Transducer Learning in Pattern Recognition*

José Oncina[1], Pedro García[2] and Enrique Vidal[2]

[1]Dept. de Sistemas Informáticos y Computación Universidad de Alicante
[2]Dept. de Sistemas Informáticos y Computación Universidad Politécnica de Valencia
Spain

Abstract

"Interpretation" is a general and interesting Pattern Recognition framework in witch a system is considered to input object representations, and output the corresponding Interpretations in terms of "semantic messages" specifying the actions to be carried out as system's responses. Viewed under the Syntactic Pattern recognition viewpoint, interpretation reduces to Formal Transduction. We propose here an efficient and effective algorithm to automatically infer a finite state Transducer from a training set of input-output examples of the Interpretation problem considered. The proposed algorithm has been shown to identify in the limit an important class of transductions known as "Subsecuential Transductions." Experimental results are presented showing the performance and capabilities of the proposed method.

1. Introduction

From a very general point of view, any Pattern Recognition (PR) system can be seen as a function that take object representations as input and produces, as output, adequate interpretations of these objects. Without loss of generality, these interpretations can be expressed as sentences of an appropriate Semantic Language. From this perspective, the classical problem of Classification, witch very often is considered as a paradigmatic problem of PR, results just in a very particular case; namely, that in which the range of the interpretation function is finite.

While classification is a well known problem which quite naturally fits the traditional decision-theoretic-based approach to PR, the interpretation setting has been much less studied. Nevertheless, the fact that interpretation results can always be expressed as sentences of certain language, strongly suggest us that the syntactic approach to PR could perhaps be the most appropriate paradigm.

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

This paradigm adopts the classical Theory of Formal Languages as one of its fundamental frameworks and, within this framework, appropriate concepts exist that can properly model the above view of interpretation functions. To be more specific, a Transducer is a formal device that takes structural representations of objects (strings over an alphabet) as input and produces, as output, sentences over another (semantic) output alphabet. Specially interesting because of their applicability to PR are the Finite Transducers [1].

However, perhaps the main drawback of the Transducer approach to PR, is the difficulty of (automatically) building such models. Here (like in the case of Languages) adequate techniques are needed to inductively learn the required (Syntactical) models from training sets of input-output examples. However, in contrast with the case of Languages, where a certain tradition already exists in Grammatical Inference methods [2-4], only very restrictive (sequential) models, such as Mealey or Moore machines, and/or rather heuristic approaches seem to have been dealt so far [5-9].

This work deals with the learnibility of an important subclass of Rational Transductions within the Gold's paradigm of identification in the limit [10]. The class considered is the class of the ***Subsecuential Transductions*** which is a subclass of the most general Rational or Finite-State Transductions and properly contains the class of Sequential Transductions [1]. A Sequential Transduction is one that preserves the increasing length prefixes of input-output strings. While this can be considered as a rather "natural property" of transductions, there are many real-world situations in which such a strict sequentiality is clearly inadmissible. The class of ***Subsequential Transductions*** comes to make this restriction milder, therefore allowing applicability in quite a few interesting practical situations.

The capabilities of subsequential transductions are illustrated through a series of experiments which also show the high effectiveness of the here proposed learning

0-8186-2915-0/92 $3.00 © 1992 IEEE

method to obtain very accurate and compact transducers for the corresponding tasks.

This paper is a preliminary report of a more comprehensive work witch is being presented elsewhere [11-12].

2. Mathematical background and notation

Let X be a finite set or *alphabet*, and X^* the *free-monoid* over X. For any string $x \in X^*$, $|x|$ denotes the *length* of x and λ is the symbol for the string of length zero. For every $x,y \in X^*$, xy is the *concatenation* of x and y. If v is a string in X^*, then X^*v (vX^*) denotes the set of all strings of X^* that end (begin) with v. $Pr(x)$ denotes the set of *prefixes* of x. Given $u,v \in X^*$, with $u \in Pr(v)$, we define $u^{-1}v$ as $u^{-1}v=w \Leftrightarrow v=uw$. Given a set $L \subseteq X^*$, the *longest common prefix* of all the strings of L is denoted as $lcp(L)$.

In general, a *transduction* from X^* to Y^* is a relation $t \subseteq (X^* \times Y^*)$. In what follows, only those transductions which are *partial functions* from X^* to Y^* will be considered. A *Subsecuential Transducer* is defined as a 6-tuple $\tau = (Q,X,Y,q_0,E,\sigma)$, where Q is a finite set of *states*, $q_0 \in Q$ is the *initial state*, E is a finite subset of $(Q \times X \times Y^* \times Q)$ whose elements are called *transitions*, and $\sigma: Q \rightarrow Y^*$ is a function that assigns output strings to the states of τ. The partial function $t: X^* \rightarrow Y^*$ that is realized by τ is defined as:

$t(x_1x_2..x_n) = y_1y_2..y_n\sigma(q)$ *iff*

$\{(q_0,x_1,y_1,q_1), (q_1,x_2,y_2,q_2), .., (q_{n-1},x_n,y_n,q_n)\} \subset E$.

By using an additional input symbol "#", not in X, to mark the end of the input strings, we can represent the function σ as a new edge of the form $(q,\#,y,q')$ where $\sigma(q)=y$ and q' is a state non necessarily in Q.

Following e.g. [10], [3], an appropriate framework for the Transducer Learning Problem can be stabilised as follows: Let $f: X^* \rightarrow Y^*$ a partial recursive function, a Transducer Learning Algorithm A is said to identify f in the limit if, for any (positive) presentation of the input-output pairs (graph) of f, A converges to a transducer τ that realizes a function $g: X^* \rightarrow Y^*$ such that $\forall x \in Dom(f)$, $g(x) = f(x)$, where $Dom(f)$ denotes the subset of X^* where f is defined.

3. Onward Subsecuential Transducers

The transducer learning algorithm to be presented in the next section requires some additional new concepts to be introduced. An *Onward Subsequential Transducer* (*OST*) is a subsequential transducer $\tau=(Q,X\cup\{\#\},Y,q_0,E)$ such that:

$$\forall p \in Q-\{q_0\} \quad mcp(\{y \in Y^* \mid (p,a,y,q) \in E\}) = \lambda.$$

Given an arbitrary subsequential transducer τ, an equivalent OST τ' can be obtained as follows [18](fig. 1):

$\forall q \in Q$, **if** $w = lcp\{v \in Y^* \mid (q,a,v,r) \in E\} \wedge w \neq \lambda$ **then**

1: substitute every outgoing transition of q: (q,a,wz,r) for (q,a,z,r)

2: substitute every ingoing transition of q: (p,b,y,q) for (p,b,yw,q)

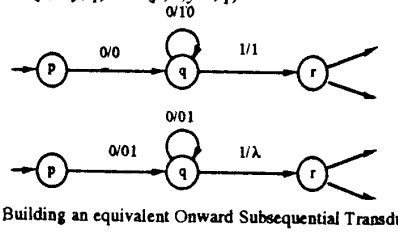

Fig.1. Building an equivalent Onward Subsequential Transducer

The transduction learning algorithm requires a finite sample of input-output pairs $T \subset (X^*\# \times Y^*)$ which is assumed to be single-valued. Such a sample can be properly represented by a *"Tree Subsequential Transducer"* (*TST*) $\tau=(Q,X\cup\{\#\},Y,q_0,E)$ with $Q = \bigcup_{(u\#,v)\in T} Pr(u\#)$, $q_0 = \lambda$, and $\forall wa \in Q$, $(w,a,\lambda,wa) \in E$, and $\forall (u\#,v) \in T$, $(u,\#,v,u\#) \in E$.

Given T, an *Onward Tree Subsequential Transducer* (*OTST*) representing T can be obtained by building an *OST* equivalent to the TST of T.

Example 2: Let $T = \{(0\#,1), (1\#,01), (00\#,10), (01\#,11), (11\#,001)\}$. The TST and OTST of T are as shown in Fig.2.

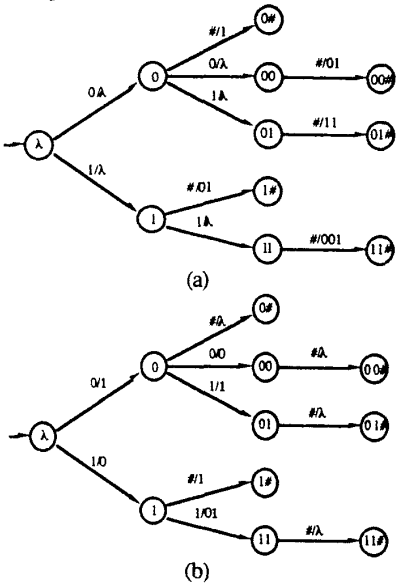

Fig 2. Tree Subsequential Transducer *TST(T)* (a), and Onward Tree Subsequential Transducer *OTST(T)* (b)

4. The Transducer Learning Algorithm

Let $T \subset (X^*\# \times Y^*)$ be a finite single-valued training set. The proposed algorithm starts building the OTST of T, $\tau = (Q, X \cup \{\#\}, Y, q_0, E) = OTST(T)$, and then proceeds by orderly trying the merge of states of τ while preserving the subsequential, deterministic nature of the resulting transducers. For this to be possible, some output (sub)strings often need to be *"pushed-back"* towards the leaves of τ. The test as to whether a transducer τ is subsequential, is assumed to be provided by a hypothetical (no cost) procedure *"Subseq"*.

The merging process requires the states of τ to be successively taken into account in a *lexicographic order* of the names given to these states through the TST construction (Sect.3) [18]. Let "<" be such an order on Q, *first* (Q) and *last* (Q) being the first and last states and let *next* (q) denote the state which immediately follows q. The merging of any two states $q', q \in Q$, with $q' < q$, results in a new transducer in which the state q no longer exists and all the outgoing transitions of q are assigned to q'. Let *merge*(τ, q', q) represent this merging.

Let $q \in Q$ be a state of τ and $(q', a, w, q) \in E$ (one of) its ongoing transition(s). Let $w = uv$. Then v can be *"pushed back"* to behind q and distributed throughout all the outgoing transitions of q as follows:

let $\tau' = push\_back(\tau, v, (q', a, uv, q))$.

Then, $E' = (E - \{(q', a, uv, q)\}) \cup \{(q', a, u, q)\} \cup \{(q, b, vz, r) : (q, b, z, r) \in E\}$

Algorithm *OSTIA*
INPUT: Single-valued set of input-output pairs $T \subset (X^*\# \times Y^*)$
OUTPUT: OST τ consistent with T
$\tau := OTST(T)$
$q := first(\tau)$
while $q <$ *last*(τ) **do**
  $q := next(\tau, q)$; $p := first(\tau)$
  while $p < q$ **do**
    $\tau' := \tau$
    merge(τ, p, q)
    while $\neg subseq(\tau)$ **do**
      let (r,a,v,s), (r,a,w,t) be two transitions of τ
    that violate the *subseq* condition, with $s < t$
      if ((v ≠ w) and (a = #)) **or**
        (s < q and v ∉ Pr(w)) **then exit while**
      u:= mcp(v,w)
      push_back$(\tau, v^{-1}u, (r,a,v,s))$
      push_back$(\tau, w^{-1}u, (r,a,w,s))$
      merge(τ, s, t)
    end while //$\neg subseq(\tau)$//
    if $\neg subseq(\tau)$ then $\tau := \tau'$ **else exit while**
    p:= *next*(τ, p)
  end while // p < q //
  if $\neg subseq(\tau)$ **then** $\tau := \tau'$
end while // q < *last*(τ)//
end //*OSTIA*//

Fig. 3. The Transducer Inference Algorithm

The algorithm that performs the above outlined procedures is called the *"Onward Subsequential Transducer Inference Algorithm (OSTIA)"* and is formally presented in Fig. 3 (see [12] for examples of how it works).

It can be shown that, using this algorithm, the class of subsecuential transducers can be identified in the limit [11][12].

5. Experiments

To illustrate the how this algorithm works with complex tasks we present two different groups of experiments. In both we have used a series of increasing-size random *training-sets*, each including the previous ones. Each set was drawn from a non-uniform distribution in witch the lengths of the strings were (approximately) equiprobable. The random procedure was prevented from generating repeated samples. Every training set was submitted to the *OSTIA* and each resulting Subsequential Transducer was used to perform the task with a test set.

The first problem was to learn to perform *integer division by seven* and the *test-set* consisted of a whole range of Integers from *1* to 10^5-*1*. The results of this experiment appear in fig. 8. The items shown are: (a) the test-set error rates, (b) the sizes of the learnt transducers and (c) the computing time required for each execution of OSTIA, on a ~25 mips. conventional RISC Computer (HP 9000/35).

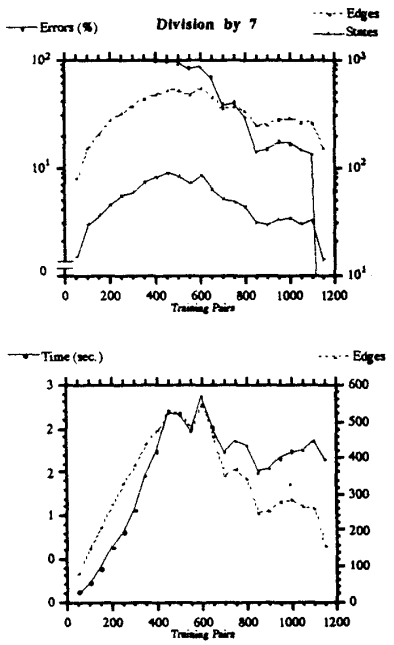

Fig. 4. Behaviour of the OSTIA for the Division by seven translation task

A second group of experiments was concerned with the more difficult task of translating *English writen numbers* in the $0...10^6-1$ range into their corresponding *Spanish* writing. The results appear in fig. 5, withc shows the same items as before.

It is worth noting that, with small training-sets, the inferred transducers tend to be rather large and error-prone, while both sizes and errors reduce dramatically as enough source structure is made available through the training data.

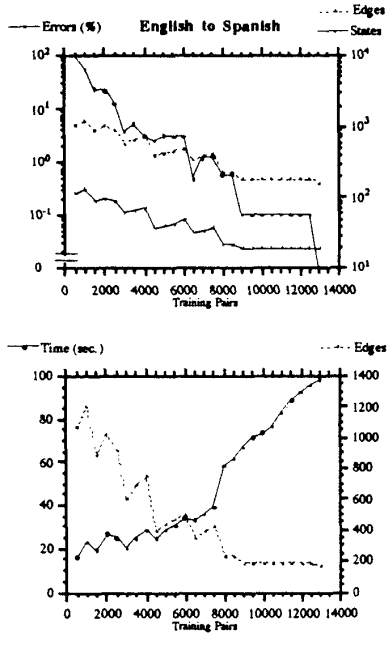

Fig. 5. Behaviour of the *OSTIA* for the English to Spanish translation task

6. Discussion and Conclusion

The results of the experiments described in the last section clearly indicate both the versatility of subsequential transduction and the effectiveness of the *OSTI* Algorithm to learn subsequential transducers from training input-output examples. While, for (small) training-sets not conveying enough structure of the unknown source transduction, the transducers produced by *OSTIA* tend to be rather large and inaccurate, very compact and exact solutions are always obtained once the training data contain a small number of *appropriate* input-output pairs. The existence of such small sets of appropriate training data has been shown through some of the experiments reported in [12], and some theoretical results regarding what an "appropriate" set of input-output training pairs is, are discussed in [11]. However, the only practical hint these results seem to suggest is that such training data should contain the "simplest" (usually also the shortest) transduction examples, and how to actually choose adequate and small sets of training data remains an open issue of practical concern. In any case, as the here presented experiments suggest, by relying on chance alone good results tend to be obtained with reasonable amounts of training-pairs.

7. References

[1] J. Berstel. "Transductions and Context-Free Languages". *Teubner*, Stuttgart 1979.

[2] K. S. Fu and T. L. Booth. "Grammatical Inference: Introduction and Survey". parts 1 and 2. *IEEE Trans. SMC*, SMC-5: pp 95-111, 409-423, 1975.

[3] D. Angluin and C. H. Smith. "Inductive inference: theory and methods". *Computing Surveys*, 15(3), pp 237-269, 1983.

[4] L. Miclet. "Grammatical Inference". In *Syntactic and Structural Pattern Recognition*. H. Bunke and A. San feliu (eds.) *World Scientific*, 1990.

[5] E. M. Gold. "Complexity of automaton identification from given data". *Information and Control*, 37 pp 302-320, 1978.

[6] L. P. J. Veelenturf. "Inference of sequential Machines from sample Computation". *IEEE Trans. on Computers*. 27, pp 167-170, 1978.

[7] P. Luneau, M. Richetin and C. Cayla. "Sequential Learning from Input-Output Behaviour". *Robotica* 1, pp 151-159, 1984.

[8] Y. Takada. "Grammatical inference for even linear languages based on control sets". *Information Processing Letters*, 28(4): 193-199, 1988.

[9] E. Vidal, P. García and E. Segarra. "Inductive Learning of finite-state transducers for the interpretation of unidimensional objects". In *Structural Pattern Analysis*. Mohr, Pavlidis and San feliu (eds.). World Scientific, 1990.

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

[11] J. Oncina, P. García. "Aprendizaje de Funciones Subsecuenciales". Universidad Politécnia de Valencia, Tech. Report DSIC II-5/1991.

[12] J. Oncina, P. García, E. Vidal. "Learning Subsecuential Transducer For Pattern Recognition Interpretation Tasks". Submited for Publication. 1991.