TABLE III
ADAPTATION WITH 270 WORDS

Adaptation Data	Phoneme	Singleton
270 words	60.2%	78.4%

TABLE IV
FIRST-PASS INCLUSION RATES

	Speaker			
Models	Prototype	*M1*	*M2*	*F*
Full training	98.0%	98.0%	88.0%	93.2%
Adaptation		92.4%	84.5%	88.1%
Cross-speaker		39.6%	37.7%	5.7%

TABLE V
WORD RECOGNITION RATES (TRIGRAM LANGUAGE MODEL)

Models	Prototype	*M1*	*M2*	*F*
Full training	92.0%	91.8%	73.1%	84.1%
Means + var		82.9%	67.5%	77.3%
Means only		76.5%	60.1%	59.1%

mance of VQ HMM's trained with large amounts of data can be improved if they are smoothed using an adaptation algorithm [12].

Smoothing is necessary, however, in estimating the covariance matrices since the Baum–Welch estimates may turn out to be singular if there are not sufficient data. The method we propose for adapting the covariance matrices clearly gives better results than merely copying them from the prototype speaker, but it is not rigorous, and other approaches are possible; it would be interesting to know more about this question.

REFERENCES

[1] S. E. Levinson, L. R. Rabiner, and M. M. Sondhi, "An introduction to the application of the theory of probabilistic functions of a Markov process to automatic speech recognition," *Bell Syst. Tech. J.*, vol. 62, pp. 1035–1074, 1983.

[2] V. N. Gupta, M. Lennig, and P. Mermelstein, "Integration of acoustic information in a large vocabulary word recognizer," in *Proc. ICASSP*, 1987, pp. 697–700.

[3] L. Deng, M. Lennig, F. Seitz, V. Gupta, P. Kenny, and P. Mermelstein, "Large vocabulary word recognition using context-dependent allophonic hidden Markov models," submitted to *Comput. Speech Language*.

[4] S. B. Davis and P. Mermelstein, "Comparison of parametric representations for monosyllabic word recognition in continuously spoken sentences," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. ASSP-28, pp. 357–365, 1980.

[5] K. Sugawara, M. Nishimura, and A. Kuroda, "Speaker adaptation for a hidden Markov model," in *Proc. ICASSP*, 1986, pp. 2667–2670.

[6] A. Jarre and R. Pieraccini, "Some experiments on HMM speaker adaptation," in *Proc. ICASSP*, 1987, pp. 1273–1276.

[7] R. Schwartz, Y.-L. Chow, and F. Kubala, "Rapid speaker adaptation using a probablistic spectral mapping," in *Proc. ICASSP*, 1987, pp. 633–636.

[8] L. R. Bahl, R. L. Mercer, and D. Nahamoo, "An algorithm for estimating the parameters of hidden Markov models from a short training script," presented at the IEEE Workshop on Speech Recognition, 1988.

[9] L. E. Baum, "An inequality and associated maximization technique in statistical estimation for probabilistic functions of Markov processes," *Inequalities*, vol. 3, pp. 1–8, 1972.

[10] E. A. Martin, R. P. Lippmann, and D. B. Paul, "Two-stage discriminant analysis for improved isolated-word recognition," in *Proc. ICASSP*, 1987, pp. 709–713.

[11] V. N. Gupta, M. Lennig, and P. Mermelstein, "Fast search strategy in a large vocabulary word recognizer," submitted to *J. Acoust. Soc. Amer.*

[12] R. Schwartz, F. Kubala, O. Kimball, P. Price, and J. Makhoul, "Improving performance of phonetic hidden Markov models in a continuous speech recognition system," presented at the IEEE Workshop on Speech Recognition, 1988.

Inference of *k*-Testable Languages in the Strict Sense and Application to Syntactic Pattern Recognition

PEDRO GARCÍA AND ENRIQUE VIDAL

***Abstract*—The inductive inference of any language from the (general) class of regular languages, from only positive samples of this language, is well known to be an undecidable problem. However, it has fortunately been shown that this is not the case for certain particular subclasses of languages. This correspondence is concerned with the inductive inference of one of these classes, namely, the class of *k*-testable languages in the strict sense (*k*-TSSL). A *k*-TSSL is essentially defined by a finite set of substrings of length *k* that are permitted to appear in the strings of the language. Given a positive sample *R* of strings of an unknown language, we obtain a deterministic finite-state automaton that recognizes the smallest *k*-TSSL containing *R*. The inferred automaton is shown to have a number of transitions bounded by $O(m)$ where *m* is the number of substrings defining this *k*-TSSL, and the inference algorithm works in $O(kn \log m)$ where *n* is the sum of the lengths of all the strings in *R*. The proposed methods are illustrated through syntactic pattern recognition experiments in which a number of strings generated by ten given (source) non-*k*-TSSL grammars are used to infer ten *k*-TSSL stochastic automata, which are further used to classify new strings generated by the same source grammars. The results of these experiments are consistent with the theory, showing as well the ability of (stochastic) *k*-TSSL's to approach other classes of regular languages.**

***Index Terms*—Learning, locally testable languages, regular grammar inference, syntactic pattern recognition.**

I. INTRODUCTION

Grammatical inference (GI) is, in syntactic pattern recognition (SPR), the *learning* or *model estimation* phase which is essential to any proper approach to pattern recognition (PR). The scope of grammatical inference (GI) is, however, strongly limited by two fundamental results due to Gold. The first one establishes the undecidability of the problem of identifying any language in the limit, using only *positive samples* from the language [14], even from the simple class of regular languages. The second result, on the other hand, seems to computationally limit the possibility of using *negative samples* for the inference of regular languages by establishing

Manuscript received March 7, 1988; revised December 28, 1989. This work was supported in part by the Spanish CAICYT under Grant PA85-86.

The authors are with the Departamento de Sistemas Informáticos y Computación, Universidad Politécnica de Valencia, 46071 Valencia, Spain.

IEEE Log Number 9036254.

that the problem of finding a minimum deterministic finite-state automaton, which accepts all the strings in a finite positive sample and rejects all the strings in another finite negative sample, is NP hard [15], [3].

From the point of view of SPR, the problem of inferring automata which represent classes of objects from given instances of these objects is, however, of paramount importance since this branch of PR should not be considered fully mature until the phase of learning the models (automata) of the classes is well understood and practical methods for carrying out this task are available. Given the above results, one should therefore pessimistically consider the possibility of ever achieving such maturity. In spite of this, the problem of GI for SPR has, in fact, been tackled by a considerable number of researchers [8], [10], [16], [22], [20], [17], [11], [33], [26], [27], [25], [19], many of whose papers seem to ignore the fundamental limitations stated above. It is worth noting that all the GI procedures proposed in these papers: 1) only make use of positive samples, and 2) do not explicitly aim at identifying any language from any given class. Rather, these methods supply target languages which represent just *heuristic* generalizations of the positive samples that are used and which, hopefully, produce acceptable practical results for the PR problems that are considered. Clearly, this does not seem to be a very appealing framework, and one ought to move towards more formal and/or general settings.

Fortunately enough, the results of Gold, although very severe even with respect to the class of regular languages, say nothing about the tractability of GI for some proper subsets of this class. Consequently, one possible approach to the problem is to restrict the inference to the appropriate subclasses of languages. Following Angluin [6], the methods falling into this approach can be further dichotomized into *heuristic* and *characterizable*, the latter differing from the former in that the corresponding subclasses of inferred languages are well-known classes and/or they are "well characterized." Most of the GI procedures referred to above can thus be classified as heuristic. On the other hand, although clearly most interesting, characterizable methods are very scarce, and only a few papers seem to have been published in this direction [4], [5], [24]. Finally, a possible tradeoff between heuristic and characterizable methods is proposed in [12], which would enable those *problem-relevant* features of the (possibly unknown) class of languages aimed at being inferred to be specified through appropriate means.

This correspondence is concerned with a new characterizable and efficient GI method. It is characterizable because the inferred classes are the well-known classes of k-testable languages in the strict sense (k-TLSS) [35]. Informally speaking, a k-TSSL is defined by a finite set of substrings of length k that are allowed to appear in the strings of the language. Concepts which are more or less related to k-TLSS's have been widely used in information theory and also in practical PR. k-testable stochastic languages in the strict sense are directly related to order-k Markov sources (see, e.g., [1]), and the frequencies (probabilities) of occurrence of substrings of increasing lengths have been utilized as successive approximations to characterize natural languages [29], [30]. On the practical side, these concepts have led to quite useful computer programs for spelling correction, like the famous TYPO on UNIX (see, e.g., [23]), and also to successful approaches to speech recognition [7] and phoneme-to-text (stenotype) transcription [9]. On the other hand, the concept of an "N-gram" (which also comes from the terminology of Markov sources) has been successfully utilized in other practical PR systems, many of which are also related to speech and/or waveform recognition [18], [26], [31], [32].

II. Locally Testable Sets

Let $Z_k = (\Sigma, I_k, F_k, T_k)$ be a four-tuple where Σ is a *finite alphabet*, $I_k, F_k \subseteq \bigcup_{i=1}^{k-1} \Sigma^i$ are two sets of *initial* and *final segments*, respectively, and $T \subseteq \Sigma^k$ is a set of *forbidden segments* of *length* k. A k-testable language in the strict sense (k-TLSS) is defined by the regular expression

$$l(Z_k) = (I_k\Sigma^*) \cap (\Sigma^*F_k) - (\Sigma^*T_k\Sigma^*). \tag{2.2}$$

The strings in $l(Z_k)$ can therefore by characterized as follows: they start with segments in I_k, they end with segments in F_k, and they do *not* have any segments of length k which is in T_k. An interesting subclass of k-TLSS is the class of 2-TLSS's, which are also referred to as *local languages* [28], [12]. On the other hand, the class of all k-TLSS's for any k is referred to as the class of *locally testable languages in the strict sense* (LTLSS). The above definitions of k-TLSS and LTLSS are quite similar to those of [21] and [35], although they are conveniently adapted to include local languages as a natural $k = 2$ case.

III. Smallest k-TLSS Containing a Positive Sample

Let R be a learning set (positive sample) and $k \geq 1$. We can uniquely associate a four-tuple $Z_k(R) = (\Sigma(R), I_k(R), F_k(R), T_k(R))$ with R as follows.

$$I_k(R) = \{u \mid uv \in R, |u| = k-1, v \in \Sigma(R)^*\} \cup \{x \in R \mid |x| < k-1\}$$
(initial segments of length at most $k-1$ of the strings in R).

$$F_k(R) = \{v \mid uv \in R, |v| = k-1, u \in \Sigma(R)^*\} \cup \{x \in R \mid |x| \leq k-1\}$$
(final segments of length at most $k-1$ of the strings in R).

$$T_k(R) = \Sigma(R)^k - \{v \mid uvw \in R, |v| = k, u, w \in \Sigma(R)^*\}$$
(segments of length k not appearing in the strings in R). (3.1)

In the sequel, a k-TLSS $l(Z_k(R))$ will be denoted as $l_k(R)$. The following results establish some important relations between R and $l_k(R)$ [13].

Lemma 3.1: $R \subseteq l_k(R)\ \forall k \geq 1$.
Theorem 3.1: $l_k(R)$ is the smallest k-TLSS that contains R.
Thoerem 3.2; Let $R' \subset R$. Then $l_k(R') \subseteq l_k(R)$.
Theorem 3.3: Let $k \geq 1$. Then $l_{k+1}(R) \subseteq l_k(R)$.
Corollary 3.1: Let $k > \max_{x \in R}|x|$. Then $l_k(R) = R$.

IV. Inference Algorithm

Based on the above construction, we propose the k-TSSI algorithm (shown in Fig. 1) for the inference of k-TLSS's. The correctness of this algorithm is established by the following theorem [13].

Theorem 4.1: Let R be a positive sample, and let A_k ($k \geq 2$) be the automaton obtained from R by the k-TSSI algorithm. Then $l_k(R) = L(A_k)$.

From this theorem and (3.1), we see that the automaton A_k inferred from R for a given value of $k \geq 2$ accepts the smallest k-TLSS containing R. Also, using Theorem 3.3 and Corollary 3.1, we can see that, for a given sample R, increasing values of k produce increasingly restricted languages. Therefore, the proposed GI algorithm permits a variety of solutions to a given inference problem to be obtained by changing the value of k from 2 to the length of the longest string in R. These solutions supply languages which span from the smallest local language (2-TLSS) containing R to exactly R (Fig. 2).

The following examples illustrate the proposed method.

Example 1: Let $R = \{aa, aba, abba, abbba\}$ and let $k = 2$. Then $Z_2(R) = (\{a, b\}, \{a\}, \{a\}, \phi)$. Also, if $k = 3$, then $Z_3(R)$

```
k-TSSI algorithm // Obtains a DFA which accepts the smallest k-TLSS containing R //
Input: k≥2, R: set of training strings.
     // the case k=1 is trivial since ∀R Z1(R)=(Σ(R),{e},{e},φ) and l1(R)=Σ(R)* //
Output: DFA Ak= (Q,Σ,δ,q0,Qf)      // Q ⊆ ∪_{i=0}^{k-1} Σ^i,  q0∈Q,  Qf ⊆ Q,  δ ⊆ (Q×Σ×Q) //
Method
   (Σ,I,F,T):= (Σ(R), Ik(R), Fk(R), Tk(R));
   Q:= {e}; δ:= φ; q0:= e;                    // e is the symbol for the nil string //
   ∀ a1...am ∈I for j:=1 to m
      Q:= Q ∪ {a1...aj};
      δ:= δ ∪ {(a1...aj-1,aj,a1...aj)};                    //ai...aj = e iff j<i//
   end for end ∀
   ∀ a1...ak ∈(Σ^k-T)
      Q:= Q ∪ {a2...ak};
      δ:= δ ∪ {(a1...ak-1,ak,a2...ak)};
   end ∀
   Qf:= F; Ak:= (Q,Σ,δ,q0,Qf);
end k-TSSI
```

Fig. 1. Inference algorithm.

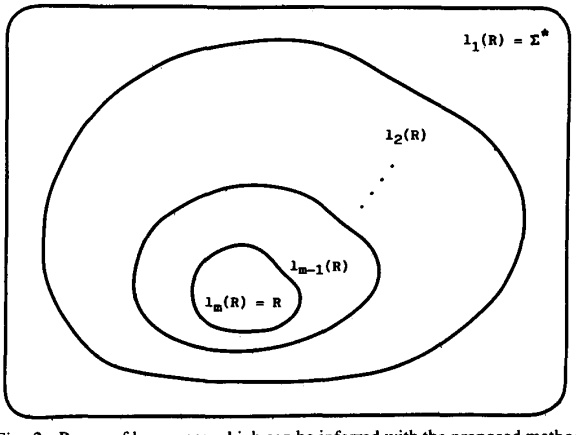

Fig. 2. Range of languages which can be inferred with the proposed method from a given positive sample R ($m = \max_{x \in R} |x|$).

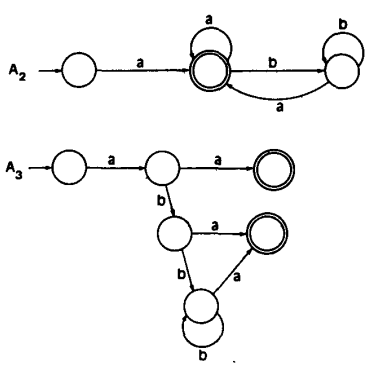

Fig. 3. Inferred automata for $k = 2$ and $k = 3$. The inferred languages are $a(b^*a)^*$ and ab^*a, respectively. Although not (always) minimal, the inferred automata are reasonably small—in fact, they are bounded as shown in Section VI.

$= (\{a, b\}, \{aa, ab\}, \{aa, ba\}, \{a, b\}^3 - \{aba, abb, bba, bbb\})$. From $Z_2(R)$, $Z_3(R)$, and following the k-TSSI algorithm, we infer the automata A_2 and A_3 shown in Fig. 3.

V. Identification in the Limit of Locally Testable Languages in the Strict Sense

A characterization of the classes of languages that are identifiable from only positive samples in the limit is given in [4]. A sufficient condition is established by the following theorem.

Theorem 5.1 (Angluin, 1980); Let $L_1, L_2, \cdots$ be an indexed class of recursive languages such that, for every nonempty finite set $R \subset \Sigma^*$, the cardinality of the set $C(R) = \{L: R \subseteq L \text{ and } L = L_i \text{ for some } i\}$ is finite. Then this class is identifiable in the limit from only positive samples.

Given that the number of different four-tuples (Σ, I, F, T) such that $\Sigma = \Sigma(R)$, $I_k(R) \subseteq I$, $F_k(R) \subseteq F$, $T_k(R) \supseteq T$ is finite, the set of different k-TLSS's containing R is finite. From Theorem 5.1, this implies that the class of k-TLSS's is identifiable using only positive samples.

Following this result and Theorems 4.1 and 3.1, it can be proved that the proposed GI method is characterizable.

Theorem 5.2: The k-TSSI algorithm identifies any k-TLSS in the limit from positive data.

Note that, despite this result, the class of locally testable languages in the strict sense (LTLSS) (k-TLSS for any k), as a whole, remains unidentifiable in the limit from only positive presentation sequences. However, the proposed inference algorithm can be effectively used to identify any language from this class in the limit through a *complete* (both positive and negative) presentation sequence of the language [14]. From Theorems 3.3, 4.1, and 5.2, this can be accomplished by starting with $k = 2$ and using successive positive samples to infer progressively larger (less restricted) 2-TLSS's until a negative sample, which is incompatible with the current language, appears. Then k is increased by one, and the process continues in the same way with the successive samples. Eventually, the correct value of k will be reached and then, following Theorems 3.1 and 3.2, no other negative sample will ever be incompatible. The inferred language will then grow progressively with the successive positive samples until the source k-TLSS is exactly identified, thus effectively stopping the changes of the output automaton, which is precisely the condition assessing the identification in the limit [14].

VI. Size of the Inferred Automata and Complexity of the Inference Algorithm

Let $Z_k = (\Sigma, I_k, F_k, T_k)$ be the four-tuple which defines a k-TLSS from which R has been drawn, and let $T' = \Sigma^k - T_k$. It follows from the k-TSSI algorithm that the maximum total number of transitions of A_k is $|\delta| = |I_0| + |T'|$ where $I_0 = \{u \in \Sigma^*: uv \in I_k, v \in \Sigma^*\}$. Therefore, since for nontrivial k-TLSS's $|I_0| \leq |T'|$, and since for every finite automaton $|Q| \leq |\delta|$, we can write

$$|\delta| = O(|T'|); \quad |Q| = O(|T'|); \quad B = O(|\Sigma|) \tag{6.1}$$

where B is the maximum number of transitions associated with any state of A_k ("branching factor"). These bounds are given in terms of the complexity of the language which is being inferred. This complexity can, in turn, be bounded for given k and Σ as $|T'| \leq |\Sigma|^k$, yielding

$$|\delta| = O(|\Sigma|^k); \quad |Q| = \left|\bigcup_{i=1}^{k-1} \Sigma^i\right| + 1 = (|\Sigma|^k - 1)/(|\Sigma| - 1)$$
$$= O(|\Sigma|^{k-1}); \quad B = O(|\Sigma|). \tag{6.2}$$

In practice, however, the source language is not often (well) known, and one would prefer the growing rate of the inferred automaton to be given as a function of only the size of the given positive sample. In this case, following (3.1), one can readily verify that if $Z_k(R) = (\Sigma(R), I_k(R), F_k(R), T_k(R))$ is the four-tuple associated with R, then $|\Sigma^k(R) - T_k(R)| \leq n = \Sigma_{x \in R} |x|$, and from (6.1), we have

$$|\delta| = O(n); \quad |Q| = O(n); \quad B = O(|\Sigma(R)|). \tag{6.3}$$

It should be noted, however, that if the source language is really a k-TLSS, the bounds (6.3) [and also (6.2)] can become rather pessimistic. This is because, as n gets larger, all the elements of Σ, I_k, F_k, and T' will eventually have already appeared in the strings

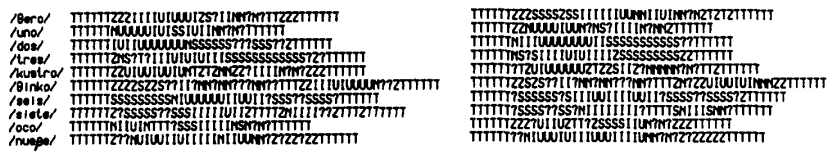

Fig. 4. A sample of the strings involved in the experiments. The symbols represent: \I\ = front vowel, \U\ = back vowel, \N\ = weak sonorant, \S\ = strong fricative, \Z\ = weak fricative, \T\ = stop or silence. \?\ = unreliable classification.

of R, and then the inferred automaton will, in fact, stop growing, while the above bounds will not.

The time and space complexities of the inference procedure defined by (3.1) and the k-TSSI algorithm are established by the following theorem.

Theorem 6.1: Let $Z_k = (\Sigma, I_k, F_k, T_k)$ be a four-tuple defining a k-testable language $l(Z_k)$, let $R \subset l(Z_k)$ be a positive sample, and let $Z_k(R)$ be the four-tuple associated with R. An automaton A_k such that $L(A_k) = l(Z_k(R))$ can be inferred in $O(kn \log m)$ time, and can be represented using $O(m|\Sigma|)$ space where $n = \Sigma_{x \in R}|x|$ and $m = |\Sigma^k - T_k|$.

These bounds come from the fact that, by using the appropriate linear data structures to represent the different sets involved in the construction of $Z_k(R)$ and A_k, the required *set find-insert operations* can be carried out in at most $O(k \log m)$ time [2], [13].

Several facts should be pointed out concerning the above bounds. First, if the source language is not known, but it is known to be a k-TLSS over the alphabet Σ, one may realize that $m \geq |\Sigma|^k$, which leads to an inference time bound in $O(k^2 n \log |\Sigma|)$. On the other hand, if nothing is known about the source language, one may see that $|\Sigma^k - T_k(R)| \leq n$ to obtain an inference time bound in $O(kn \log n)$. In this case, however, the same remarks that have been made above about the bounds (6.3) apply.

The class of k-TLSS's is shown in [13] to be included in the class of k-reversible languages [5]; consequently, the methods proposed in [5] could be seen as applicable for the inference of k-TLSS's. However, the time complexity of Angluin's inference algorithm is $O(n^3)$, and then, if the languages of interest can be assumed to belong to (or to be conveniently approached by members of) the class of LTLSS's, the methods proposed here can be chosen with significant advantage over the (most general) methods available for reversible languages.

VII. Stochastic Extension

The proposed inference method can be straightforwardly modified to deal with direct inference of grammars rather than automata [13]. Therefore, we can alternatively use automata or grammars at convenience. For a stochastic extension, however, the latter seems more adequate, given that probability estimation can be more properly formulated with grammars rather than with automata [16], [11].

Given that the inferred automata or grammars are *unambiguous*, a maximum likelihood estimate of the probabilities of the grammar productions can be straightforwardly obtained from their relative frequency of use for the parsing of the strings in R [36], [37]. Furthermore, this estimation can be easily performed incrementally and simultaneously with the inference procedure (see, e.g., [16]).

VIII. Empirical Tests

The GI method proposed in the previous sections has been implemented and applied to a pattern recognition task. The considered task is that of recognizing the class membership of strings which are generated by ten source regular grammars, each representing a class. The classes correspond to the ten (spoken) Spanish digits, and the strings are sequences of "microphonetic" broad-phonetic labels. Details of such a representation can be found in [34] and [27]. Fig. 4 shows a sample of the strings that are used in the experiments. The source grammars used to synthesize the strings are exactly the same as those used in [27] to recognize *natural* spoken Spanish digits (i.e., strings directly obtained from real speech signals). The reason for this way of obtaining the strings, instead of directly using natural samples, is twofold. First, this allows for large amounts of training samples to be obtained without effort; such large amounts are required to empirically investigate the asymptotical behavior of the proposed methods. Second, in this way, the regularity of the considered samples is absolutely guaranteed since they are all, in fact, generated by regular grammars; this prevents the results from being affected by the possible inadequacy of choosing regular languages as models of actually more complex natural sources (speech). Note that, although the grammars used to generate the samples are of a certain particular type [27], they are by no means guaranteed to generate k-TLSS's, at least for the small values of k which are used in the experiments. Therefore, the results of these experiments can also be considered as a measure of the ability of stochastic k-TLSS's to approach other classes of regular languages.

For each of the ten classes, the corresponding (nonstochastic) *source grammar* was utilized to randomly generate 200 strings. From these 2000 strings, 500 (50 from each class) were set aside for testing (test set), and the rest (150 from each class) were considered as positive samples R^i, $i = 1, \cdots, 10$ (training set). Each R^i was, in turn, divided into 30 disjointed subsets of five strings each, and the grammatical inference process was carried out in successive steps, each considering an increasing number of such subsets. At the end of each step, the 500 strings in the test set were submitted for recognition to each of the ten currently inferred grammars G^i, $i = 1, \cdots, 10$. The inference was performed with a stochastic version of k-TSSI as outlined in Section VII. Recognition was carried out with a very simple deterministic procedure. However, in order to allow for the parsing of strings not exactly in $L(G^i)$, a type of error-correcting trick was applied. It consisted of allowing the use (with a very low probability, $\epsilon = 0.001$) of an adequate "wild chart production" whenever the current input symbol cannot be parsed with the current production of G^i. This technique was found to reduce the rejection rate to zero, while leaving unchanged the confusion rate for those test strings which actually were in the language of some inferred grammar.

Fig. 5 shows the overall recognition error rate which was obtained for four values of k as a function of the size of the training set. It is worth noting how the results improve dramatically from $k = 2$ to $k = 3$. For other values of k (3, 4, and 5), good results (error $< 2\%$) are obtained for training-set sizes of over 40 strings, whereas for sizes greater than 100, recognition rates greater then 99.5% are obtained, reaching 100% for 150 samples and $k = 4$.

The complexity of the inferred grammars (or the corresponding automata) is also shown in Fig. 5 in terms of the average (over the ten classes) number of states inferred for each value of k as a function of the size of the training set. Note that this complexity is very much smaller than the bound (6.3) ($O(n)$, $n \approx 40\,|R|$ for the strings involved). In fact, the number of states appears to tend asymptotically to a constant. For the higher values of k, however, this constant is significantly smaller than the absolute bound (6.2) for the corresponding value of k. This is explicitly presented in Table I where the above asymptotic values are presented for several

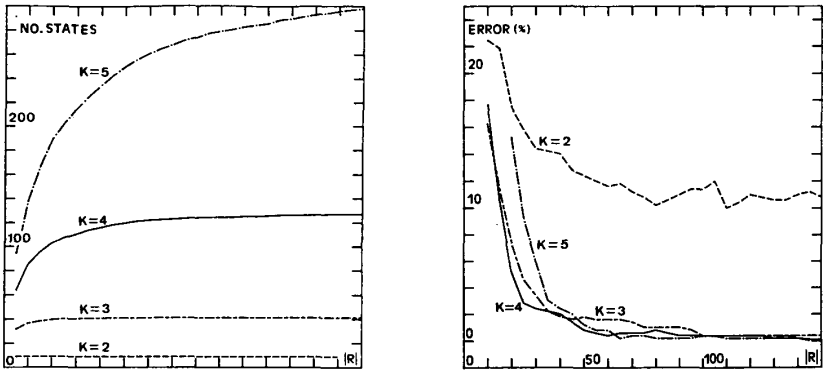

Fig. 5. Overall recognition error (right) and average complexity of the inferred automata (left) as a function of the size of the training set.

TABLE I
ASYMPTOTIC AVERAGE COMPLEXITY AND PERFORMANCE OF THE INFERRED AUTOMATA
(T = THEORETICAL BOUND; E = VALUE FOUND EMPIRICALLY)

	k=2		k=3		k=4		k=5		Source
	T	E	T	E	T	E	T	E	
No.states	9	9	58	42	401	127	2802	298	105
Branch.F.	7	4.5	7	3	7	2.3	7	1.9	2.2
Error %	10.8		0.4		0		0.2		0

values of k. In the same table, the corresponding values of the average branching factor are presented, which are also well below the theoretical bounds. It is worth noting that the average complexity of the original source grammers (or the corresponding equivalent automata) was 105 states and the branching factor was about 2.2 [27], while for slightly higher values (127 states and a branching factor of 2.3), perfect recognition is achieved with the inferred automata. Furthermore, with significantly lesser complexity (42 states and a branching factor of 3), quite useful approximations (recognition rate = 99.6%) can be achieved.

ACKNOWLEDGMENT

The authors gratefully acknowledge the thoughtful suggestions and corrections of one anonymous referee.

REFERENCES

[1] N. Abramson, *Information Theory and Coding*. New York: McGraw-Hill, 1966.
[2] A. V. Aho, J. Hopcroft, and J. D. Ullman, *The Design and Analysis of Computer Algorithms*. Reading, MA: Addison-Wesley, 1974.
[3] D. Angluin, "On the complexity of minimum inference of regular sets," *Inform. Contr.*, vol. 39, pp. 337-350, 1978.
[4] —, "Inductive inference of formal languages from positive data," *Inform. Contr.*, vol. 45, pp. 117-135, 1980.
[5] —, "Inference of reversible languages," *J. Ass. Comput. Mach.*, vol. 29, no. 3, pp. 741-765, 1982.
[6] D. Angluin and C. H. Smith, "Inductive inference: Theory and methods," *Comput. Surveys*, vol. 15, no. 3, pp. 237-269, 1983.
[7] L. R. Bahl, F. Jelinek, and L. R. Mercer, "A maximum likelihood approach to continuous speech recognition," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. PAMI-5, pp. 179-190, Mar. 1983.
[8] A. W. Biermann and J. A. Feldman, "On the synthesis of finite-state machines from samples of their behavior," *IEEE Trans. Comput.*, vol. C-21, pp. 592-597, 1972.
[9] A. M. Derouault and B. Merialdo, "Natural language modeling for phoneme to text transcription," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. PAMI-8, pp. 742-749, 1986.
[10] K. S. Fu and T. L. Booth, "Grammatical inference: Introduction and survey, Parts 1 and 2," *IEEE Trans. Syst., Man, Cybern*, vol. SMC-5, pp. 95-111, 409-423, 1975.
[11] K. S. Fu, *Syntactic Pattern Recognition and Applications*. Englewood Cliffs, NJ: Prentice-Hall, 1982.
[12] P. García, E. Vidal, and F. Casacuberta, "Local languages, The successor method, and a step towards a general methodology for the inference of regular grammars," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. PAMI-9, pp. 841-845, Nov. 1987.
[13] P. García, "Explorabilidad local en inferencia inductiva de lenguajes regulares y aplicaciones," Doctoral dissertation, Departamento de Sistemas Informatícos y Computacion, Universidad Politécnica de Valencia, Valencia, Spain, 1988.
[14] E. M. Gold, "Language identification in the limit," *Inform. Contr.*, vol. 10, pp. 447-474, 1967.
[15] —, "Complexity of automaton identification from given data," *Inform. Contr.*, vol. 37, pp. 302-320, 1978.
[16] R. C. Gonzalez and M. G. Thomason, *Syntactic Pattern Recognition, An Introduction*. Reading, MA: Addison-Wesley, 1978.
[17] S. Y. Itoga, "A new heuristic for inferring regular grammars," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. PAMI-3, pp. 191-197, Mar. 1981.
[18] O. Venta, "A very fast sentence reconstruction method for the post-processing of computer-recognized continuous speech," in *Proc. IEEE ICPR '84*, 1984, pp. 1240-1243.
[19] M. Kudo and M. Shimbo, "Efficient regular grammatical inference techniques by the use of partial similarities and their logical relationships," *Pattern Recognition*, vol. 21, no. 4, pp. 401-409, 1988.
[20] B. Levine, "Derivatives of tree Sets with applications to grammatical inference," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. PAMI-3, May 1981.
[21] R. McNaughton, "Algebraic decision procedures for local testability," *Math. Syst. Theory*, vol. 8, no. 1, pp. 60-76, 1974.
[22] L. Miclet, "Regular inference with a tail-clustering method," *IEEE Trans. Syst., Man, Cybern.* vol. SMC-10, pp. 737-747, 1980.
[23] J.L. Peterson, *Computer Programs for Spelling Correction*. New York: Springer-Verlag, 1980.

[24] V. Radhakrishnan and G.Nagaraja, "Inference of regular grammars via skeletons," *IEEE Trans. Syst., Man, Cybern.*, vol. SMC-17, no. 6, 1987.
[25] ——, "Inference of even linear grammars and its application to picture description languages," *Pattern Recognition*, vol. 21, no. 1, pp. 55-62, 1988.
[26] M. Richetin and F. Vernadat, "Efficient regular grammatical inference for pattern recognition," *Pattern Recognition*, vol. 17, no. 2, pp. 245-250, 1984.
[27] H. Rulot and E. Vidal, "Modelling (sub)string-length-based constraints through a grammatical inference method," in *Pattern Recognition Theory and Applications*, P. A. Devijver and J. Kittler, Eds. New York: Springer-Verlag, 1987, pp. 451-459.
[28] A. Salomaa, *Jewels of Formal Language Theory*. Rockville, MD: Computer Science Press, 1981.
[29] C. E. Shannon, "A mathematical theory of communication," *Bell Syst. Tech. J.*, vol. 27, pp. 390-397, 1948.
[30] ——, "Prediction and entropy of printed English," *Bell Syst. Tech. J.*, vol. 30, no. 1, pp. 50-64, 1951.
[31] A. R. Smith, J. N. Denemberg, T. B. Slack, C. C. Tan, and R. E. Wohlford, "Application of a sequential pattern learning system to connected speech recognition," in *Proc. IEEE ICASSP '85*, 1985, pp. 31.2.1-31.2.4.
[32] O. Venta and T. Kohonen, "A non-stochastic method for the correction of sentences," in *Proc. IEEE ICPR '86*, 1986, pp. 1214-1217.
[33] F. Vernadat and M. Richetin, "Regular inference for syntactic pattern recognition: A case study," in *Proc. IEEE ICPR '84*, 1984, 1370-1372.
[34] E. Vidal, F. Casacuberta, E. Sanchis, and J. M. Benedi, "A general fuzzy parsing scheme for speech recognition," in *NATO-ASI New Systems and Architectures for Automatic Speech Recognition and Synthesis*, R. De Mori and C. Y. Suen, Eds. New York: Springer-Verlag, 1985, pp. 427-446.
[35] Y. Zalcstein, "Locally testable languages," *J. Comput. Syst. Sci.*, vol. 6, pp. 151-167, 1972.
[36] F. J. Maryanski and T. L. Booth, "Inference of finite state probabilistic grammars," *IEEE Trans. Comput.*, vol. C-26, pp. 531-536, 1977.
[37] R. Chaudhuri and A. N. V. Rao, "Approximating grammar probabilities: Solution of a conjecture," *J. Ass. Comput. Mach.*, vol. 33, no. 4, pp. 702-705, 1986.