consonants tend to be separated from nonsharped equivalents. The phonemes `k, r, t` are singled out (arc 4 $\rightarrow$ 5) because they combine only with nonsharped consonants to form stem-final clusters. The groupings `S` (0 $\rightarrow$ 6) and `L,M,P,R` (6 $\rightarrow$ 1) reflect stem-initial consonant clusters.

These groupings are typical of the automata discovered in all 10 runs. Occasionally `ka` (the feminine diminutive ending) was singled out as a separate ending, and the stem vowels were frequently grouped together more efficiently.

Different degrees of generalisaton were

[2]The set of phonemic symbols used here is based on Halle (Halle, 1971). Capital symbols represent sharped versions of non-capitalised counterparts.

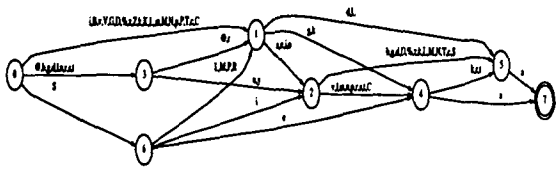

Figure 1: Best automaton for Russian noun set.

achieved with different weight settings. The automaton shown here corresponds most closely to (Halle, 1971).

5 Conclusion and Further Research

This paper presented a GA for FSA induction and preliminary results for a toy data set and a simple set of Russian nouns. These results indicate that genetic search can successfully be applied to the automatic discovery of finite-state automata that encode (parts of) phonological grammars from arbitrary subsets of positive data. A more detailed description of the GA and a report on subsequent results can be found in Belz and Eskikaya (1998). The method is currently being extended to allow for sets of negative examples.

References

D. Angluin. 1978. On the complexity of minimum inference of regular sets. *Information and Control*, 39:337–350.

A. Belz and B. Eskikaya. 1998. A genetic algorithm for finite-state automaton induction. Cognitive Science Research Paper 487, School of Cognitive and Computing Sciences, University of Sussex.

K. S. Fu and T. L. Booth. 1986. Grammatical inference: introduction and survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-8:343–375.

E. M. Gold. 1978. Complexity of automaton identification from given data. *Information and Control*, 37:302–320.

D. E. Goldberg. 1989. *Genetic Algorithms in search, optimization and machine learning.* Addison-Wesley.

M. Halle. 1971. *The Sound Pattern of Russian.* Mouton, The Hague.

B. B. Trakhtenbrot and Ya. Barzdin. 1973. *Finite Automata.* North Holland, Amsterdam.