SYNTHESIS OF A PROBABILISTIC FINITE-STATE GRAMMAR DESCRIBING A GIVEN SET OF SEQUENCES

N. A. Esin and M. I. Shlezinger

UDC 007:621.391.19

Probabilistic Grammar. Posing the Problem

A probabilistic finite-state grammar is a particular case of the probabilistic grammars considered in [3] and applied in problems of recognition of signals distorted by noise.

A probabilistic finite-state grammar is specified by the quadruple $G = \langle S, V, P, p\rangle$. We denote by S and V the finite sets called, respectively, the nonterminal and terminal alphabets. We denote by p the probability distribution on set S, i.e., the function p(s) such that $p(s) \geq 0$, $s \in S$, and $\sum_{s\in S} p(s) = 1$. We denote by P the conditional probability distribution on set $S \times V \times S$, i.e., function $p(s', v/s)$ such that $\sum_{s'\in S}\sum_{v\in V} p(s', v/s) = 1$ $(s \in S)$ and $p(s', v/s) \geq 0$ $(s' \in S, v \in V, s \in S)$.

The process of generating a sequence of length n which belongs to given grammar G is implemented as follows.

On the zeroth step, in accordance with probability distribution p(s), there is randomly selected nondeterministic symbol s_0.

Prior to the beginning of the i-th step $(0 < i \leq n)$, let there have been obtained the sequence $v_1, v_2, v_3, \ldots, v_{i-1}, s_{i-1}$, containing only the one nondeterministic symbol s_{i-1} at the end of this sequence. On the i-th step, in accordance with the probability distribution $p(s, v/s_{i-1})$, the random pair (v_i, s_i) is determined and the substitution $s_{i-1} \to v_i, s_i$ is used, i.e., the sequence $v_1, v_2, v_3, \ldots, v_{i-1}s_{i-1}$ is replaced by the sequence $v_1, v_2, v_3, \ldots, v_{i-1}, v_i, s_i$. After n steps we will have obtained the sequence $v_1, v_2, v_3, \ldots, v_{n-1}, v_n, s_n$ to which, on the (n + 1)-th step, substitution $v_n s_n \to v_n$ is applied.

Let $m = (m_0, m_1, \ldots, m_n)$ be a sequence of substitutions such that $m_i = (s_{i-1} \to v_i, s_i)$. For any such sequence we can compute its probability with the aforementioned method of generating the word $p(m) = p(s_0) \cdot \prod_{i=1}^{n} p(s_i, v_i/s_{i-1})$.

Any sequence m generates a unique word V = f(m). We denote the set of sequential substitutions generating word V by M(V). It is obvious that the probability of generating word V of probabilistic grammar G equals

$$P_G(V) = \sum_{m\in M(V)} p(m) = \sum_{s_0\in S}\sum_{s_1\in S}\cdots\sum_{s_n\in S} p(s_0)\prod_{i=1}^{n} p(s_i, v_i/s_{i-1}). \tag{1}$$

Thus, in contradistinction to simple grammars generating sets of words, a probabilistic grammar generates a probability distribution on a set of words.

In the given paper we shall consider an insignificant generalization of such probabilistic-grammar constructions. Specifically, we shall consider the case when probability distribution P, in the process of generating words, does not necessarily remain constant, but may also vary, in dependence on the ordinal number of the step i. In this case, the probability of generating word V is expressed by the formula

$$P_G(V) = \sum_{s_0}\sum_{s_1}\cdots\sum_{s_n} p(s_0)\prod_{i=1}^{n} p_i(s_i, v_i/s_{i-1}), \tag{2}$$

which differs from formula (1) in that the latter is written under the assumption that $p_i(s_i, v_i/s_{i-1}) = p(s_i, v_i/s_{i-1})$. Such a probabilistic construction will be specified by the alphabets S and V, function p, and functions P_i $(i = 1, 2, \ldots, n)$. Henceforth, instead of writing P_i $(i = 1, 2, \ldots, n)$, we shall use the notation $\{P_i\}$.

Translated from Kibernetika, No. 4, pp. 116-120, July-August, 1977. Original article submitted June 2, 1975.

The problem solved in the present paper is the following.

We are given set of words $V^1, V^2, \ldots, V^l$. Each such word is presented by a sequence of terminal symbols of length n; v_i^j (i = 1, 2, . . . , n; j = 1, 2, . . . , l) is the i-th component of the j-th word. It is known that each of the aforementioned words is generated by grammar G. For this grammar, there are known the alphabets **S** and **V**, while functions p and $\{\mathbf{P}_i\}$ are unknown. It is required to find functions p and $\{\mathbf{P}_i\}$ providing maximum probability $\prod_j P_G(V^j)$.

In other words, it is required to find functions p and $\{\mathbf{P}_i\}$ providing a maximum to the functional

$$F(\mathbf{p}, \{\mathbf{P}_i\}) = \prod_{j=1}^{l} \sum_{s_0} \sum_{s_1} \cdots \sum_{s_n} p(s_0) \prod_{i=1}^{n} p_i(s_i, v_i^j/s_{i-1}). \tag{3}$$

Analysis of the Problem

Function $P_G(V)$, expressed earlier by formula (2), can be written in a somewhat different form:

$$P_G(V) = \sum_{s_0} \sum_{s_1} \cdots \sum_{s_n} p(s_0) \prod_{i=1}^{n} p_i(s_i/s_{i-1})\, p_i(v_i/s_{i-1}, s_i), \tag{4}$$

where

$$p_i(s_i/s_{i-1}) = \sum_{v_i} p_i(s_i, v_i/s_{i-1}), \tag{5}$$

$$p_i(v_i/s_{i-1}, s_i) = \frac{p_i(s_i, v_i/s_{i-1})}{p_i(s_i/s_{i-1})}. \tag{6}$$

If we introduce the notation

$$S = (s_0, s_1, \ldots, s_n), \tag{7}$$

$$P(S) = p(s_0) \prod_{i=1}^{n} p_i(s_i/s_{i-1}), \tag{8}$$

$$P(V/S) = \prod_{i=1}^{n} p_i(v_i/s_{i-1}, s_i), \tag{9}$$

then function $P_G(V)$ assumes the form $P_G(V) = \Sigma_S P(S) P(V/S)$.

The logarithm of function $F(\mathbf{p}, \{\mathbf{P}_i\})$ which should be maximized with respect to the parameters p and $\{\mathbf{P}_i\}$, will have the form

$$L(\mathbf{p}, \{\mathbf{P}_i\}) = \sum \ln \Sigma_S P(S/\mathbf{p}, \{\mathbf{P}_i\})\, P(V/S, \{\mathbf{P}_i\}). \tag{10}$$

The problem of maximization of expressions similar to (10) was treated in [1] where it was called the problem of self-teaching.

The problem treated in the present paper differs from that considered in [1] in the following features.

1. In [1], on the a priori distribution P(S), apart from the constraints $P(S) \geq 0$; $\Sigma_S P(S) = 1$, no additional constraints were imposed.

In the present paper, probability distribution P(S) must necessarily have the form (8).

2. In [1], each conditional probability distribution P(V/S), for distinct S, was unambiguously characterized by its own group of parameters.

This means that the choice of distribution P(V/S) when $S = S_1$ in no way depends on the choice of this distribution when $S = S_2$. In the present paper, however, the parameters we seek, $\{\mathbf{P}_i\}$, simultaneously affect all the distributions P(V/S), $S' \in \underbrace{\mathbf{S} \times \mathbf{S} \times \ldots \times \mathbf{S}}_{n+1}$.

3. The algorithm presented in [1] required the computation of $l \times k$ numbers, where l is the quantity of words in the teaching sample while k is the quantity of values of S. In the present paper, S assumes $|\mathbf{S}|^{n+1}$ values, where $|S|$ is the number of symbols in alphabet S, while n is word length. Therefore, the algorithm

presented in [1] cannot be used for the solution of the problem considered here, even when the first two distinctions were not relevant.

Despite these differences, the ideas of solving the problem of synthesis of grammar are very close to the ideas of solving the self-teaching problem, and, in some respects, its development.

Assertion 1. Let $\alpha(S, j)$ be any function such that $\alpha(S, j) \geq 0$, $\Sigma_S \alpha(S, j) = 1$. Then for the likelihood function $L(p, \{P_i\})$, one can write the equations

$$L(p, \{P_i\}) = \sum_j \ln \sum_S P(S/p, \{P_i\}) P(V^j/S,$$

$$\{P_i\}) = \sum_S \sum_j \alpha(S, j) \ln(P(S/p, \{P_i\}) \tag{11}$$

$$\times P(V^j/S, \{P_i\})) - \sum_j \sum_S \alpha(S, j) \ln \frac{P(S/p, \{P_i\}) P(V^j/S, \{P_i\})}{\Sigma_S P(S/p, \{P_i\}) P(V^j/S, \{P_i\})}.$$

Assertion 2. Let p' and $\{P_i'\}$ be some values of parameters p and $\{P_i\}$. Let $\alpha(S, j)$ be the quantity defined by the formula

$$\alpha'(S, j) = \frac{P(S/p', \{P_i'\})}{\sum_S P(S/p', \{P_i'\})} \frac{P(V^j/S, \{P_i'\}}{P(V^j/S, \{P_i'\}}. \tag{12}$$

Also, let p'' and $\{P_i''\}$ be values of parameters p and $\{P_i\}$ which are not equal to p' and $\{P_i'\}$ and which do provide a maximum to the first term on the right of (11); i.e.,

$$(p'', \{P_i''\}) = \arg\max_{(p, \{P_i\})} \sum_S \sum_j \alpha'(S, j)$$

$$\times \ln P(S/p, \{P_i\}) P(V/S, \{P_i\}). \tag{13}$$

Then $L(p'', \{P_i''\}) > L(p', \{P_i'\})$.

This inequality is proved analogously to Theorem 1 of [1].

From the validity of these assertions follows an unrealizable algorithm for maximization of the likelihood function.

Let p^0, $\{P_i^0\}$ be some initial values of the unknown parameters.

Let p^{t-1}, $\{P_i^{t-1}\}$ be the values of the parameters obtained by the beginning of the t-th step of the algorithm operations.

The step with ordinal number t amounts to the performance of two steps.

Step I. Compute the quantities $\alpha^{t-1}(S, j)$ for all pairs (S, j) by formula (12).

Step II. Compute quantities p^t, $\{P_i^t\}$ by formula (13).

The given algorithm cannot be realized directly, since it requires calculation of $l \times |S|^{n+1}$ numbers, which is ridiculous. Therefore, we consider below an algorithm which is equivalent to the one just cited, but which requires a smaller volume of computation.

Analysis of Second Step of the Algorithm

Let be given the sequence of words $V^1, V^2, \ldots, V^j, \ldots, V^l$. Each word constitutes a sequence of terminal symbols of length n:

$$V^j = v_1^j, v_2^j, \ldots, v_i^j, \ldots, v_n^j.$$

Let function $\alpha^{t-1}(S, j)$ be known and given for any $j = 1, 2, \ldots, l$ and for any sequence S of nonterminal symbols of length n + 1: $S = s_0, s_1, s_2, \ldots, s_i, \ldots, s_n$.

The meaning of the second step of the algorithm for grammar synthesis is the search for values p^t and $\{P_i^t\}$ of parameters p and $\{P_j\}$ maximizing the expression $\sum_S \sum_j \alpha(S, j) \ln P(S/p, \{P_j\}) P(V^j/S, \{P_i\})$. In other words, on the second step of the algorithm operations, it is required to find functions $p^t(s_0)$, $p_i^t(v_i/s_{i-1}, s_i)$, and $p_i^t(s_i/s_{i-1})$ providing a maximum to the functional:

$$D(\mathbf{p}, \{\mathbf{P}_i\}) = \sum_j \sum_{s_0} \sum_{s_1} \cdots \sum_{s_n} \alpha^{t-1}(s_0, s_1, \ldots, s_n, j) \ln p(s_0) \prod_{i=1}^{n} p_i(s_i/s_{i-1}) \, p_i(v_i^j/s_{i-1}, s_i). \qquad (14)$$

Maximization of this expression is based on the following assertion of [1].

Assertion 3. Let α_i be fixed positive numbers; let x_i be numbers such that $\sum_i x_i = 1$. A maximum of the expression $\sum_i \alpha_i \ln x_i$ is achieved when $x_i = \frac{\alpha_i}{\sum_j \alpha_j}$.

For our subsequent exposition, we introduce the notation

$$b_0(s_0, j) = \sum_{s_1} \sum_{s_2} \cdots \sum_{s_n} \alpha(s_0, s_1, s_2, \ldots, s_n, j),$$

$$b_i(s_{i-1}, s_i, j) = \sum_{s_0} \sum_{s_1} \cdots \sum_{s_{i-2}} \sum_{s_{i+1}} \cdots \sum_{s_n} \alpha(s_0, s_1, s_2, \ldots, s_n, j). \qquad (15)$$

Replacing the logarithm of the product in (14) by the sum of the logarithms, we obtain the following expression for $D(\mathbf{p}, \{\mathbf{P}_i\})$:

$$D(\mathbf{p}, \{\mathbf{P}_i\}) = \sum_j \sum_{s_0} b_0(s_0, j) \ln p(s_0) + \sum_j \sum_i \sum_{s_{i-1}} \sum_{s_i} b_i(s_{i-1}, s_i, j) \ln p_i(s_i/s_{i-1})$$

$$+ \sum_j \sum_i \sum_{s_{i-1}} \sum_{s_i} b_i(s_{i-1}, s_i, j) \ln p_i(v_i^j/s_{i-1}, s_i), \qquad (16)$$

On the basis of Assertion 3, the maximum of the first term in expression (16) is reached when

$$p(s_0) = \frac{\sum_j b_0(s_0, j)}{\sum_j \sum_{s_0} b_0(s_0, j)}. \qquad (17)$$

On the basis of this same assertion, the maximum of the second term in (16) is reached when

$$p_i(s_i/s_{i-1}) = \frac{\sum_j b_i(s_{i-1}, s_i, j)}{\sum_j \sum_{s_i} b_i(s_{i-1}, s_i, j)}. \qquad (18)$$

In order to maximize the third term in (16), it is necessary to transform this term somewhat. We partition set J of the values of index j into subsets $J(v, i)$, $v \in \mathbf{V}$; i.e., $J = \bigcup_{v \in \mathbf{V}} J(v, i)$. By $J(v^*, i)$ we shall understand the set of ordinal numbers of the words in which symbol v^* is in the i-th position.

It is obvious from this that operator $\sum_{j \in J}$ can be replaced by the operator $\sum_{v \in \mathbf{V}} \sum_{j \in J(v, i)}$, while the third term in (16) assumes the form

$$\sum_i \sum_{s_{i-1}} \sum_{s_i} \sum_{v \in \mathbf{V}} \Big(\sum_{j \in J(v, i)} b_i(s_{i-1}, s_i, j) \ \ln p_i(v_i^j/s_{i-1}, s_i) \Big). \qquad (19)$$

By virtue of Assertion 3, the maximum of expression (19) is reached under the condition

$$P_i(v_i/s_{i-1}, s_i) = \frac{\sum_{j \in J(v, i)} b_i(s_{i-1}, s_i, j)}{\sum_{j \in J} b_i(s_{i-1}, s_i, j)}. \qquad (20)$$

It is obvious from this that, for the realization of the second step of the algorithm for grammar-synthesis, it is required to specify, not function $\alpha(s_0, s_1, \ldots, s_n, j)$, but only function $b_0(S_0, j)$ and functions $b_i(s_{i-1}, s_i, j)$ $(i = 1, 2, \ldots, n)$.

We now consider how to change the algorithm of maximizing expression (16) if it is known that the functions $\{P_i\}$, i.e., the probabilities of the substitutions, do not depend on i.

By taking into account that variables s_{i-1}, s_i in expression (16) are random variables, we can denote them by any other symbols. For example, instead of s_i we shall write s^+ and, instead of s_{i-1}, we shall write s^-. In that case, expression (16) assumes the form

$$D(\mathbf{p}, \mathbf{P}) = \sum_i \sum_{s \in S} b_0(s, j) \ln p(s) + \sum_i \sum_j \sum_{s^- \in S} \sum_{s^+ \in S} b_i(s^-, s^+, j) \ln p(s^+/s^-)$$

$$+ \sum_i \sum_{s^- \in S} \sum_{s^+ \in S} \sum_{v \in V} \sum_{j \in J(v,i)} b_i(s^-, s^+, j) \ln p(v_i^j / s^-, s^+).$$

The maximum of this functional is reached when condition (17) and the following two conditions hold:

$$p(s^+/s^-) = \frac{\sum_i \sum_j b_i(s^-, s^+, j)}{\sum_i \sum_j \sum_{s^+ \in S} b_i(s^-, s^+, j)}, \tag{18a}$$

$$p(v/s^-, s^+) = \frac{\sum_i \sum_{j \in J(v,i)} b_i(s^-, s^+, j)}{\sum_i \sum_{j \in J} b_i(s^-, s^+, j)}. \tag{20a}$$

Thus, quite independently of whether functions $\{\mathbf{P}_i\}$ are identical for all i, for the realization of the second step of the algorithm for synthesis it is required to know only the functions $b_0(s_0, j)$ and $b_i(s^-, s^+, j)$ $(i = 1, 2, \ldots, n)$.

Analysis of First Step of the Algorithm

The aim of the first step is the computation of the quantities $b_i(s^-, s^+, j)$ for $i = 1, 2, \ldots, n$ (n is the word length), $j = 1, 2, \ldots, l$ (l is the size of the teaching sample), $s^- \in \mathbf{S}$, $s^+ \in \mathbf{S}$. The algorithm for computing these quantities does not depend on whether or not the functions $\{\mathbf{P}_i\}$ are identical for all i.

With expressions (12) and (15) taken into account, one can write the expression

$$b_i(s_{i-1}, s_i, j) = \frac{\sum_{s_0} \sum_{s_1} \sum_{s_2} \cdots \sum_{s_{i-2}} \sum_{s_{i+1}} \sum_{s_{i+2}} \cdots \sum_{s_n} p(s_0) p_{i-1}(s_{i-1}, v_{i-1}^j / s_{i-2}) p_i(s_i, v_i^j / s_{i-1}) \prod_{\substack{u=1 \\ u \neq i-1, u \neq i}}^{n} p_u(s_u, v_u^j / s_{u-1})}{\sum_{s_0} \sum_{s_1} \sum_{s_2} \cdots \sum_{s_n} p(s_0) \prod_{u=1}^{n} p_u(s_u, v_u^j / s_{u-1})}. \tag{21}$$

It is not realistic to compute the quantities $b_i(s_{i-1}, s_i, j)$ immediately by formula (21). In what follows we propose an algorithm for computing these quantities, equivalent to expression (21) but, in addition, using an acceptable and necessary outlay of time and memory. The quantities $b_i(s^-, s^+, j)$ as is obvious from formula (21), are a posteriori probabilistic events $(s_{i-1} = s^-) \wedge (s_i = s^+)$ with the conditional events $V^j = v_1^j, v_2^j, \ldots, v_n^j$:

$$b_i(s^-, s^+, j) = p(s_{i-1} = s^-, s_i = s^+ / v_1^j, v_2^j \ldots, v_n^j). \tag{22}$$

The algorithm for computing the quantities of (22) amounts to a preliminary calculation of the solvabilities $p(s_{i-1} = s^- / s_i = s^+, V^j)$ for $i = 1, 2, \ldots, n$; $s^-, s^+ \in \mathbf{S}$, and of probability $p(s_n = s^+ / V^j)$. Starting from the rules for generating words V^j, there follows the validity of the equations

$$p(s_{i-1} = s^- / s_i = s^+, v_1^j, v_2^j, \ldots, v_n^j) = p(s_{i-1} = s^- / s_i = s^+, v_1^j, v_2^j, \ldots, v_i^j). \tag{23}$$

After the probabilities in (23) are found, we then compute the probabilities we seek by the recursion formulas

$$p(s_{i-1} = s^-, s_i = s^+ / V^j) = p(s_i = s^+ / V^j) p(s_{i-1} = s^- / s_i = s^+, v_1^j, v_2^j, \ldots, v_i^j), \tag{24}$$

$$p(s_{i-1} = s^+ / V^j) = \sum_{\tilde{s} = S} p(s_{i-1} = s^+, s_i = \tilde{s} / V^j). \tag{25}$$

Formulas (24) and (25) are computed successively starting with $i = n$ and, thereafter, for $i = n - 1$, etc., down to $i = 1$.

The probabilities of (23) are computed by recursion relationships which are successively used, first for $i = 1$, then for $i = 2$, etc., and, finally, for $i = n$.

Let these recursion relationships have been used i times, and let there have been obtained the probabilities $p(s_{i^*-1} = s^-/s_{i^*} = s^+, v_1^j, v_2^j, \ldots, v_{i^*}^j)$ and $p(s_{i^*} = s^+/v_1^j, v_2^j, \ldots, v_{i^*}^j)$ for all $i^* \le i$.

The probabilities $p(s_i = s^-/s_{i+1} = s^+, v_1^j, v_2^j, \ldots, v_{i+1}^j)$ and $p(s_{i+1} = s^+/v_1^j, v_2^j, \ldots, v_{i+1}^j)$ are computed by the following formulas:

$$p(s_i = s^-/s_{i+1} = s^+, v_1^j, v_2^j, \ldots, v_{i+1}^j) = Cp(s_i = s^-/v_1^j, v_2^j, \ldots, v_i^j)\, p(s_{i+1} = s^+, v_{i+1}^j/s_i = s^-),$$

where C is a normalizing coefficient and

$$p(s_{i+1} = s^+/v_1^j, v_2^j, \ldots, v_{i+1}^j) = \frac{\sum_{s^- \in S} p(s_i = s^-/v_1^j, v_2^j, \ldots, v_i^j)\, p(s_{i+1} = s^+, v_{i+1}^j/s_i = s^-)}{\sum_{s^+ \in S} \sum_{s^- \in S} p(s_i = s^-/v_1^j, v_2^j, \ldots, v_i^j)\, p(s_{i+1} = s^+, v_{i+1}^j/s_i = s^-)}$$

The algorithm described here was experimentally verified. Results of the experimental test are contained in [2].

LITERATURE CITED

1. M. I. Shlezinger, "Interconnection of teaching and self-teaching in pattern recognition," Kibernetika, No. 2 (1968).
2. Methods of Optimal Synthesis of Reading Automata and Algorithms for Correction of Connected Tests. Report [in Russian], Izd. Inst. Kibern. Akad. Nauk UkrSSR, Kiev (1974) (Deposited at VINITI, No. B 360912).
3. K. S. Fu and T. Huang, "Stochastic grammars and languages," Int. J. Comput. Inf. Sci., 1, No. 2 (1972).

METHOD FOR OPTIMAL CLOTH CUTTING

V. A. Trubin and I. N. Grad

UDC 007:518.9

At the present time a method for calculating cloth layout, in which only small remnants remain after layout, is used in clothing factories. The problem of efficient cloth-cutting calculation is solved by linear programming and the method of directed search, on which are based the special-purpose computers EMRT-2 and "Kashtan" [1]. However the existing methods use only data about the length of the piece and do not take into account information about width variations, which may attain large values – up to 5-6 cm. In addition, only one width, usually the minimum, cloth in the piece is taken into account which leads to substantial loss of fabric. On the other hand, taking into account the varying width of the cloth can assure more efficient utilization of materials in the clothing industry by minimizing costs as a result of increasing the useful surface in the wider sections of material.

The task of optimal calculation of cutting cloth pieces of varying width can be formalized in the following way. A set of points $I = \{0, 1, \ldots, M\}$ is given, where the zero point is the start and the M-th the end of the fabric. To each point i ($i = \overline{1, M}$) there correspond two numbers: r_i, the longitudinal coordinate of the i-th point, and h_i, the constant width of fabric in the segment $[r_{i-1}, r_i]$, $r_0 = 0$; $r_M = L$, i.e., the length of the piece. It is assumed that the fabric is symmetrical about the longitudinal axis. Thus, the cloth is approximated as shown in Fig. 1. There exists a set of standard layouts (rectangular patterns) with indices $J = \{1, 2, \ldots, N\}$. An arbitrary layout j ($j = \overline{1, N}$) is defined by its length l_j and width d_j.

The task of optimal calculation consists in dividing a piece of cloth into a certain set of layouts minimizing a given loss function. The following requirements must be satisfied.

1. If the cut fabric is intended for the j-th layout, then its minimal width h must not be less than the width d_j, and the length must be not less than the length l_j.

Translated from Kibernetika, No. 4, pp. 121-124, July-August, 1977. Original article submitted January 6, 1976.