CHAPTER 41

A Progress Report on Machines to Learn to Translate Languages and Retrieve Information*

R. J. SOLOMONOFF

Zator Company, 140 Mt. Auburn, Cambridge 38, Mass.

I. INTRODUCTION

During the last year and a half, work has progressed on a project to devise a machine to make inductive inferences in certain situations. The present paper will report on those aspects of this work that seem relevant to mechanical translation and information retrieval.

Most of the recent work on this project has been on inductive inference techniques applied to "formal languages." A formal language is defined to be a finite or infinite set of finite sequences of symbols. Each such sequence is called an "acceptable sentence." The symbols themselves are called "words," and are to be taken from a finite vocabulary of words. A "grammar" of such a language may take the form of a set of rules by which all sentences in the language may be generated, or a grammar may take the form of a test to determine if a proposed sequence of words is, indeed, an acceptable sentence.

N. Chomsky (1,2) has described three types of formal languages of increasing complexity and has investigated the properties of each of them. The recent work on inductive inference has involved the language type of intermediate complexity, called a "phrase structure language." In a certain large subclass of such languages (namely, the ones whose grammars do not employ context-dependent substitution), methods have been devised for discovering the grammar of a language after one has been given a finite set of acceptable sentences in that language (3,4). The training situation under which these grammar discovery techniques will work is not a very practical one, but the methods do suggest other more practical

*Work supported by the U. S. Air Force Office of Scientific Research through Contract AF 49(638)-376.

approaches to the problem. The deficiencies of the present methods of grammar discovery, as well as the methods by which these deficiencies may be overcome, will be discussed later.

Grammar discovery is a special case of the more general problem of inductive inference, in which we are given a set of examples of some of the members of a class of objects, and we must try to devise a general rule to describe all members of that class of objects.

The most complex of Chomsky's grammar types, called the "transformation grammar" appears to be adequate for expressing most of the rules of English grammar. It also appears to be possible to express most of the English language with a phrase structure grammer, but more grammar rules would be required—more than for an equivalent transformation grammar.

The concept of language has been generalized to include any patterns that might be of interest in inductive inference. In particular, it has been possible to look upon the problem of learning to translate between two languages as being identical to the problem of discovering the grammar rules of a third language of the more general type. There is a more exact discussion of translation learning in Section 6 of Ref. 3, and we shall discuss this problem in Section II of the present paper.

Another use of the language concept is in information retrieval. A very simple example is the mechanization of the assignment of descriptors or other search indices to documents, abstracts, or titles of documents. Suppose we have a set of documents about sulfur refining. Each of these documents may then be considered as a single sentence in a language of unknown grammar. In certain cases, we can discover a grammar of such a language, and thereby implement the mechanized assignment of documents to the category "sulfur refining." Section III discusses the problems that must be solved before such a mechanization of descriptor assignment can be realized in any useful way.

The techniques for grammar discovery that have been devised, and the new methods that are being investigated, all consist of well-defined deterministic rules. As such, they can be used as instructions for present-day general purpose digital computers. It is clear, then, that all of our work on devising rules for discovering grammars or discovering formation rules of patterns of more complex types is immediately applicable to the programming of present-day computers to accomplish these tasks without human intervention. The practicality of such programming at the present time will be discussed.

In the present paper, when we speak of a "machine" to accomplish a certain task, we will refer not to a physical mechanism but to a set of instructions by which a general purpose digital computer could be made to accomplish this task.

II. A MACHINE TO LEARN TO TRANSLATE LANGUAGES

Ideally, we would like to have a machine of the following type: The machine is given a large number of sentences in a language that we will call L_1. For each of these sentences in L_1, the machine is also given the sentence in a second language L_2 that is an acceptable translation of the sentence in L_1. The machine then analyses this set of sentence pairs, and devises a set of rules that relate a sentence of L_1 to the corresponding sentence in L_2. If we then present the machine with an entirely new sentence in L_1 (using, of course, no words or meanings of words that have not already been presented to the machine), the machine will use its set of rules to translate this sentence into a corresponding sentence in L_2.

Clearly, there will be restrictions on the operation of such a machine. The sentence pairs upon which the machine bases its rules must be sufficiently numerous so that they include in their structures all of the translation rules for the two languages. The translations will not be very good ones if we limit excessively the complexity of the translation rules that the machine is able to devise.

At the present time, we cannot construct such a machine. The following discussion will describe a machine that we can construct, which appears to be a significant step in the desired direction. We shall also discuss the problems that must be overcome before we can build the machine that we would like.

In the analysis method used, we consider two languages, L_1 and L_2, such that there exists a translation rule between them. In the simplest case, this will mean that there is a one-to-one correspondence between the acceptable sentences in L_1 and the acceptable sentences in L_2.

Let us then construct a third "language," L_3, in the following way: The "acceptable sentences" in L_3 consist of ordered pairs of sentences. The first sentence in each pair is a sentence from L_1 the second is the corresponding sentence in L_2.

To speak of L_3 as a "lanugage" implies a certain generalization of the concept of language. The "words" of L_3 may be pairs of ordinary words: the "phrases" of L_3 may be pairs of ordinary phrases—or there may be no useful way to divide the sentences of L_3 into words and phrases.

Knowing the grammar of L_3 will then be equivalent to being able to translate from L_1 to L_2. Here, "knowing the grammar" may mean that if a proposed sentence pair is given us, we can use the grammer rules to determine whether that sentence pair is in L_3 or not. Another, more useful kind of grammar is one that gives methods to determine all possible legal sentences that could contain a certain fixed phrase as part of them. In this latter case, we need

only present the grammar rules with a sentence from L_1, and if the grammar rules are to complete this partial sentence of L_3 to form an acceptable complete sentence in L_3, they must give us the translation in L_2 of the sentence in L_1.

The above discussion would be of little interest if the grammar rules of L_3 were unreasonably complex. However, if L_1 and L_2 are phrase structure languages, it is sometimes possible to look upon L_3 as a kind of generalized phrase structure language. It is also sometimes possible to devise translation rules utilizing a translation language L_4, which is a phrase structure language, in which neither L_1 nor L_2 are phrase structure languages. In all such cases, L_3 is included in L_4. A simple example of such a situation is one in which L_1 and L_2 are identical, and are essentially non-phrase structure languages. In such a case L_3 is not a phrase structure language, yet it can be easily imbedded in a larger L_4 that is a simple phrase structure language.

At the present time, we can construct a machine that will be able to discover the grammar rules of any phrase structure language not employing context-dependent substitution. To discover the grammar of such a language, the machine must be first given an "adequate" set of sentences in the language. The machine analyses these sentences, then devises various hypotheses as to what the grammar might be. The machine then uses these hypotheses to create hypothetical "trial" sentences to test the hypotheses.

These trial sentences are presented as one output of the machine. An external "teacher" is required to tell the machine which of the trial sentences are acceptable sentences within the language of interest.

If there are two languages, L_1 and L_2, such that there exists a suitably generalized phrase structure language that can translate between L_1 and L_2, then we can at the present time, construct a a machine that will learn to translate between L_1 and L_2.

Since much of English, and perhaps much of some other languages, are expressible by phrase structure grammars, it would appear to be worth while to construct such a machine. However, let us examine more carefully the characteristics of such a device.

First, many of the phrase structure grammar rules that have been constructed for English employ context dependent substitution. At the present time, we do not know whether an adequate set of phrase structure rules not employing content dependent substitution can be devised for any pair of existing ethnic languages. Even if we had such a pair of ethnic languages, the number of questions (i.e., pairs of trial translation sentences) would be quite large, so that it would probably be impractical to utilize a human teacher to respond to the machine's questions. A human teacher would be too slow and so the learning would take too much time.

A further factor that rules out a human teacher is the fact that any errors made by the teacher are likely to make it impossible for

the machine to discover the correct translation grammer. We would find few humans who felt they knew translation well enough to be able to decide whether a set of translations of a very large set of sentences were right or wrong. That we would be able to find such a human who would make no errors is fairly unlikely.

Another serious difficulty is that when we use a phrase structure grammar for English, the number of grammar rules is quite large. The number of phrase structure grammar rules needed for translation is also likely to be very large, so the amount of time needed by the machine, and the number of questions it would ask would be correspondingly large.

At the present time, however, it is possible to devise wholly synthetic languages that are intertranslatable through a phrase structure formalism. In cases in which we know the "translation grammar" we can set up machines to act as teachers. These machines can answer questions very rapidly, and need make no mistakes. As a result, if we already know the phrase structure translation rules of a pair of languages, we can teach them to a grammar discovery machine through a suitable training sequence. This is as well as we can do at the present time.

How can we best overcome the deficiencies of our simple translation learning machine? Two closely related methods have been investigated to some extent, and are both quite promising.

The first method involves the concept of "approximation languages"; the second, the concept of "stochastic languages."

Suppose we were given a large set of sequences of symbols that were known to be "acceptable sentences," and another large set of strings of symbols that were "unacceptable sentences." One kind of problem of devising an "approximation language" would be to find a phrase structure language of limited complexity (e.g., one with less than twenty rules in its grammar) such that as many of the acceptable sentences and as few of the unacceptable sentences as possible were included in the language described by the grammar. We must, of course, assign suitable weights to the inclusion and noninclusion of sentences in the language, so that we have a suitable "goodness of fit" criterion for any language that we may try. Our goal, then, would be to find a language that fits the sentences in an optimum manner. We can associate with each language a probability of correctness for its predictions of the inclusion or noninclusion of various sentences in the set of interest.

It will be noted that the problem of devising an approximation language to fit a set of sentences bears a close correspondence to the problem of devising an optimum curve to fit some data points. Limiting the complexity of a language corresponds to limiting a curve to, say, five adjustable parameters.

Applying the approximation language idea to mechanical translation, we can obtain translation languages (the L_3 of the previous discussion) that have relatively few grammar rules, but the trans-

lations that are obtained are poor. As we increase the number and/or kinds of grammar rules that are allowable in the translation language, the average quality of our translations improve.

In order to learn mechanical translation in this approximate manner, a machine would have to have as initial input data a large set of sentences in one language and the correct translations of them into another language. It would also be necessary to furnish a set of translations that were known to be incorrect. In addition, it would be necessary to give the machine the desired limits on the complexity of the translation language. After having been given this input data, the machine would ask no questions, but would proceed directly to compute an optimum translation language of the required complexity.

Approximation languages are discussed somewhat more rigorously in Appendix I.

The concept of "stochastic languages" is closely related to that of approximation languages. A stochastic language consists of a finite or infinite set of sequences of symbols, with a positive real number assigned to each of the sequences. These numbers give the probabilities of their corresponding sequences. All sequences not previously assigned numbers are given probability zero.

One application of a stochastic language is to determine the relative probabilities of all possible completions of a sentence once we are given part of the sentence.

If we are given a set of sentences that are all members of some stochastic language of unknown grammar, then we can, if we know the a priori likelihood of all grammars of interest, use Bayes' Theorem to determine the likelihood that any particular stochastic grammar produced these sentences. In this way, we can determine the most likely grammar that could have produced the sentences of interest. It is also possible to use the probabilities of these stochastic grammars to determine the liklihood that any new proposed sentence would be created by the same language that created the set of sample sentences. A more detailed mathematical treatment of these ideas is given in Appendix II.

We can use stochastic translation languages to solve the problem of mechanical translation, given a suitably large set of sentences and their translations. It is also required that we have some sort of a priori likelihood assignment to all translation languages that we might be interested in. As a solution to this mechanical translation problem, we would obtain, not a single translation of the sentence to be translated, but a set of possible translations, with with a probability of correctness being assigned to each translation. We can, if we like, select the translation of greatest "probability of correctness" and be satisfied with it—or we may ask for a somewhat more complex machine output.

While an acceptable formal mathematical solution to the above stochastic language problem can be written (a tentative assignment of a priori liklihoods to various translation grammars has been devised), a practical solution to the resulting equations has not yet been found. Present progress on this problem is, however, quite promising.

III. A MACHINE TO LEARN INFORMATION RETRIEVAL

The information retrieval learning machine described in Section I could be constructed if we had a workable solution to the problem of approximate language construction, or the problem of finding an optimum stochastic language to fit a given set of acceptable sentences. At the present time, it is possible for a kind of retrieval learning machine to be programmed on existing computers. However, such a machine would require a teacher who knew of a phrase structure language to describe each of the documents appropriate to each descriptor of interest. As in the case of mechanical translation, the number of questions asked of the teacher would be excessively large and any mistakes made by the teacher would probably make the machine unable to discover the correct grammar.

Approximation languages and stochastic languages are, however, particularly well adapted to the information retrieval problem. This is so becuase, first of all, the exact syntactic qualities of a document that make it likely to be relevant to, say sulfur refining, are fairly complex, and it is unlikely that the exact language would be discovered by a machine of reasonable size, in a reasonable length of time. However, present work on approximation and stochastic languages is oriented toward finding languages that will give the best possible results within the limitations of computer capacity and computation time.

The finding of exact languages for retrieval is also made less likely, in view of the fact that the categorizations of documents that are presented to the machine as a training sequence will not be performed altogether consistently by the human cataloger.

Lastly, a stochastic language has particular applicability to information retrieval, in that a machine using such languages will assign a probability to the relevance of any descriptor to any document. A machine of this sort can then be asked for documents satisfying certain requirements with the user assigning weights to the importance of each requirement. The machine can then give a list of documents in order of probability that they will satisfy the desired requirements. Such a machine would save the user much time when the bibliographies were very large. Also, the machine would list the documents most likely to be relevant, even if this likelihood were extremely low. A machine using nonstochastic languages would often, in similar cases, list no documents at all.

IV. SUMMARY

Some machines have been described that are to be able to learn to translate languages and to retrieve information from a collection of documents. At the present time, theoretical work has progressed to the point at which it is possible to program a present-day digital computer to learn mechanical translation. However, the pairs of languages between which translation could be learned would be very limited. Also, a "teacher" would be required who was able to translate without error, and was able to answer rapidly an enormous number of questions asked by the machine.

A machine that would learn to retrieve information could also be programmed at present, but it would have similar limitations.

Work is now progressing in the study of approximation languages and stochastic languages. By using languages of these types, it appears to be possible to eliminate the difficulties mentioned above, and to design physically realizable machines to learn translation and information retrieval.

In the information retrieval problem, a machine using stochastic languages could be programmed to list documents in order of probability of relevance to a listed set of requirements.

APPENDIX I. APPROXIMATION LANGUAGES

Let R_1 be a set of sequences of symbols that we want to include in our language, and let R_2 be a set of sequences of symbols that we want to be excluded from our language.

If L_1 is a language, then let $N(L_1 \cap R_1)$ denote the number of sequences of symbols that L_1 and R_1 have in common.

For the language L_1, the expression $a\,N(L_1 \cap R_1) - b\,N(L_1 \cap R_2)$ is one possible "goodness of fit" criterion. Here the constants "a" and "b" give relative weights assigned to correct and incorrect judgments made by language L_1. Different "goodness of fit" criteria must be devised for each possible application of the approximation language desired.

Suppose that $[L_1]$ $(i = 1,2,\ldots,n)$ is a finite set of languages, each of which has a "complexity" less than a certain threshold. One possible measure of the complexity of a language is the number of rules in its grammar. To use this measure, we must stipulate the form in which the grammar may be expressed. Complexity is discussed at greater length in Appendix II.

Our problem, then, is to search through the set $[L_i]$ to find a language with maximum "goodness of fit." Since the set $[L_i]$ is usually very large, we must devise various heuristic procedures to direct our search into directions that are likely to be successful as early as possible.

In some problems we may want to find a language such that the ratio of goodness of fit to complexity is maximized.

APPENDIX II. STOCHASTIC LANGUAGES

Suppose $[\alpha_i]$ $(i = 1,2,\ldots,n)$ is a set of n sequences of symbols, that constitutes a sample of n sentences to be analyzed. Each α_i is a sentence and we are trying to find a language which in some sense best fits this sample. One language that includes all of these sequences is the universal language, L_0. L_0 contains all finite sequences of the symbols used in the α_i's. L_0 is also a phrase structure language (and a finite state language) of fairly low complexity, since there are relatively few rules in its grammar, if it is expressed in the form of substitution rules.

While it is clear that L_0 "fits" the set of sequences $[\alpha_i]$, in the sense of including all of them, it is also clear that L_0 is not particularly characteristic of the set $[\alpha_i]$, since L_0 includes any other sets of sequences utilizing the same set of symbols.

Let us consider L_1, a finite language consisting of all of the sequences in $[\alpha_i]$ and no others. While it is clear that L_1 is very narrow in its specification of the sequences of $[\alpha_i]$, it is also clear that if n is very large (as it will be in most practical cases), the description of L_1 (i.e., the grammar of L_1) obtained by listing its members will be an exceedingly "complex" description, in the sense of having very many "grammar rules."

It is also clear that neither L_0 nor L_1 is any good for extrapolation from the initial set, $[\alpha_i]$, though for different reasons.

What we seek is some sort of compromise between L_0 and L_1. We want a language that, in some sense, specifies the set $[\alpha_i]$ as narrowly as possible, and yet we want the description (i.e., the grammar) of this language to be as simple as possible (i.e., contain few rules).

A particularly useful quantification of the notion of "specificity" is given by stochastic languages. Suppose L_j is a stochastic language that assigns probability p_{ji} to the sequence α_i. Furthermore, suppose the p_{ji}'s to be normalized over all sequences included in L_j, i.e., $\Sigma_i\, p_{ij} = 1$.

If we select sequences from an L_j at random, and the probability that sequence α_i will be chosen is p_{ji}, then the probability that the entire set $[\alpha_i]$ will be chosen in any order by n selections from L_j is:

$$\Pr([\alpha_i] \mid L_j) = n! \prod_{i=1}^{n} p_{ji}$$

We denote this quantity by P_j. This is the conditional probability of the selection of the entire set $[\alpha_i]$, given L_j.

The language L_j that "most narrowly specifies" the set $[\alpha_i]$ is defined to be the one whose P_j is maximum.

We will use the a priori likelihood of a language as a quantification of complexity. Simple languages have high a priori probability; complex languages have low a priori probability.

Designate Q_j as the a priori probability of language L_j, i.e., $Q_j = Pr(L_j)$. Then

$$Pr(L_j \mid [\alpha_i]) = \frac{Q_j P_j}{\sum_i Q_i P_i}$$

is the a posteriori probability that L_j generated $[\alpha_i]$. The summation extends over all possible values of i.

Suppose β is a sequence of symbols. β may or may not be included in the set $[\alpha_i]$. $p_{j\beta}$ is the probability assigned to sequence β by language L_j. Then our system would assign the value

$$\frac{\sum_j Q_j P_j p_{j\beta}}{\sum_j Q_j P_j}$$

to the probability that β would be generated by the same language that created the set $[\alpha_i]$.

The summations extend over all possible values of j. It is to be noted that the L_j's are mutually exclusive causes in the sense that only one L_j was the cause of the set $[\alpha_i]$.

In order to make this equation more concrete, some examples of stochastic languages will be given. These will make possible the evaluation of the p_{j_i}'s. We shall also discuss some possible methods of assigning Q_j's to various grammars.

A kind of stochastic language that has been studied at great length is the Markov chain. In one form of Markov process, we have a machine that is initially in state S_0, but may eventually pass through one or more of the n states S_i $(i = 0,1,\ldots,n-1)$. From state S_0, the machine may pass to certain of the n states. From this new state, the machine may pass to any of several states, and from there to any of several states, and so on, until it has returned to its initial state, S_0. In passing from one state to another, the machine emits transition symbols, a_j $(j = 1,2,\ldots,m)$. The strings of symbols emitted in the intervals between returns to the initial state, S_0, constitute acceptable sentences in a stochastic language.

The rules for the transitions between states and the emission of transition symbols are simple probabilistic rules. If the machine is in state S_i, then the probability that it will go to S_j next is given by the fixed matrix element M_{ij}. If the machine passes from state S_i to state S_j, the probability that it will emit transition symbol a_k is given by the constant T_{ijk}.

If the transition symbols are English words, we can construct a stochastic grammar that generates sentences very much like English. Chomsky has shown (Ref. 1, Section 2.3) that a deterministic Markovian grammar cannot be an adequate grammar for English. It is likely that a stochastic Markovian grammer is likewise deficient.

A more interesting grammar type is the stochastic phrase structure grammar. One way to describe a deterministic phrase structure language (Ref. 1, Section 3.2) is to give a single initial string of symbols, and a set of substitution rules for various symbols. After the substitution rules have been applied several times, a string of of symbols will result in which no further substitutions can be made. An example of a simple deterministic phrase structure language is one that starts with the initial symbol S and has the following set of permissable substitution rules:

S ⟶ AB	A ⟶ Ac	B ⟶ cA
S ⟶ Bd	A ⟶ cB	B ⟶ b
	A ⟶ a	

Here we read "S ⟶ AB" as "S may be replaced by AB."

A permissable derivation of an acceptable sentence is:

1. S	5. ccAcA
2. AB	6. cccBcA
3. AcA	7. cccBca
4. cBcA	8. cccbca

The last string of symbols, cccbca, is an acceptable sentence, since we can make no further substitutions in it.

It will be noted that for each of the symbols S, A, and B, there is more than one substitution possible. By assigning probabilities to each of these substitutions, we describe a stochastic phrase structure grammar.

A possible assignment of probabilities in the previous grammar is

S ⟶ AB , 0.1	A ⟶ Ac , 0.2	B ⟶ cA , 0.3
S ⟶ Bd , 0.9	A ⟶ cB , 0.2	B ⟶ d , 0.7
	A ⟶ a , 0.6	

The number written after each substitution rule is the probability value assigned to that substitution. In the derivation of the sentence cccbca, the substitutions S ⟶ AB, B ⟶ cA, A ⟶ cB, B ⟶ cA, A ⟶ cB, A ⟶ a, and B ⟶ b were used in that order. These substitutions have the respective probabilities of 0.1, 0.3, 0.2,

0.3, 0.2, 0.6, and 0.7. The resultant probability of the sentence cccbca is

$$0.1 \times 0.3 \times 0.2 \times 0.3 \times 0.2 \times 0.6 \times 0.7 = 0.0001512$$

One way to assign an a priori probability to a stochastic phrase structure language is to first assign an a priori probability to the corresponding deterministic phrase structure language. Associated with each such deterministic language is a continuous multidimensional space of stochastic languages, each point of which corresponds to a set of possible values of the substitution probabilities. We may assume a uniform a priori probability distribution over this space.

The problem of assigning an a priori probability to a stochastic phrase structure language thereby becomes one of assigning an a priori probability to the corresponding deterministic phrase structure language.

A possible method is to make the a priori probability of a phrase structure language some decreasing function of the number of substitution rules in the simplest grammar (i.e., having the fewest rules) of the language.

If we restrict our grammar rules to binary substitutions (of the form $A \longrightarrow Bc$, then there are

$$\binom{m(m+n)\ (m+n+1)}{r}$$

grammars of phrase structure languages that have r substitution rules, n terminal symbols (symbols for which no substitutions can be made) and m intermediate symbols (symbols for which substitutions must be made). It can be shown that confining ourselves to binary substitutions imposes no restrictions on the kinds of languages than can be generated.

We have the additional constraint that $r \geq m$, since each intermediate symbol must have at least one substitution rule associated with it.

It should be noted that the formula given above is really an upper bound for the number of grammars, since many of the languages described by the grammars included in the formula are identical to each other. Also, many of the languages included may be described by simpler grammars.

In order that the total a priori probability of all languages under consideration be unity, it is necessary that the a priori probability of languages decrease significantly more rapidly with r than does the number of possible grammars. For this purpose, an upper bound on the rate of increase of the number of grammars as a function of r may be all that is needed in order to devise a suitable a priori probability assignment.

REFERENCES

1. Chomsky, N., Three Models for the Description of Language, "IRE Transactions on Information Theory," Vol. IT-2, Proceedings of the Symposium on Information Theory, September 1956, pp 113–124.
2. Chomsky, N., "Syntactic Structures," Mouton and Co., 's-Gravenhage (The Hague), Netherlands, 1957.
3. Solomonoff, R. J., The Mechanization of Linguistic Learning, "Transactions," Second International Conference on Cybernetics, Association for Cybernetics, Namur, Belgium, September 3–10, 1958. (Published as ZTB-125 by Zator Company, Cambridge, Mass., April 1959. AFOSR-TN-59-246; ASTIA Document No. AD 212 226).
4. Solomonoff, R. J., A New Method for Discovering the Grammars of Phrase Structure Languages, "Transactions," International Conference on Information Processing, UNESCO House, Paris, France, June 13–23, 1959. (Published as ZTB-124 by Zator Company, Cambridge, Mass., April 1959. AFOSR-TN-59-110; ASTIA Document No. AD 210 390).