LEARNING LOCALLY TESTABLE LANGUAGES IN THE STRICT SENSE(*)

Pedro García, Enrique Vidal and José Oncina

Dpto. Sistemas Informáticos y Computación
Universidad Politécnica de Valencia

Corresponding author address:
Pedro García, Dept. Sist. Informáticos y Computación,
Universidad Politécnica Valencia, 46071 VALENCIA, SPAIN.

ABSTRACT

A Locally Testable Language in the Strict Sense (LTSS) is a language that is strictly k-Testable for some k. A k-Testable Language in the Strict Sense (k-TLSS) is essentially defined by a finite set of substrings of length k that are permitted to appear in the strings of the language. This paper is concerned with the inductive inference of automata that recognize LTSS's from samples of these languages. Given a positive sample R of strings of an unknown language, an algorithm is proposed which obtains a deterministic finite automaton that recognizes the smallest k-TLSS containing R. Moreover this algorithm can be implemented to run in O(knlogm), where n is the sum of the lengths of all the strings in R and m is the number of permitted segments of length k. Also, for a given k, the proposed method is shown to actually identify the source k-TLSS language in the limit, from only a positive presentation of this language; furthermore, if it is only known that the source language is a LTLSS language, then a method is given which identifies this language in the limit, from a complete presentation (both positive and negative samples).

* Work partially supported by the Spanish CICYT under grant TIC-0448/89

Proceedings of the Workshop on Algorithmic Learning Theory

1. INTRODUCTION.

It is known that the class of regular languages remains unidentifiable from positive data in the limit [11]. However, a characterization of the classes that are identifiable from positive presentation was established by Angluin [3]. Moreover, she has proved the learnability of the class of k-Reversible languages and proposed a polynomial time algorithm which identifies such a class from positive data in the limit [4].

This paper deals with the inferability of an important family of formal languages: the class of **Locally Testable Languages** in the **Strict Sense** (LTLSS). The family of Locally Testable languages is a proper sub-family of Star Free languages of dot-depth 1 [5]. Previous works on the inference of the class of LTLSS's can be found in [9] and more recently in [21]. Some applications of learning of k-LTLSS's to Pattern Recognition (PR) problems was developed in [10].

Informally speaking, a **k-Testable Language in the Strict Sense** (k-TLSS) 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 succesive approximations to characterize natural languages [16], [17]. 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. [13], and also to successful approaches to speech recognition [6], and phoneme to text (stenotype) transcription [7]. On the other hand, the concept of "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 with speech and/or waveform recognition, [18], [19], [20].

2. 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 $T_k \subseteq \Sigma^k$ 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

$$l(Z_k) = (I_k\Sigma^*) \cap (\Sigma^*F_k) - (\Sigma^*T_k\Sigma^*) \qquad (2.2)$$

The strings in $l(Z_k)$ can therefore be characterized as follows: they start with segments in I_k, they end with segments in F_k, and they do not have any segment 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 [15], [8]. 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 entire family of k-Testable Languages is defined as the boolean clousure of k-TLSS's. The above definitions of k-TLSS and LTLSS are quite similar to those of [12] and [23], [5] though conveniently adapted to include Local Languages as a natural k=2 case.

3. 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:

$\Sigma(R)$: set of the symbols that appear in the words of R.

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

$F_k(R) = \{v \mid uv\in R, |v|=k-1, u\in\Sigma(R)^*\} \cup \{x\in R \mid |x|<k-1\}$ (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 of 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 $l_k(R)$ [9].

Lemma 3.1. $R \subseteq l_k(R) \quad \forall k \geq 1$.

Theorem 3.1. $l_k(R)$ is the smallest k-TLSS that contains R.

Proof. Let l' be a k-TLSS such that $R \subset l'$. We have to prove that, for any such l', $l_k(R) \subseteq l'$.

For every $x \in \Sigma^*$ and $k \geq 1$, let us define a **k-test vector** $(i_{k-1}(x), f_{k-1}(x), t_k(x))$ as follows:

$$i_{k-1}(x) = \begin{cases} x \textbf{ if } |x|<k \\ u \textbf{ if } x=uv,\ |u|=k-1,\ v \in \Sigma^+ \end{cases}$$

$$f_{k-1}(x) = \begin{cases} x \textbf{ if } |x|<k \\ v \textbf{ if } x=uv,\ |v|=k-1,\ u \in \Sigma^+ \end{cases}$$

$$t_k(x) = \{v \mid x=uvw,\ |v|=k,\ u,v \in \Sigma^*\}$$

By contradiction, let us assume that $l_k(R)-l' \neq \phi$. Then, there exist an $x \in \Sigma^*$ such that $x \in l_k(R)$, and $x \notin l'$:

$i_{k-1}(x) \in I_k(R)$ and $f_{k-1}(x) \in F_k(R)$ and $t_k(x) \subseteq \Sigma^k - T_k(R)$.

If (Σ, I', F', T') is the four-tuple defining l', then:

$i_{k-1}(x) \notin I'$ or $f_{k-1}(x) \notin F'$ or $t_k(x) \not\subseteq \Sigma^k - T'$. But then $\exists y \in R$ such that $y \notin l'$, and therefore $R \not\subset l'$. ■

Theorem 3.2. Let $R' \subset R$. Then $l_k(R') \subseteq l_k(R)$.

Theorem 3.3. Let $k \geq 1$. Then $l_{k+1}(R) \subseteq l_k(R)$ [9].

Corollary 3.1. Let $k > \max_{x \in R} |x|$. Then $l_k(R) = R$.

4. INFERENCE ALGORITHM.

Based on the above construction, we propose the k-TSSI algorithm for the inference of k-TLSS's:

<u>k-TSSI algorithm</u> // Obtains a DFA which accepts the smallest k-TSSL containing R//

<u>Input</u>: $k \geq 2$, R: set of training strings.

// the case k=1 is trivial since $\forall R$ $Z_1(R)=(\Sigma(R),\{e\},\{e\},\phi)$ and $l_1(R)=\Sigma(R)^*$//

<u>Output</u>: DFA $A_k=(Q,\Sigma,\delta,q_0,Q_f)$ //$Q \subseteq \bigcup_{i=0}^{k-1} \Sigma^i$, $q_0 \epsilon Q$, $Q_f \subseteq Q$, $\delta \subseteq (Q \times \Sigma \times Q)$//

<u>Method</u>

```
(Σ,I,F,T):= (Σ(R), I_k(R), F_k(R), T_k(R));
Q:= {e};  δ:= φ;  q_0:= e;      // e is the symbol for the null
                                       string //
∀ a_1...a_m εI for j:=1 to m
    Q:= Q U {a_1...a_j};
    δ:=  δ  U {(a_1...a_{j-1},a_j,a_1...a_j)};    //a_i...a_j = e iff
                                                     j<i//
end for end ∀
∀ a_1...a_k ε(Σ^k-T)
    Q:= Q U {a_2...a_k};
    δ:= δ U {(a_1...a_{k-1},a_k,a_2...a_k)};
end ∀
Q_f:= F; A_k:= (Q,Σ,δ,q_0,Q_f);
end k-TSSI
```

The following examples illustrate the proposed learning algorithm:

<u>Example 1</u>. Let R = {abba, aaabba, bbaaa, bba} and k = 2. Then $Z_2(R) = (\{a,b\}, \{a,b\}, \{a\}, \phi)$. Also if k = 3, then $Z_3(R)=(\{a,b\}, \{aa,ab,bb\}, \{aa,ba\}, \{a,b\}^3-\{aaa,aab,abb,baa,bba\})$. From $Z_2(R)$ and $Z_3(R)$ and following the k-TSSI algorithm, we obtain the automata A_2 and A_3 show in Fig. 4.1

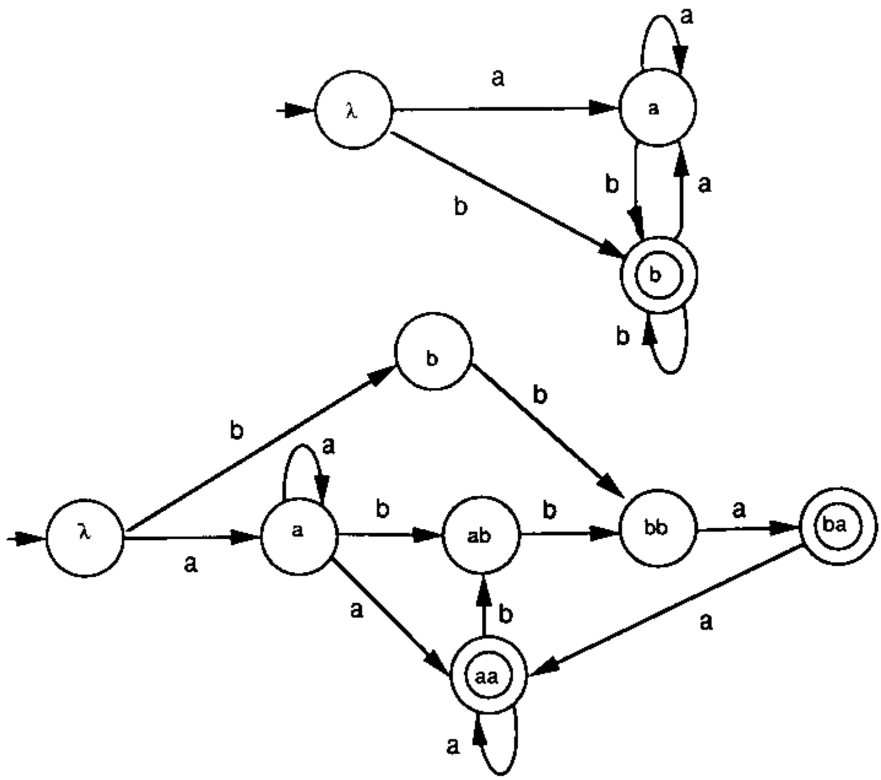

Fig. 4.1: Inferred automata for k=2 and k=3.

The correctness of the k-TSSI algorithm is established by the following theorem [9]:

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

Proof.

1. $l_k(R)\subseteq L(A_k)$.

Let $x=a_1\ldots a_m\in l_k(R)$

a) Let $m\geq k$. Then:

$a_1\ldots a_{k-1}\in I_k(R)$; $a_{m-k+2}\ldots a_m\in F_k(R)$;

$a_j\ldots a_{j+k-1}\in\Sigma^k(R)-T_k(R)$, $j=1,\ldots,m-k+1$.

From the k-TSSI algorithm

$\delta(a_1 \ldots a_{j-1}, a_j) = a_1 \ldots a_j$, $j=1,\ldots,k-1$, $//a_1a_0=e//$

$\delta(a_i \ldots a_{i+k-2}, a_{i+k-1}) = a_{i+1} \ldots a_{i+k-1}$, $i=1,\ldots,m-k+1$

$a_{m-k+2} \ldots a_m \epsilon Q_f$

We can now conclude that $\delta(e, a_1 \ldots a_m) = a_1 \ldots a_m \epsilon Q_f$, and $x \epsilon L(A_k)$.

b) Let m<k. Then: $a_1 \ldots a_m \epsilon I_k(R) \cap F_k(R)$.
From the above construction $\delta(a_1 \ldots a_{j-1}, a_j) = a_1 \ldots a_j$, $j=i,\ldots,m$, and $a_1 \ldots a_m \epsilon Q_f$.

But then $\delta(e, a_1 \ldots a_m) = a_1 \ldots a_m \epsilon Q_f$, and $x \epsilon L_k(R)$.

2. $L(A_k) \subseteq l_k(R)$
Since, by construction, A_k is deterministic, and consequently unambiguous, this can be easily proved by following the steps of the previous part of the proof in reverse order. ■

From this theorem and 3.1, we see that the automaton A_k inferred from R for a given value of k≥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 4.2).

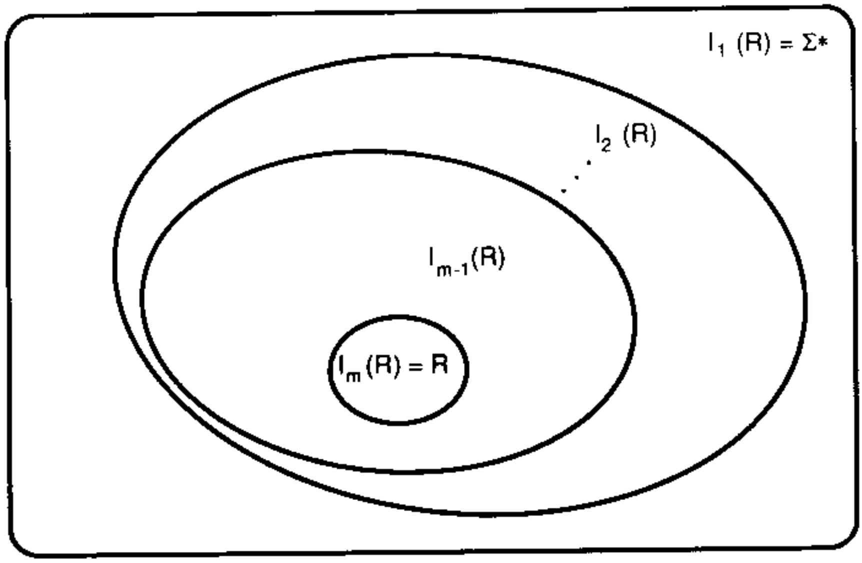

Fig.4.2. Range of languages which can be inferred with the k-TSSI algorithm from a given positive sample ($m>\max_{x\in R}|x|$).

5. 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 [3]. In particular, every finite collection of languages is identifiable in the limit from positive presentation. For instance, given a finite class of languages, an algorithm with input R which obtains the smallest language in the class containing R, identifies such a class in the limit.

Theorem 5.1. The k-TSSI Algorithm identifies any k-TLSS in the limit from positive data.

Proof. Given a finite alphabet Σ and a positive integer k, the number of different k-TLSS's over Σ is finite. Theorems 4.1 and 3.1 suffices for proving that the k-TSSI algorithm identifies the class of k-TLSS's. ∎

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 [11]. From Theorems 3.3, 4.1, and 5.1, 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 nègative 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 [11].

6. 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 non-trivial k-TLSS's $|I_0|\le|T'|$, and since for every finite automaton $|Q|\le|\delta|$, we can write:

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

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:

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

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 $|\Sigma^k(R)-T_k(R)| \leq n = \sum_{x\epsilon R}|x|$, 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 of R, and then the inferred automaton will in fact stop growing, whilest 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 $Z_k=(\Sigma_k,I_k,F_k,T_k)$ be a four-tuple defining a k-testable language $l(Z_k)$, let $R\subseteq l(Z_k)$ 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 $O(knlogm)$ time, and represented using $O(m|\Sigma|)$ space, where $n=\sum_{x\epsilon R}|x|$ 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(klogm) time [2],[9].

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 $m \le |\Sigma|^k$, 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 $|\Sigma^k - T_k(R)| \le n$ to obtain an inference time bound in O(knlogn). In this case, however, the same remarks that have been made above about the bounds (6.3) apply.

7. LOCALLY TESTABLE LANGUAGES IN THE STRICT SENSE AND REVERSIBLE LANGUAGES.

Recently, Angluin has proposed so called k-RI algorithm which allows the identification of the class of k-Reversible (k-R) languages, ($k \ge 0$) from positive data [4]. Following Angluin, a regular language L is said to be k-R iff whenever $u_1vw, u_2vw \in L$ and $|v| = k$, then $(u_1v)^{-1}(L) = (u_2v)^{-1}(L)$ where, for every $x \in \Sigma^*$, $x^{-1}(L) = \{y \in \Sigma^* \mid xy \in L\}$

<u>Theorem 7.1</u>. Let L be a regular language. If L is (k+1)-TLSS, then L is k-Reversible.

<u>Proof</u>

One may verify that for every $u,v,w \in \Sigma^*$ and $|v|=k$, the following relation hold:
$i_k(uvw) = i_k(uv)$, $f_k(uvw) = f_k(vw)$, and
$t_{k+1}(uvw) = t_{k+1}(uv) \cup t_{k+1}(vw)$.

Let L be a (k+1)-TLSS, and $u_1vw, u_2vw \in L$, $|v|=k$. Then:

$(i_k(u_1v) \cup i_k(u_2v)) \subset I_{k+1}$; $f_k(vw) \in F_{k+1}$

$(t_{k+1}(u_1v) \cup t_{k+1}(u_2v) \cup t_{k+1}(vw)) \subset \Sigma^{k+1} - T_{k+1}$

We now show that for every $z \in \Sigma^*$, $u_1vz \in L$ <u>iff</u> $u_2vz \in L$.

Let $u_1vz \in L$. Then $f_k(vz) \in F_{k+1}$ and $t_{k+1}(vz) \subset \Sigma^{k+1} - T_{k+1}$. But then $u_2vz \in L$. The reciprocal follows analogously. ■

Theorem 7.1 implies that the methods proposed in [4] could be seen as applicable for the inference of k-TLSS's. However, the time complexity of Angluin's inference algorithm is $O(kn^3)$, and then, if the languages of interest can be assumed to belong to the class of LTLSS's, the methods proposed here can be chosen with certain advantage over the (most general) methods available for Reversible Languages. It should be noted that although the automata inferred by the k-TSSI algorithm are not minima, their size is bounded; then, from a practical point of view efficient recognition algorithms can be applied [10]. Furthermore, even taking into account the work required for minimizing these automata, the overall cost remains advantageous.

8. RELATED WORKS

Recently an algorithm wich is essentialy the same that our k-TSSI has been presented, although from one point of view slightly different [21]. The more original aspect of this work is the study of the polynomial identifiability of the algorithm considered in the way of Pitt [14]; i.e. a identifiable class of languages C is identifiable in polynomial time using a given class of representations iff there exist an algorithm identifying C such that: 1) it has the polynomial update time property, and 2) there exist a polynomial p such that for any n, and for any $L \in C$ that has a correct representation of size n, and for every presentation of L, the number of implicit errors of prediction made by the learning algorithm is at most p(n). Yokomori proved in his version of the k-TSSI algorithm that the number of implicit errors of prediction is bounded by $|\Sigma|n$, where n is the number of states of the cannonical acceptor of the unknown k-TLSS [22].

8. CONCLUDING REMARKS

As it has been discussed in section 1, k-Testable Languages in the Strict Sense can be adequately used to modelize certain interesting Pattern Recognition problems [10]. Furthermore, following the representation theorem that establishes that every regular set is the homomorphic image of a local language, the inference of k-TLSS's can be used as the basis of a general methodology for the inference of regular languages [8]. An extension of these results to the inference of the most general class of (non-strict) k-Testable Languages from positive data does not seem easy. It can be easily shown from [3] that this class is also identifiable. The question remains as to whether an efficient procedure for carrying out the inference exists. While a straighforward linear-time algorithm is proposed in [21] for obtaining the sets of k-test vectors that represent the k-Testable Language desired, whether a standard language representation (grammar or automaton) for this language can be efficiently obtained from these sets remains to be done.

REFERENCES.

[1] Abramson, N. "Information Theory and Coding". McGraw-Hill, 1966.

[2] Aho, A.V., Hopcroft, , and Ullman, J.D. "The Design and Analysis of Computer Algorithms". Addison-Wesley, 1974.

[3] Angluin, D. "Inductive Inference of formal languages from positive data". Inf. and Control, 45, 1980, pp. 117-135.

[4] Angluin, D. "Inference of reversible languages". Journal of the Association for Computing Machinery, vol.29, No.3, 1982, pp. 741-765.

[5] Brzozowski, J.A., and Simon, I. "Characterizations of locally testable events". Discrete Mathematics, 4, 1973, pp. 243-271.

[6] Bahl, L.R., Jelinek, F., and Mercer, L.R. "A maximum likelihood approach to continuous speech recognition". IEEE Trans. PAMI-5, No.2, 1983, pp. 179-190.

[7] Derouault, A.M. and Merialdo, B. "Natural language modelling for phoneme to text transcription". IEEE Trans. PAMI-8, 1986, pp. 742-749.

[8] García, P., Vidal, E., and Casacuberta, F. "Local Languages, the Successor Method, and a step towards a general methodology for the inference of Regular Grammars" IEEE Trans. PAMI-9, N. 6, 1987, pp. 841-845.

[9] García, P. "Explorabilidad Local en Inferencia Inductiva de Lenguajes Regulares y Aplicaciones". Doct. Diss., Departamento de Sistemas Informáticos y Computación. Universidad Politécnica de Valencia, 1988.

[10] García, P., and Vidal, E. "Inference of k-Testable Languages in the Strict Sense and applications to Syntactis Pattern Recognition". IEEE Transactions on Pattern Analysis and Machine Intelligence, (to be published), 1988.

[11] Gold, E.M. "Language identification in the limit". Information and Control, 10, 1967, pp. 447-474.

[12] McNaughton, R. "Algebraic decision procedures for local testability". Mathematical Systems Theory, vol.8, No.1, 1974, pp. 60-76.

[13] Peterson, J.L. "Computer programs for spelling correction". Springer-Verlag, 1980.

[14] Pitt,L. "Inductive inference, dfas, and computational complexity". In Proceedings of 2st Workshop on Analogical and Inductive Inference, Lecture Notes in Artificial Intelligence, Springer-Verlag 397, 1989, pp. 18-44.

[15] Salomaa,A. "Jewels of Formal Language Theory". Computer Science Press, 1981.

[16] Shannon, C.E. "A Mathematical Theory of Communicatiòn". Bell System Tech. J. Vol. 27, 1948, pp. 390-397.

[17] Shannon, C.E. "Prediction and entropy of printed English". Bell System Tech. J. Vol. 30, No. 1, 1951, pp. 50-64.

[18] Smith, A.R., Denemberg, J.N., Slack, T.B., Tan, C.C., and Wohlford, R.E. "Application of a Sequential Pattern Learning System to Connected Speech Recognition". ICASSP-85, 1985, pp. 31.2.1-31.2.4.

[19] Venta, O. "A very fast sentence reconstruction method for the postprocessing of computer-recognized continuous speech. ICPR 84 Proc., 1984, pp. 1240-1243.

[20] Venta, O. and Kohonen, T. "A non-stochastic method for the correction of sentences". ICPR-86, 1986, pp. 1214-1217.

[21] Yokomori, T. "Learning Local Languages from Positive Data". Procc. in the FUJITSU IIAS-SIS Workshop on Computational Learning Theory '89.

[22] Yokomori, T. "A note on the polynomial-time identification of Strictly Local Languages in the Limit. (A preliminary report)". Unpublished manuspcript. 1990.

[23] Zalcstein, Y. "Locally testable languages". Journal of Computer and System Sciences 6, 1972, pp. 151-167.