Language Understanding and Subsequential Transducer Learning

Antonio Castellanos,

Departamento de Informática
Universidad Jaime I de Castellón, Spain

Enrique Vidal,

Departamento de Sistemas Informáticos y Computación,
Universidad Politécnica de Valencia, Spain

Miguel A. Varó and José Oncina

Departamento de Lenguajes y Sistemas Informáticos,
Universidad de Alicante, Spain

Running head: L. Understanding & Subseq. T. Learning

Abstract

Language Understanding can be considered as the realization of a mapping from sentences of a natural language into a description of their meaning in an appropriate formal language. Under this viewpoint, the application of the Onward Subsequential Transducer Inference Algorithm (OSTIA) to Language Understanding is considered. The basic version of OSTIA is reviewed and a new version is presented in which syntactic restrictions of the domain and/or range of the target transduction can effectively be taken into account. For experimentation purposes, a task proposed by Feldman, Lakoff, Stolcke and Weber (1990) for assessing the capabilities of language learning and understanding systems has been adopted and three semantic coding schemes have been defined for this task with different sources of difficulty. In all cases the basic version of OSTIA has consistently proved able to learn very compact and accurate transducers from relatively small training sets of input-output examples of the task. Moreover, if the input sentences are corrupted with syntactic incorrectness or errors, the new version of OSTIA still provides understanding results that only degrade in a gradual and natural way.

1 Introduction

The process of understanding language can be seen as the realization of a mapping from the set of sentences of the given (input) language into a set of (output) semantic messages (SM) that belong to the semantic universe of the language considered. In most cases of interest, these SM are just convenient ways of specifying the actions to be carried out as a response to the meaning conveyed by the corresponding input sentences. Thus, an appropriate and general way of representing SM is in terms of strings of an adequate output (semantic) language in which the required actions can be specified. For instance, the sequence of operations to be performed by a robotized machine tool as a response to an input specification formulated in natural language, can be properly specified as a sentence of the command language of the machine. Similarly, an SQL command constitutes an adequate way to specify the retrieving actions to be carried out as a response to a natural language input query to a certain Data Base.

Such a point of view of Language Understanding (LU) directly fits within the framework of *Transduction*. A transducer is a formal device which inputs strings from a certain input language and outputs strings from another (usually different) output language. While many interesting properties of transducers are known from the classical theory of Formal Languages, and while their application in the field of Computer Languages has become quite popular, Formal Transduction has only recently started to be explored in the field of LU (Vidal, García & Segarra, 1990). Pieraccini, Levin & Vidal (1993) addressed the problem of performing sequential transductions through stochastic finite state networks. The information-theoretic connectionist approaches for language learning and understanding presented by Gorin, Levinson, Gertner & Goldman (1991) and Gorin (1995) are based on the basic idea of mapping input sentences into semantic actions, although they are not properly based on the concept of transduction. More recently, a statistical approach to language understanding based on the source-channel paradigm has been presented which explicitly models the understanding process as a transduction from natural language into a formal language, through a statistical translator (Epstein, Papineni, Roukos, Ward & Della Pietra, 1996; Della Pietra, Epstein, Roukos & Ward, 1997).

In this paper, we consider the use of a class of transducers, known as subsequential transducers, for representing adequate input-output mappings associated with LU tasks. The class of *subsequential transducers* (*transductions*) is a subclass of the class of *rational* or *finite state transducers* (*transductions*) and properly contains the class of *sequential transducers* (*transductions*) (Berstel, 1979). A sequential transduction is one that preserves the increasing length prefixes of input-output strings (Berstel, 1979). 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 makes this restriction milder, therefore allowing application in many interesting practical situations. Apart from this flexibility, perhaps more important is that subsequential transducers have been recently proved *learnable* from positive presentation of *input-output examples* (Oncina & García, 1991; Oncina, García & Vidal, 1993).

Theoretical issues related with *subsequential transducer learning* have been thoroughly studied in (Oncina, 1991) and (Oncina & García, 1991). The main result is an algorithm called OSTIA (Onward Subsequential Transducer Inference Algorithm) and a proof that, using OSTIA, the whole class of *total* subsequential transductions can be identified in the limit (Gold, 1967; Angluin & Smith, 1983) from positive presentation of input-output examples. Section 2 of this paper outlines this basic algorithm and its theoretical properties.

On the practical side, subsequential transduction learning has been suggested as an appropriate way to deal with *interpretation* Pattern Recognition problems (Oncina *et al.*, 1993). Also, several examples showing the capabilities of subsequential transductions and OSTIA have been presented in (Oncina, 1991; Oncina, García & Vidal, 1992; Oncina *et al.*, 1993). While very accurate transducers where obtained in all the transduction tasks considered in these works, it has been argued that translations can become quite incorrect if even slightly incorrect input strings are submitted to the learned transducers (Oncina *et al.*, 1993). This is related to the *partial function* nature of the mappings underlying these tasks: it can be seen that partial subsequential transductions are not guaranteed to be learned using *only* positive information. In order to avoid these problems some hints were already proposed in (Oncina *et al.*, 1993) which would make use of complementary information about the mapping to be learned. In particular, knowledge about the domain and/or range of this mapping can effectively be used (Oncina, Castellanos, Vidal & Jiménez, 1994; Oncina & Varó, 1996), leading to the algorithm called OSTIA-DR (Onward Subsequential Transducer Inference Algorithm with Domain and Range) which is presented in detail in section 3.

As previously mentioned, the present work deals with an application of subsequential transductions and OSTIA in the field of LU. While one can argue that the kind of syntactic-semantic mapping actually underlying general LU can be quite contrived, and that no simple class of formal devices would perhaps ever be powerful enough to completely model such a mapping, we will show throughout this paper that, in practical situations, simple and useful semantic languages can be quite naturally adopted that allow the mapping to be properly modeled through subsequential transductions. Furthermore, we will show how the OSTIA can be effectively and efficiently applied to automatically discover such a mapping from adequate sets of input-output examples.

For this study, we have adopted a compact and theory-free learning task that was recently introduced in the general context of Cognitive Science as a *touchstone* for showing the capabilities of learning systems. This task is the so-called *Miniature*

Language Acquisition (MLA) task, proposed by Feldman, Lakoff, Stolcke & Weber (1990). It presents fundamental challenges to several areas of Cognitive Science including language, inference and learning. Thus, it may easily be reformulated to be a paradigmatic task in the LU framework as well. The task consists of understanding the meaning of pseudo-natural English sentences that describe simple visual scenes. These scenes may involve different objects in different relative positions and each object possibly has a different shape, size and/or shade. A restricted version of the MLA task was considered by Stolcke (1990) using Recurrent Neural Networks, with fairly good results. A detailed description of the MLA task is given in section 4.

In order to frame the MLA task into our transducer learning paradigm, an adequate output language is required to conveniently state the semantic contents of each English input sentence of this task; i.e., to describe the visual scene involved. In this work, we have adopted three logic languages with different sources of difficulty which will be described in section 5. Both OSTIA and its new version, OSTIA-DR, were then used to infer subsequential transducers for the MLA task. These experiments are described in section 6. The main conclusions of this work, as reported in section 7, are that, in limited domain tasks, LU can be properly and conveniently formulated as a problem of subsequential transduction and that the required transduction devices can be quite effectively learned using OSTIA and OSTIA-DR

2 Onward Subsequential Transducer Inference Algorithm (OSTIA)

Formal and detailed descriptions of the Onward Subsequential Transducer Inference Algorithm (OSTIA) have been presented elsewhere (Oncina *et al.*, 1992; Oncina *et al.*, 1993); nevertheless, for the sake of completeness, we will review here some basic concepts and procedures.

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^* and $L \subseteq X^*$, then Lv (vL) denotes the set of strings $xy \in L$ such that $y = v$ ($x = v$). Hence, X^*v (vX^*) denotes the set of all strings of X^* that end (begin) with v, while $\emptyset v = v\emptyset = \emptyset$ (the empty set). $Pr(x)$ denotes the set of *prefixes* of x; i.e., $Pr(x) = \{y \in X^* | yz = x, z \in X^*\}$. Given $v \in X^*$ and $u \in Pr(v)$, we define the suffix of v with regard to u 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^* into Y^* is a function from X^* into 2^{Y^*} of subsets of Y^*; i.e., $t : X^* \rightarrow 2^{Y^*}$. A *partial function* t from X^* into Y^*, $t : X^* \rightarrow Y^*$, is a transduction from X^* into Y^* such that $\forall x \in X^*$, $|t(x)| \leq 1$, where $|t(x)|$ stands for the cardinality of $t(x)$. A *total function* t from X^* into Y^*, $t : X^* \rightarrow Y^*$, is a partial function such that $\forall x \in X^*$, $t(x) \neq \emptyset$; i.e., a transduction from X^* into Y^* such that $\forall x \in X^*$, $|t(x)| = 1$. In what follows, only those transductions which are partial functions from X^* to Y^* will be considered, and they will be called transductions or functions indistinctly. Given a transduction t, $dom(t)$ and $ran(t)$ denote the sets of input (domain) and output (range) strings of the pairs of t, respectively.

A *sequential transducer* is defined as a 5-tuple $\tau = (Q, X, Y, q_0, E)$, where Q is a finite set of *states*, X and Y, respectively, are input and output *alphabets*, $q_0 \in Q$ is the *initial state*, and E is a finite subset of $(Q \times X \times Y^* \times Q)$ whose elements are called *edges* or *transitions*. This definition is completed requiring τ be *deterministic*; that is to say, $\forall (q, a, u, r), (q, a, v, s) \in E \Rightarrow (u = v \wedge r = s)$.

The *sequential transduction* realized by a sequential transducer τ is the partial function $t : X^* \rightarrow Y^*$, defined as:

$$t(x_1 x_2 \ldots x_n) = y_1 y_2 \ldots y_n \Leftrightarrow (q_0, x_1, y_1, q_1)(q_1, x_2, y_2, q_2) \ldots (q_{n-1}, x_n, y_n, q_n) \in E^*$$

with $n \geq 0$; that is to say, $y_1 y_2 \ldots y_n$ is the concatenation of the output substrings associated to the corresponding input symbols $x_1 x_2 \ldots x_n$ which match a sequence of edges (*path*) of the transducer, starting at the initial state. When intermediate states are not important, a sequence of edges $(p, x_1, y_1, q_1) \ldots (q_{n-1}, x_n, y_n, q_n) \in E^*$ with $n \geq 0$ will be expressed as $(p, x_1 \ldots x_n, y_1 \ldots y_n, q_n) \in E^*$. Sequential transductions have the *property of preserving prefixes*, that is, $t(\lambda) = \lambda$ and if $t(uv)$ exists then $t(uv) \in t(u)Y^*$.

This property becomes an important limitation in many real world situations. For instance, translation of the adjectives of a noun from Spanish into English cannot be represented by a sequential transduction. Two simple translations, such as $t(\text{un coche}) = \text{a car}$ and $t(\text{un coche rojo}) = \text{a red car}$, allow for illustrating that, in general, English translation of an article and a noun as a whole Spanish sentence does not constitute an output prefix for the English translation of a new Spanish sentence which adds an adjective to the article and noun. However, this representation limitation for sequential transductions is overcame through subsequential transductions.

A subsequential transducer can be easily defined on the basis of a sequential transducer in the following way. A *subsequential transducer* is a 6-tuple $\tau = (Q, X, Y, q_0, E, \sigma)$, where $\tau' = (Q, X, Y, q_0, E)$ is a sequential transducer and $\sigma : Q \rightarrow Y^*$ is a partial function that assigns output strings to the states of τ. The *subsequential transduction* realized by τ is defined as the partial function $t : X^* \rightarrow Y^*$ such that, $\forall x \in X^*, t(x) = t'(x)\sigma(q)$, where $t'(x)$ is the sequential transduction provided by τ' and q is the last state reached with the input string x. Note that if $\sigma(q) = \emptyset$, $t(x) = \emptyset$ which means that no transduction is defined for x (i.e., q is not an accepting state). For any state q of a subsequential transducer, τ, the set of all the transductions that start in q is called *tail* of q, and denoted $T_\tau(q)$; i.e., $T_\tau(q) = \{(x_1 \ldots x_n, y_1 \ldots y_n \sigma(q_n)) \in X^* \times Y^* | (q, x_1 \ldots x_n, y_1 \ldots y_n, q_n) \in E^*, n \geq 0\}$.

The concept of tail in subsequential transductions is an important concept, since it allows for characterizing minimum subsequential transducers that realize subsequential transductions, in a similar way than the corresponding concept is used in regular languages (Oncina, 1991; Oncina & García, 1992; Oncina *et al.*, 1993).

Fig. 1 illustrates all the above definitions. The sequential function t (Fig. 1(a) and (b)), which simply changes a for 0 and b for 1 in arbitrary strings of a's and b's, can be straightforwardly implemented as a one state sequential transducer (Fig. 1(c)). Function t' (Fig. 1(d) and (e)) is similar to t, but it adds one A or B at the end of the output string depending upon whether the last symbol of the input string is a or b. This function is subsequential but not sequential because for generating the output string additional information which is not provided by each input symbol itself is required. In other words, knowing that a symbol is the last symbol of the string is only possible *after* having read this symbol. In general, this fact makes impossible to associate output substrings requiring such information to the input symbols (i.e., edges of a deterministic transducer). Fig. 1(f) shows a subsequential transducer that implements function t' with the help of additional states and state output symbols[1].

(Figure 1 about here)

From the above definition, it is clear that any subsequential transduction can be realized by different subsequential transducers. Nevertheless, for any subsequential transduction there exists a *canonical subsequential transducer* that has a *minimum number of states* and is *unique* up to isomorphism (Oncina, 1991; Oncina & García, 1991). This transducer adopts an *onward form.* Intuitively, an *Onward Subsequential Transducer* (OST) is one in which the output strings are assigned to the edges and states in such a way that they are as "close" to the initial state as they can be. Formally, a subsequential transducer $\tau = (Q, X, Y, q_0, E, \sigma)$ is an *OST if* $\forall p \in Q - \{q_0\}, lcp(\{y_1 \dots y_n\sigma(q_n) \in Y^* | (p, x_1 \dots x_n, y_1 \dots y_n, q_n) \in E^*, n \geq 0\}) = \lambda.$

The transducers shown in Fig. 1(c and f) are examples of OSTs (see also Fig. 2).

Any unambiguous or *single-valued* finite sample of input-output pairs $T \subset (X^* \times Y^*)$ can be properly represented by a *Tree Subsequential Transducer* (TST) $\tau = (Q, X, Y, q_0, E, \sigma)$, where $Q = \bigcup_{(u,v)\in T} Pr(u)$, $E = \{(w, a, \lambda, wa) | w, wa \in Q\}$, $q_0 = \lambda$, and $\sigma(u) = v \Leftrightarrow (u, v) \in T$ (i.e., $\sigma(u) = \emptyset \Leftrightarrow \forall (u', v') \in T,\ u' \neq u$).

Given T, an *Onward Tree Subsequential Transducer* (OTST) representing T can be obtained by building the OST equivalent to the TST of T. The procedure consists of moving the longest common prefixes of the output strings, level by level, from the leaves of the tree toward the root. Fig. 2 shows an example of TST obtained from a given set of pairs T and the equivalent OTST, according to the above discussed constructions.

(Figure 2 about here)

The *Onward Subsequential Transducer Inference Algorithm* (OSTIA) (Oncina *et al.*, 1993), which is formally presented in Fig. 3, takes as input a finite single-valued training set $T \subset (X^* \times Y^*)$ and produces as output an OST that is a compatible generalization of T. To this end, the OSTIA begins building the OTST which represents T (line 4) and then tries to merge pairs of states of the OTST. Conceptually, each subtree (rooted at the corresponding state) of the OTST represents a subsequential transduction which was contained in the original (source) subsequential transduction from which T has been drawn. Thus, in principle, any two subtrees that represent transductions which are not in contradiction to each other can be *merged* to obtain a new transduction which includes these transductions and, possibly, suitable generalizations compatible with them.

(Figure 3 about here)

The operation $merge(\tau, q, q')$ is assumed to supply a new version of τ in which states q and q' are merged; i.e., all the outgoing edges of q' are assigned to q and q' is removed. After the first merging operation, the resulting whole transducer no longer adopts a tree form, but it is rather a graph. This graph encompasses two significantly different parts. First, a proper subgraph appears that, as the subsequent merging process will be going on, will consolidate as a partial transducer with regard to all previous merge operations. The remaining part of the whole graph contains untouched subtrees of the initial OTST. By iteratively and orderly merging states of the currently consolidated partial transducer with the remaining states that are roots of subtrees, an OST compatible with the whole set T is obtained. This process is carried out in lines 9-19 of the algorithm, which starts merging the root of a subtree with one state of the partially consolidated transducer (initially, it is a subtree too) and then verifies the compatibility of the subtree with this partial transducer.

The possible compatibility is sometimes not obvious and *pushing back* some output substrings toward the leaves of the currently merged subtree is needed to help matching the corresponding structures. If (q, a, v, q') is an edge of the transducer and $u \in Pr(v)$, $push\_back(\tau, u^{-1}v, (q, a, v, q'))$ moves the suffix $u^{-1}v$ in front of the output strings associated to the outgoing edges of q'. This undoes in part the operations carried out to obtain the initial OTST, but allows for adjusting the output strings of the tails of the OTST to the partially consolidated transducer. If, at the end, the subtree is proven not to be compatible, then all the transformations carried out through lines 9 to 19 are discarded and the transducer is restored to its previous consolidated status (line 20); else, all these transformations are consolidated as a new partial transducer. In any case, a new pair of states (the root of a remaining subtree and a state of the partial transducer) will be considered for a next merging.

The merging process requires the pairs of states of the initial tree to be successively taken into account in a certain *order*. It must guarantee that only states rooting remaining real subtrees are merged with states of the currently consolidated partial transducer. The *lexicographic order* of the names given to the states through the TST construction is appropriate for implementing such an order. In the algorithm, the two external loops (lines 6 and 8) manage this ordered state selection through functions *first*(), *last*() and *next*(). In addition, function *next*() takes into account the "jumps" with regard to the initial state ordering produced by the states removed by successful merge operations. The application of OSTIA to the set T in Fig. 2 yields the transducer shown in Fig. 1(f).

Finally, based on this construction and other considerations, it can be shown that, using this algorithm, the class of *total* subsequential transductions can be identified in the limit (Gold, 1967; Angluin & Smith, 1983) from positive presentation of input-output pairs (Oncina, 1991; Oncina & García, 1991; Oncina *et al.*, 1993). In other words, for any total subsequential transduction OSTIA will exactly obtain the minimum OST that realizes the subsequential transduction from a large enough set of input-output pairs of the function.

3 Onward Subsequential Transducer Inference Algorithm with Domain and Range (OSTIA-DR)

Experimental work using the OSTIA in many applications has clearly shown that very accurate mappings can be obtained with fairly small learned transducers (Oncina, 1991; Oncina *et al.*, 1992; Oncina *et al.*, 1993; Castellanos, Galiano & Vidal, 1994). In the case of *partial* functions (i.e., translation is undefined for certain "wrong" input sentences), the vast majority of *syntactically correct* input sentences were perfectly translated by the learned transducers into correct target sentences in these applications

However, even if perfect output is obtained for proper input, incorrect input sentences (not belonging to the domain of the function) tend to be translated rather disparately. Examples of this behavior will be discussed in detail in subsection 6.2.2 (Tables II and IV). It has been argued that, if some information about the syntax of the input and/or output languages could be supplied to the learning algorithm, the learning strategy of OSTIA could be improved by taking advantage of this information (Oncina *et al.*, 1993). The transducers learned incorporating such an information would exhibit a more reasonable behavior upon not perfectly correct inputs. Instead of obtaining rather disparate translations (even for slight input incorrectness), the transducer could obtain at least an approximately correct translation, or simply an error output message. This situation becomes particularly relevant if rather than having a "clean" *input text* we have to deal with corrupted and distorted signals, as in the case of *hand-written text* or *speech input*.

Apart from these problems, learning partial functions leads to an even more important issue: while identification in the limit of *total* subsequential transductions is guaranteed by OSTIA, the class of *partial* subsequential transductions can not be learned by only using positive presentation. The next example illustrates a particular case of the inability of OSTIA to learn partial subsequential transductions, which yields no convergence in the limit. This is perhaps the most undesirable case for practical applications, since no unseen positive input sentence will ever be able to be translated by such a transducer learned by OSTIA.

Example 1: Let $t : \{a, b, c\}^* \to \{0, 1, 2\}^*$ be a partial subsequential transduction defined by:

$$t = \{(c^m, 2^m) | m \geq 0\} \cup \{(c^m a c^{2n}, 2^m 0^{2n}) | m, n \geq 0\} \cup \{(c^m b c^{2n+1}, 2^m 1^{2n+1}) | m, n \geq 0\}$$

Fig. 4 shows the canonical OST for this function. The OTST of a sample T, which contains all the input-output pairs up to input string length six, is depicted in Fig. 5(a). The transducer learned by OSTIA from this OTST is displayed in Fig. 5(b). It can be observed that the input training sentences which end with c's yield an increasing sequence of edges and states in the learned transducer, which would keep growing as longer examples are being included in the training set. As a result, no convergence can be reached in the limit. Such a transducer is obtained when OSTIA merges a branch of the OTST which represent an input string containing a final *odd* number of c's with another branch containing a final *even* number of input c's. This successive merging of states is possible because the successive input symbols (associated to the edges) match and the corresponding output symbols can be pushed-back up to an edge or state where they are not in contradiction to each other.

(Figure 4 about here)

This problem actually arises from the fact that in the target transducer, the *tails* of two different states (i.e., transductions starting at these states) have domains which do not intersect. In this case, OSTIA has no criterion to forbid the merging of this states. In the transducer τ of Fig. 4, $T_\tau(q) = \{(c^{2n}, 0^{2n}) | n \geq 0\}$ and $T_\tau(q') = \{(c^{2n+1}, 1^{2n}) | n \geq 0\}$, which means that $dom(T_\tau(q)) \cap dom(T_\tau(q')) = \emptyset$. Thus, no positive training pair can exist that would help distinguishing these states. $\square$

(Figure 5 about here)

The general problem of learning partial subsequential transductions is thus related with the possibility of distinguishing pairs of states whose domains are different. Obviously, in order to distinguish such states, additional information not contained in the positive training pairs themselves, must be used. This information can actually be considered *negative* information about what the learning algorithm should not do, and we describe here below a modification of OSTIA which uses a finite state model of the domain of the function to represent this additional information. Moreover, the modified versions of OSTIA, that are introduced in the next subsections, allow for learning transducers that only accept input sentences and/or only produce output sentences compatible with the modeled input (domain) and/or output (range) syntactic constraints. Using only domain constrains lead to the so-called OSTIA-D technique, while using only range constrains results in OSTIA-R. Both techniques can be straightforwardly combined, leading to the so-called OSTIA-DR algorithm.

3.1 Learning Using Domain Information

In many transduction tasks a description of the domain is available or can be inferred from the input strings of the training pairs using appropriate Grammatical Inference techniques (Angluin & Smith, 1983; Miclet, 1990; Vidal, 1994). Since the domain language of a subsequential transduction is regular, we can assume that the minimum *Deterministic Finite Automaton* (DFA) that describes this language (or an approximation thereof) is available. Let us denote this automaton by $D = (Q_D, X, \delta_D, p_0, F_D)$ and let $L(D)$ be the language accepted by D. The use of a DFA representing the domain language can effectively help distinguish states that would be non distinguishable by (positive) translation examples only.

Example 1 above illustrates a concrete case in which two states q and q' cannot be distinguishable by their translations. The general condition for non distinguishability of two states generalizes this case to situations in which the intersection of domains of the tails of states is not empty. If q and q' are *different* states of the target canonical transducer τ, then $T_\tau(q) \neq T_\tau(q')$. And it can be seen that the general condition for these states to be non distinguishable becomes: $dom(T_\tau(q)) \neq dom(T_\tau(q'))$ and $\forall x \in dom(T_\tau(q)) \cap dom(T_\tau(q'))$, $y = y'$, where $(x, y) \in T_\tau(q)$ and $(x, y') \in T_\tau(q')$. In other words, q and q' would be distinguishable by OSTIA if there would exist an input string x belonging to the intersection of the domains of the tails of q and q', such that the output strings, y and y', associated to x in the tails of q and q', respectively, would be different. The following example illustrates this general non distinguishability situation.

Example 2: The canonical OST shown in Fig. 6(a) defines the partial subsequential transduction $t : \{a, b, c\}^* \to \{0, 1, 2, B\}^*$ such that:

$$t = \{\lambda, \lambda\} \cup \{((ac^j bb^k)^n, (02^j 11^k)^n B) | j, k \geq 0 \wedge n \geq 1\} \cup$$
$$\{(b^i (ac^j bb^k)^n, 1^i (02^j 11^k)^n B) | i \geq 1 \wedge j, k, n \geq 0\}$$

In this OST, states q and q' are different; i.e., $T_\tau(q) \neq T_\tau(q')$, where $T_\tau(q) = t$ and

$T_\tau(q') = \{(c^l bb^i (ac^j bb^k)^n, 2^l 11^i (02^j 11^k)^n B) | l,i,j,k,n \geq 0\}$. Then, $dom(T_\tau(q)) = \{b^i (ac^j bb^k)^n | i,j,k,n \geq 0\}$ and $dom(T_\tau(q')) = \{c^l bb^i (ac^j bb^k)^n | l,i,j,k,n \geq 0\}$, which means that $dom(T_\tau(q)) \neq dom(T_\tau(q'))$. But, $\forall x \in (dom(T_\tau(q)) \cap dom(T_\tau(q'))) = \{b^i (ac^j bb^k)^n | i \geq 1 \wedge j,k,n \geq 0\}$, $y = y'$, since $S \subset T_\tau(q)$ and $S \subset T_\tau(q')$, where $S = \{(b^i (ac^j bb^k)^n, 1^i (02^j 11^k)^n B) | i \geq 1 \wedge j,k,n \geq 0\}$. Thus, states q and q' will not be distinguishable by OSTIA. $\square$

(Figure 6 about here)

On the other hand, if D is the minimum DFA such that $L(D) = dom(t)$, then[2] $\forall p,p' \in Q_D, p \neq p' \Leftrightarrow T_D(p) \neq T_D(p')$. Moreover, if $L(D) = dom(t)$, then $\forall q \in Q, \exists p \in Q_D$ such that $dom(T_\tau(q)) = T_D(p)$. Thus, $\forall q,q' \in Q$ such that $dom(T_\tau(q)) \neq dom(T_\tau(q'))$ there exist $p,p' \in Q_D$ $(p \neq p')$ such that $T_D(p) \neq T_D(p')$. Therefore, if the merge of any two states of the transducer τ, like q and q', is forbidden whenever the corresponding states of D, p and p', are different, then the merge of non distinguishable states can be avoided and identification in the limit can be achieved. Obviously, for a state q of $\tau = (Q,X,Y,q_0,E,\sigma)$ such that $(q_0,x_1,y_1,q_1)\ldots(q_{n-1},x_n,y_n,q) \in E^*$, $x = x_1 \ldots x_n$, the corresponding state of D is $p = \delta_D(p_0,x)$.

Note that $q,q' \in Q$, $q \neq q'$, such that $dom(T_\tau(q)) = dom(T_\tau(q'))$ there exists only one state $p \in Q_D$ such that $T_D(p) = dom(T_\tau(q)) = dom(T_\tau(q'))$. But, if $q \neq q'$ in the canonical OST for t, then $T_\tau(q) \neq T_\tau(q')$, which means that there exists $x \in T_D(p)$ such that $(x,y) \in T_\tau(q)$, $(x,y') \in T_\tau(q')$ and $y \neq y'$. The following example illustrates these two ways to distinguish states; that is, with the help of the domain or by the transductions themselves.

Example 3: DFA shown in Fig. 6(b) is the minimum DFA describing $dom(t)$. In this DFA, $T_D(p) = \{b^i (ac^j bb^k)^n | i,j,k,n \geq 0\} = dom(T_\tau(q)) = dom(T_\tau(q''))$ and $T_D(p')) = \{c^l bb^i (ac^j bb^k)^n | l,i,j,k,n \geq 0\} = dom(T_\tau(q'))$. Thus, the state q' is now distinguishable from q and q'' because $p \neq p'$, and q is distinguishable from q'' because

$\lambda \in T_D(p) = dom(T_\tau(q)) = dom(T_\tau(q''))$, but $(\lambda, \lambda) \in T_\tau(q)$ and $(\lambda, B) \in T_\tau(q'')$. $\square$

Based on the above discussion, the new algorithm is shown in Fig. 7. In this algorithm, function *input_prefix*: $Q \to X^*$ is introduced. For each state $q \in Q$ of the *OTST* this function returns the (unique) string $x \in X^*$ that leads from q_0 to q in the *OTST*. The result of $\delta_D(p_0, input\_prefix(q))$ is thus the state of D that is reached with x. This can be computed at no cost by labeling each state q of the *OTST* with the state of D that is reached with x, as previously mentioned. These labels will not change during execution of the algorithm because only states with the same label can be merged.

(Figure 7 about here)

The new algorithm works as the previous version (outlined in the last section), but now an additional condition is tested before every state merging operation: line 9 of the new algorithm tests if the states to be merged are reached with input prefixes which lead to the same states in the input automaton. Only if the condition succeeds, the algorithm continues trying to merge the states of the transducer as it previously did.

It can be easily seen that (if all the input strings of the training pairs are accepted by D) this technique ensures that the domain of the obtained transducer is always included in the language accepted by D, even if D is not minimum or does not describe exactly the domain of the target transducer. Moreover, for any *partial* subsequential transduction t, it can now be shown that if $L(D) = dom(t)$, OSTIA-D identifies t in the limit (Oncina & Varó, 1996).

Example 4 Let t be the partial function defined in Example 1. Let D be the minimum DFA describing its domain language (Fig. 8) and let $T = \{(a, \lambda), (acc, 00), (acccc, 0000), (bc, 1), (bccc, 111), (c, 2)\}$ be a sample drawn from this function.

(Figure 8 about here)

As in the basic OSTIA, the OSTIA-D algorithm begins building the $OTST$ of the training sample but, in this case, the states are labeled with the corresponding state of D (Fig. 9(a)). Then, each state $q \in Q_{OTST} - \{\lambda\}$ has a label $label(q) = \delta_D(label(q'), a)$ where $(q', a, w, q) \in E_{OTST}$ and $label(\lambda) = \lambda$ (Fig. 9(a)).

(Figure 9 about here)

Then, it tries to merge the states λ and a but it fails because they have different labels. Following the lexicographic order, the next pair of states with identical labels are λ and c. They can be merged and the transducer of Fig. 9(b) is obtained. Next, the algorithm, tries to merge the states b and ac (Fig. 9(c)). Since this transducer is not subsequential, the inner loop will try to transform it into a subsequential one by pushing back symbols and merging states. Note that, due to the determinism of D, all the labels of the merged states are always identical and need not be compared. At the end, the transducer of Fig. 9(d) is obtained. This transducer does not fulfills the second **if** condition in the inner most loop. Therefore, it is discarded and the transducer in Fig. 9(b) is recovered.

Following the algorithm, the next successful merges are a with abb (Fig. 9(e)) and b with bcc, leading to the inferred transducer shown in Fig. 9(f). □

3.2 Learning Using Range Information

This last technique can be extended to control the output language too. In many real world tasks it is very important to ensure that the output strings belong to a fixed and known language. For instance, if (possible ungrammatical) English sentences are to be translated into formal queries to access a data base, syntax errors should be carefully avoided in the *output language*. Similarly, when translating from a language into another, well formed output sentences should be obtained.

As in the previous case, the output language of a subsequential transducer can be described by a regular language. Thus, a (minimum) *DFA* describing the range can be available. Let us denote $R = (Q_R, Y, \delta_R, p_0, F_R)$ this automaton. Here again, each state of the $OTST$ can be labeled with the state of R that is reached with the (unique) string v leading from q_0 to q in the $OTST$. Then, if only merges of states with the same label are allowed, the output language will be a sublanguage of R.

The algorithm is presented in Fig. 10, which introduces function *output_prefix*: $Q \rightarrow Y^*$. For each state q, this function returns the (unique) string y leading from q_0 to q in the $OTST$. The result of $\delta_R(p_0, output\_prefix(q))$ can be computed at no cost. Let $y = output\_prefix(q) = y_1 y_2 \ldots y_n$. Each symbol of y can be labeled with a state of R such that: (i) $label(\lambda) = p_0$; and, (ii) $label(y_i) = \delta_R(label(y_{i-1}), y_i)$. Then, $\delta_R(p_0, output\_prefix(q)) = label(y_n)$ and each state of the $OTST$ can be labeled in this way. During the execution of OSTIA-R the labels of the symbols do not change but, since certain symbols of the output strings can be moved from an edge to another by the *push_back* operation, state labels can change; nevertheless, they can be recalculated easily as a by-product of the *push_back* operation.

(Figure 10 about here)

Example 5 Let t be the partial subsequential transduction defined in Example 1. Let the automaton describing the range language be the one shown in Fig. 11, and let $T = \{(a, \lambda), (acc, 00), (acccc, 0000), (bc, 1), (bccc, 111), (c, 2), (cc, 22)\}$ be a sample drawn from this function.

(Figure 11 about here)

As in the previous cases, the algorithm begins building the $OTST$ of the training sample, but in this case the states are labeled with the corresponding states of R. Then, each state $q \in Q_{OTST} - \{\lambda\}$ has a label $label(q) = \delta_R(label(q'), x)$ where $(q', a, w, q) \in E_{OTST}$ and $label(\lambda) = \lambda$ (Fig. 12(a)).

(Figure 12 about here)

The algorithm follows trying the merging of the equally labeled states λ and a (Fig. 12(b)). This transducer has the edges $(\lambda, c, 2, c)$ and $(\lambda, c, 00, ac)$ that violate the *subsequential* condition and the innermost **while** loop is entered. The edges fulfills the first **if** condition of this loop and then the strings "00" and "2" must be *pushed-back* in order to merge the states c and ac (Fig. 12(c)). Note that the state ac can now be accessed from λ with output λ. Thus, the label of this state must be changed to λ (the label of the state λ). Now both states can be merged and, after some additional steps, the transducer in Fig. 12(d) is obtained. Since there are states acc and cc that do not fulfill the second **if** condition of the inner most loop, the transducer in Fig. 12(a) is recovered.

Following the algorithm, the next successful merges are: λ with c (Fig. 12(e)), b with bcc (Fig. 12(f)), and ac with $accc$, leading to the inferred transducer shown in Fig. 12(g). Note that the transducer is not isomorphic with the canonical one, but both realize the same transduction. $\square$

As previously mentioned, both OSTIA-D and OSTIA-R can trivially be combined leading to the so-called OSTIA-DR algorithm.

4 The Visual Scenes Description (VSD) Task

In order to test the capabilities of subsequential transductions, OSTIA and OSTIA-DR, the so-called *Miniature Language Acquisition* (MLA) task (Feldman *et al.*, 1990) has been considered. Although this task is very easy for humans, in its most general formulation it clearly exceeds the capabilities of the current computer learning systems. So, a more specific formulation was provided by Feldman *et al.* (1990) in order to define the scope of the task precisely. As a mere matter of convenience, we have renamed this specific formulation as Visual Scenes Description (VSD), since it conceptually consists of understanding the meaning of pseudo-natural English sentences that describe simple visual scenes.

To implicitly constrain the conceptual domain of the pseudo-English sentences of the VSD task, a simple phrase structure grammar was given by Feldman *et al.* (1990) for specifying the descriptive language. The fact that a grammar is used does not imply that the system should learn exactly these syntactic rules. It is provided only for strictly bounding the set of objects, their attributes and relations, that are allowed by the descriptive language (Feldman *et al.*, 1990). On the other hand, it should be noticed that, although this grammar takes the form of a context-free grammar and the language it defines is very large (as many as $1.6 \cdot 10^8$ sentences), it constitutes in fact a finite —ergo regular— language.

Up to four objects may appear in the scenes of the VSD task, each one having one of three possible shapes (circle, square and triangle) and one of two distinct shades (light and dark). Size and position of the objects can be arbitrary within the image boundaries. But, obviously, in the semantic domain of the VSD task only three different sizes (small, medium and large) and nine relative positions (touch, [far] above, [far] below, [far] to the left, [far] to the right) are taken into account. In addition, objects may not occlude or overlap one another. Fig. 13 shows some scenes along with corresponding descriptive English sentences.

(Figure 13 about here)

The VSD task, as specified by Feldman *et al.* (1990), defines a formal relation between the set of sentences and the set of scenes, but it does not define a partial function from the former into the latter. In other words, the task is ambiguous in the sense that for a given scene there may be more than one descriptive sentences applicable and, also, a descriptive sentence can be consistent with many different scenes. Thus, to define a partial function and, consequently, to remove the ambiguity, the semantic representation that will be introduced in the next section implicitly constrains the task in that a one-to-one correspondence between the set of sentences and the set of semantic representations of the scenes is assumed. Removing the ambiguity is necessary to properly frame the VSD task within our learning paradigm. Stolcke (1990) also made this assumption (and imposed further constraints) to establish a restricted MLA learning task to be approached through Simple Recurrent Networks.

5 Semantic Coding Schemes for the VSD Task

To state the semantic contents of each pseudo-English input descriptive sentence of the VSD task within our transducer learning framework, a *semantic coding scheme* is required for representing the scenes. Moreover, such a coding scheme has to be consistent with the kind of transduction that we are trying to infer; that is to say, it has to be an unambiguous transduction. From this consideration, we have adopted three *logic languages* with a limited number of variables. They are similar in their formulation, but they aim at supplying differences in the difficulty of the transduction to be inferred, with regard to the input-output asynchronies involved; i.e., different number, variability and distance of asynchronies between corresponding words in the input and output sentences.

In the sentences of these semantic languages, up to four variables (x, y, z and w) can appear which represent the four possible objects in a scene. Then, for an object that is in the scene, its possible attributes are represented as unary predicates on the variable which represent the object. A predicate which appears in the sentence means that the corresponding object has this attribute. Unary predicates are $\mathsf{C}(\cdot)$, $\mathsf{S}(\cdot)$ and $\mathsf{T}(\cdot)$ for representing the *shape* (circle, square and triangle) of an object; $\mathsf{Li}(\cdot)$ and $\mathsf{D}(\cdot)$ for the *shade* (light and dark); and $\mathsf{Sm}(\cdot)$, $\mathsf{M}(\cdot)$ and $\mathsf{La}(\cdot)$ for the *size* (small, medium and large). The order in which these predicates may appear in a semantic sentence is, in principle, arbitrary. Nevertheless, taking our purpose of controlling the difficulty of the transductions into account, in the training data they are ordered so as to more or less closely follow the flow of concepts conveyed by the corresponding English sentence. In addition, a connective symbol (&), for joining the unary predicates, and parenthesis, for separating the two parts of a simple relation between objects, are introduced. They have no meaning for the VSD task, but are used to comply with the usual syntax of logic formulae.

We have used two kinds of predicates to define the *relative position* of the objects. In the first semantic language, L1, nine constant predicates (To [touching], A [above], B [below], L [left], R [right], FA [far above], FB [far below], FL [far to the left] and FR [far to the right]) can be used. They appear in the semantic sentence in the same relative position as the corresponding English description does in the input string. Therefore, L1 defines in fact a *pure sequential* transduction task.

For the second and third semantic languages, L2 and L3, respectively, nine binary predicates (To(·,·), A(·,·), B(·,·), L(·,·), R(·,·), FA(·,·), FB(·,·), FL(·,·) and FR(·,·)) are eligible to specify the relative position of the different objects (x and y). From these, up to four can appear in a semantic sentence, due to the fact that a simple relative position relation in the input English sentence can involve up to four paired individual relations between objects.

The difference between L2 and L3 lies in the *ordering* adopted in the training data for these binary predicates with regard to the corresponding English sentences. In L2, a binary predicate involving two objects appears in the sentence as soon as the existence of the two objects in the scene can be *predicted*; i.e., immediately before the first unary predicate on the second object involved in the binary predicate. In contrast, in L3, all possible binary predicates are placed at the end of the string. Table I shows two input English sequences along with the corresponding output semantic sentences in the three languages specified. They clearly illustrate the different difficulty of the mappings in the sense of the input-output asynchronies involved, caused by the different ordering of the predicates for relative positions.

(Table I about here)

6 Learning the VSD Understanding Task

A series of experiments were carried out to test the capabilities of OSTIA and OSTIA-DR for learning to translate VSD English sentences into the corresponding logic semantic description, according to the different semantic coding schemes discussed in the previous section. For this purpose, a *training set* of input-output (English-semantic) pairs is required, from which OSTIA or OSTIA-DR will produce a subsequential transducer, τ. Also, in order to assess the degree to which this transducer accounts for the true transduction underlying the VSD task, an independent *test set* of input-output pairs is required. Let (x, y) be one of these test pairs. The input, English sentence, x, is submitted to transduction by τ, resulting in a semantic sentence $\hat{y} = \tau(x)$. This sentence is then compared with the true semantic description, y, and an error is counted whenever $\hat{y} \neq y$.

The generation of these training and test sets of input-output pairs was governed by the (English) grammar proposed by Feldman *et al.* (1990), which was appropriately augmented in order to supply also the required semantic transductions, according to the different coding schemes. Thus, starting from the axiom, `S`, of this augmented grammar, a random rewriting process was carried out to produce each English sentence along with its corresponding semantic sentence. This process assumed all rules sharing the same left-hand nonterminal to be equiprobable.

Following this procedure, a large set of input-output pairs was initially generated for each of the three semantic coding schemes. Each of these initial sets was further reduced by first removing repeated pairs and then randomly decimating a number of pairs so as to yield a standard set of 120,000 pairs. Each of these sets was used to evaluate the OSTIA and OSTIA-DR performance using a leaving-k-out-like or *cross-validation* procedure (Raudys & Jain, 1991). For this purpose, from each 120,000 pairs set, 6 disjoint training sets of 20,000 pairs each were randomly selected and supplied to OSTIA and OSTIA-DR learning in increasing blocks of 1,000, 2,000,

... up to 20,000 pairs. For each training set, the remaining 5 sets (100,000 samples) were then used as a test set to measure the performance of the successively learned transducers. This process was repeated 6 times, one for each disjoint training set of 20,000 pairs and the obtained results were averaged over the 6 trials.

6.1 OSTIA Learning Experiments

The above experimental protocol was followed with the basic OSTIA technique. In addition, in order to investigate the effect of the *order of presentation* of the training material to OSTIA, an additional group of three experiments (corresponding to each of the three semantic coding schemes) was carried out. In this case, each of the 6 sets of 20,000 training pairs was sorted according to the *length* of the input (English) strings and the same 6 trials as above were then carried out.

The results for the three semantic coding schemes considered are shown in Fig. 14. The left panels of the figure correspond to the *random presentation* and the right ones show the results for the corresponding *length sorted training*. In each case, three curves are presented: error rate, number of edges and number of states of the learned transducers. In all cases, random presentation results in very accurate transducers (less than 1% error rate) learned from less than or about 12,000 training pairs (Fig. 14: (a), (c) and (e)). Moreover, for the L1 scheme (Fig. 14(a)) these very accurate transducers are already obtained with the first block of 1,000 training pairs and "perfect" transducers (0% error rate for all the 6 test sets) are learned starting from 7,000 training pairs in the 6 trials.

(Figure 14 about here)

Length sorted training, on the other hand, generally yields smoother convergence, and larger training sets (near 20,000 pairs) are required for getting the same 1% accuracy with L2 and L3 schemes (Fig. 14: (d) and (f)). Nevertheless, with L1 scheme (Fig. 14(b)), only 2,000 length sorted training pairs were required for OSTIA to start producing transducers with 0% error rates in the 6 trials. This behavior is due to the fact that L1 entails a pure sequential mapping and short sentences tend to convey all the required information about such a mapping. In contrast, L2 and L3 require much longer sentences to show the relation of the English text and the corresponding binary predicates involved.

It should be noted that the number of training pairs required for convergence is quite small in all cases, as compared with the size of the language involved (approximately $1.6 \cdot 10^8$). It is also interesting to note that these results are obtained with very small learned transducers; namely, with less than 10 states in the L1 scheme and less than 50 for the other coding schemes. Obviously, these compact learned transducers imply small memory requirements for representing transductions, which is a clear advantage of the OSTIA technique. Nevertheless, the small transducers learned with OSTIA have also another feature which can not be considered an advantage of the method, as we will see in the next subsection. An example of learned transducer for the L1 scheme is shown in Fig. 15.

(Figure 15 about here)

Finally, some OSTIA timing results are shown in Fig. 16 for the three semantic coding schemes and the two modes of presentation (random and length sorted). These results were obtained using a HP 9000/735 computer and clearly show the high efficiency of OSTIA learning. In particular, none of the 720 transducers learned in the whole experimentation made OSTIA run more than 70 seconds. Both random and length sorted presentations yield rather smooth patterns in which the actual almost *linear* time growth of OSTIA appears clearly. All these practical timing results, along with those presented in (Oncina *et al.*, 1992; Oncina *et al.*, 1993) are far better than the theoretical, rather pessimistic, worst case cubic time complexity bound proposed in (Oncina *et al.*, 1993).

(Figure 16 about here)

6.2 OSTIA-DR Learning Experiments

6.2.1 Decreasing Overgeneralization

As mentioned in the previous section, OSTIA learned transducers tend to be very compact; i.e., small number of states and edges. This is achieved at the expense of an overgeneralization of the partial function nature of the task, that becomes quite pernicious if not exactly correct input sentences are submitted to translation by the learned devices. Examples of such a behavior can be observed in the transducer shown in Fig. 15. From a practical point of view, OSTIA-DR aims at controlling the possible overgeneralization by using information about the domain and/or range for the mapping to be learned. With the main purpose of comparing OSTIA and OSTIA-DR, two series of experiments have been carried out. In the first one, the minimum *exact* DFAs representing the domain language (pseudo-English) and the range languages (L1, L2 and L3 semantic languages) have been used. In the second one, *approximate* models for these languages where automatically obtained from the corresponding input and output sentences of the training pairs using Grammatical Inference techniques. More specifically, these models were *k-testable* (k-TS) automata, which have been shown to be identifiable in the limit from only positive data (García & Vidal, 1990). Statistical extensions of k-testable automata, often called k-grams, are frequently used as Language Models in natural language or speech recognition tasks (Jelinek, 1976). This experimentation with approximate domain/range models aims at showing to which extent approximate models, automatically obtained from the training data, can approach the performance of exact models that are presumably unaffordable in many real world situations.

In both series of experiments, the same 120,000 sample corpus corresponding to the semantic language considered was used as above, split into 6 training sets of 20,000 pairs, and the same cross-validation procedure described in the beginning of this section was also followed for each combination of semantic language and

OSTIA-DR setting. Only random presentation of the training sets was used in all these experiments.

In addition, *negative data* tests have also been carried out. A unique set of 100,000 negative English (domain) sentences was used in the cross-validation tests to measure the degree of over-generalization of the different transducers obtained. This set was generated starting from a set of 120,000 positive sentences that were distorted through a standard probabilistic error model involving random insertion, deletion and substitution of words of the domain (English) alphabet. Then, those distorted sentences which still belonged to the domain language were removed and 100,000 sentences were randomly chosen among the remaining ones. Examples of these negative sentences can be seen in Table II. The parsing of a negative English sentence through a learned transducer accounts as an error if any output string is obtained, which implicitly means that the negative sentence is accepted by the transducer. Otherwise, it is considered a recognition success (i.e., the sentence is rejected).

(Table II about here)

For the first series of experiments, an exact DFA, with only 25 states and 85 edges, of the domain language (English) has been obtained from the context-free grammar supplied by Feldman *et al.* (1990). Also, exact DFAs for each of the three range (logic semantic) languages have been manually built. The sizes of these DFAs have been the following ones: 28 states and 89 edges for L1; 200 states and 489 edges for L2; and 128 states and 240 edges fo L3. In each case, the three possible combinations of OSTIA-DR learning have been analyzed (using domain, range, and both domain and range).

Fig. 17 shows the results of these experiments. Only in the case of L2 the use of the range in OSTIA-DR produces better transduction rates than the original OSTIA for the *positive* data test. For L1, the results are similar in all the cases, and for L3 the use of the domain in OSTIA-DR improves the results as compared with the use of the range, but not with respect to those of the original OSTIA. On the other hand, the introduction of the domain and/or range in learning the transducers dramatically improves the results for the *negative* data test in all the cases. Interestingly, the introduction of the range DFAs significantly reduces the acceptance of the sentences which do not belong to the domain, with regard to the original OSTIA. And, as expected, the use of the domain DFA yields a 0% recognition error rate. Overall, the use of exact domain and range models leads to significantly better learned transducers.

(Figure 17 about here)

With regard to the size of the learned transducers, it is interesting to note that the introduction of the domain DFAs lead to larger transducers than the introduction of the range DFAs for the three semantic languages. More concretely, the larger transducers obtained for L1 by using the domain DFA and the domain and range DFAs have less than 30 states and 90 edges. For L2 and L3, the transducers obtained in these cases have less than 200 states and 800 edges.

For the second series of experiments, different values of k have been used for learning different domain and/or range k-testable automata, since increasing the value of k decreases the generalization of the approximate automata learned from a set of given training samples. These automata were obtained with the k-TSI algorithm (García & Vidal, 1990), followed by a standard minimization procedure that yielded the canonical acceptors for the learned k-TS languages (Hopcroft & Ullman, 1979). Experiments with not minimized k-TS automata were also carried out with results similar to those presented here (Castellanos, Vidal, Varó & Oncina, 1996).

For the output languages L1 and L2, only the results for $k = 4$ (the best) are shown below, while results using three values of k (2, 3 and 4) are shown for L3.

The behavior of L1 and L2 with k set to 2 and 3 is quite similar to that of L3, and these results have been omitted here for the sake of brevity (Castellanos *et al.*, 1996). The three possible combinations for OSTIA-DR have been evaluated for each combination of k-testable automata, by using the same cross-validation procedure above mentioned and the corpus corresponding to L3. Therefore, a total number of 3 cross-validation tests have been carried out for each value of k, each involving the 6 training sets of 20,000 pairs, the corresponding 6 independent test sets of 100,000 positive pairs and the unique set of 100,000 negative sentences. In contrast with the exact DFA experiment, in which each DFA was unique (and previously defined), here each training set of $20,000$ pairs is used first for learning k-TS automata of domain and/or range, by using separately the input and output sentences of the training pairs, and then for learning the transducers, by using the training set of pairs and the corresponding previously learned automata.

Table III summarizes the results averaged on the six training sets of $20,000$ samples for the transducers learned with OSTIA and OSTIA-DR. The increase of the parameter k tends to improve the overall (positive and negative) behavior, though it seems that larger values of k would lead to worse results. The most interesting result of this experiment is that performance (for $k > 2$) is quite similar to that shown by the exact DFAs experiment on L3.

(Table III about here)

6.2.2 Translation of Noisy Input Sentences

Until now, the capabilities of OSTIA and OSTIA-DR with several combinations of domain and/or range automata for learning appropriate transducers for understanding "clean" input sentences have been evaluated; i.e., the input sentences belonged or did not belong to the input (English) language of the target transducer and, if they belonged to this language, they had associated correct output sentences which had to *exactly* match the output sentences produced by the learned transducer. However, from a practical point of view, this may be not a realistic framework. For a (slightly) incorrect input sentence we would rather like the transducer to accept it and produce some output. This output sentence could or could not be very different to the expected one. What would be desirable is that the degree of incorrectness in the output be directly related with the degree of incorrectness in the input sentence.

In order to analyze the capabilities of OSTIA and OSTIA-DR to achieve this goal the following framework has been established. First, for each semantic language, different "clean" transducers have been learned using one of the six $20,000$ samples training sets. Then, different distorted test sets have been generated from an initial independent set of $1,000$ correct input-output pairs, by increasing the degree of distortion of the input sentences using the same procedure outlined in subsection 6.2.1 (Castellanos *et al.*, 1996). Afterwards, the increasingly distorted test sets have been analyzed by each learned transducer through a standard *Error Correcting* (Dynamic Programming) parsing technique based on the Viterbi algorithm (Forney, 1973; Amengual & Vidal, 1995). For each distorted input sentence, this parsing technique obtains a path in the transducer whose input string minimizes the number of insertions, deletions and substitutions needed to produce the distorted input sentence; then, the output sentence associated to this path is produced. Finally, the output sentences obtained in this way are compared with the target output sentences by a standard Levenshtein Distance algorithm (Sankoff & Kruskal, 1983), yielding a measure of the *word error rate* in the output sentences.

For each semantic language, five distorted test sets were obtained from the initial set of $1,000$ input-output pairs by setting five increasing degrees of distortion of the input sentences (10%, 20%, 30%, 40% and 50%). Examples of these distorted sentences are shown in Table II.

For each semantic language, the original OSTIA and the three combinations of OSTIA-DR were learned using 20,000 input-output pairs. The exact DFAs and approximate 2, 3 and 4-testable minimized automata were used as domain, range or both, in learning these transducers. The results of this experiment are summarized in Fig. 18. For each semantic language, the performance of the transducer learned with the original OSTIA is compared with the two best results of the transducers learned with OSTIA-DR; namely, exact DFAs and minimized 4-TS automata of both domain and range. In all the cases not shown, the results were quite similar to those shown in Fig. 18 (Castellanos *et al.*, 1996). The overall results of this experiment clearly show a significantly better behavior of the transducers learned with OSTIA-DR with respect to those learned with OSTIA for tasks in which no clean input sentences can be expected. Some selected examples of semantic sentences obtained by parsing the distorted English sentences of the Table II through transducers learned by OSTIA and OSTIA-DR are shown in Table IV. The overwhelming superiority of the OSTIA-DR learned model is clear in these qualitative results.

(Figure 18 about here)

(Table IV about here)

7 Discussion and Conclusions

Two learning algorithms for subsequential transductions have been described: OSTIA and OSTIA-DR. In previous works, OSTIA was shown capable of learning the class of *total* subsequential transductions (Oncina, 1991; Oncina *et al.*, 1993) and, more recently, OSTIA-DR with an exact DFA of the domain has been shown capable of learning the class of *partial* subsequential transductions (Oncina & Varó, 1996). The application of these learning algorithms to real world tasks still remains to be consolidated. Therefore, in this paper, the capabilities of subsequential transductions and their OSTIA and OSTIA-DR learning have been widely studied for a rather challenging pseudo-natural LU task recently proposed by Feldman *et al.* (1990). Three different semantic coding languages, based on first-order logic formulae, have been defined for this task.

Two main series of experiments have been carried out to assess the capabilities of the learning algorithms. In the first one, the behavior of OSTIA in learning the three transduction tasks derived from the three semantic coding schemes has been studied. For this purpose, two presentation modes of the training data have been considered: completely random data and data sorted by the length of the input strings. Length sorted presentation has appeared to be superior for the simplest, purely sequential, semantic coding scheme (L1). The other coding schemes (L2 and L3) seem to require longer strings to allow for discovering the more intricate associated mappings. Correspondingly, they have shown better performance from random presentation of the training data in which all string lengths have a chance to occur in small training sets. The second series of experiments mainly has dealt with the behavior of the learned transducers with imperfect input sentences. OSTIA-DR has been compared to OSTIA in learning the same three transduction tasks. Exact DFAs and automatically learned k-TS automata have been used for representing the domain and/or range languages of target transductions. As a general result, transduction error rates obtained by OSTIA and OSTIA-DR learned transducers on *perfect* test sentences have been fairly similar, but results for *imperfect* input have been dramatically and consistently better for OSTIA-DR learned transducers.

In general, both OSTIA and OSTIA-DR have consistently proven able to automatically discover the corresponding English-semantic mapping from training sets of input-output pairs that are relatively very small as compared with the size of the pseudo-natural language involved. However, paying a more detailed attention to the size of the training sets with regard to the different mappings, it has appeared for both algorithms that the increase of the difficulty of the mapping has yielded the increase of the size of the training sets required to learn the mapping and also the increase of the size of the transducer learned. In this sense, the results obtained for L2 and L3 suggest, in general, that the number and variability of the input-output asynchronies have shown to increase these sizes as much as the relative distance of the asynchronies between input and output words at least.

An approach to LU based on subsequential transductions allow for using semantic languages which represent actions as rather freely structured compositions of elementary requirements (sentences of a regular language) and which can be designed without taking the understanding network architecture into account. The statistical approach presented by Epstein *et al.* (1996) and Della Pietra *et al.* (1997) also separates semantic representation from system architecture. In approaches like those presented by Gorin *et al.* (1991) and Gorin (1995), the network architecture and the semantic language are strongly related to each other. In this sense, in the experiments presented by Stolcke (1990) on a restriction of the original MLA task (Feldman *et al.*, 1990) and on a particular extension to this restricted task, differences in the results clearly reflected temporary interdependence between the Recurrent Neural Network architecture and the semantic representation selected. On the other hand, immediate advantages of subsequential transductions are that they properly contain sequential transductions, overcoming sequentiality assumptions like those needed in the stochastic approach by Pieraccini *et al.* (1993), and that they can represent all finite transductions (Berstel, 1979).

Transducer learning techniques were already successfully applied to understand spoken VSD Spanish sentences (Jiménez *et al.*, 1994) and to translate written and spoken VSD Spanish sentences into English and German (Oncina *et al.*, 1994; Castellanos *et al.*, 1994; Jiménez, Castellanos & Vidal, 1995; Vilar, Marzal & Vidal, 1995). The series of exhaustive experiments presented here have allowed for confirming the behavior of the learning algorithms observed in all these applications.

Although VSD task is a small and relatively simple task, in principle, the introduction of three semantic languages with meaningfully different sources of transduction difficulties and the availability of a great amount of data have constituted a very useful tool for widely and controlledly studying several practical features of the learning algorithms. Therefore, this experimental study is a required step to better understand the behavior of transducer learning techniques towards their application to real limited domain LU tasks, in particular, and translation tasks, in general.

Within the framework of the European project EuTrans, further works on these learning techniques have demonstrated their applicability to more complex text-to-text and speech-to-speech translation tasks (Amengual *et al.*, 1996a). More concretely, subsequential transducers have been learned which accurately have modeled translations from Spanish into English, German and Italian of the *Traveler* task. The *Traveler* task consists of translating usual sentences that a foreign traveler can say in a hotel reception, from the traveler's language into the receptionist's language (Amengual *et al.*, 1996b). Vocabularies of the four languages are composed of as many as 600 words, and paired sentences of translations present more variability and complexity than those of the VSD task (Amengual *et al.*, 1996b).

With the purpose of reducing the number of training samples required to start to produce accurate transducers, lexical and subsentence categories have been introduced in both learning and translation stages. Categories have reduced the number and variability of the input-output asynchronies, which has yielded subsequential transducers that are obtained from acceptably large sets of training samples and adequately learn the complexity of the *Traveler* task (Amengual *et al.*, 1996a; Amengual *et al.*, 1997c). In addition, the use of Error Correcting parsing and continuous acoustical models have produced the expected improvements in the accuracy of text-to-text and speech-to-speech translation prototypes, respectively (Amengual *et al.*, 1996a; Amengual *et al.*, 1997a; Amengual *et al.*, 1997b). Due to the paradigm adopted, the results of the introduction of categories in the basic algorithms on the *Traveler* (translation) task can be extrapolated for LU tasks with similar complexity.

As a general conclusion from the work presented here, subsequential transducers constitute simple and useful models for representing mappings from an input (natural) language into an output (semantic) language and can be quite accurately learned from sets of paired sentences. The difficulty of the target mapping, in terms of asynchronies between corresponding words in the input and output sentences, affects the number of training samples required to accurately learn the mapping. Since required sizes of training sets for such difficult mappings can be excessively large as compared with the number of examples that should be reasonably expected in a real task, complementary tools able to compensate the lack of samples for difficult mappings are required, like the introduction of categories. This kind of improvements can allow for scaling up basic transducer learning techniques for real limited domain translation tasks and, in particular, for LU tasks, and they are currently under study and development within the framework of the European project EuTrans.

Acknowledgements

The authors wish to thank Dr. North for providing the *"dot"* software (Gansner, Koutsofios, North & Vo, 1993) used to draw the deterministic finite automata and subsequential transducers of Figures 1, 2, 4, 5, 6, 8, 11 and 15. The authors also thank the anonymous reviewers who helped to improve the quality and the presentation of this paper.

This work has been partially supported by the Spanish CICYT, under grant TIC97–0745–C02. Miguel A. Varó is supported by a postgraduate grant from the "Conselleria d'Educació i Ciéncia de la Generalitat Valenciana".

References

Amengual, J. C., Benedí, J. M., Beulen, K., Casacuberta, F., Castaño, A., Castellanos, A., Jiménez, V. M., Llorens, D., Marzal, A., Ney, N., Prat, F., Vidal, E., & Vilar, J. M. (1997). Speech Translation based on Automatically Trainable Finite-State Models. In *Proceedings of the 5th European Conference on Speech Communication and Technology*, Rhodes, Greece, pp. 1439–1442.

Amengual, J. C., Benedí, J. M., Casacuberta, F., Castaño, A., Castellanos, A., Jiménez, V. M., Llorens, D., Marzal, A., Prat, F., Rulot, H., Vidal, E., Vilar, J. M., Delogu, C., Di Carlo, A., Ney, H., Vogel, S., Espejo, J. M., & Freixenet, J. R. (1996). EuTrans: Example-Based Understanding and Translation Systems: First-Phase Project Overview. Technical Report D4, Part1, EuTrans IT-LTR-OS-20268 (Restricted).

Amengual, J. C., Benedí, J. M., Casacuberta, F., Castaño, A., Castellanos, A., Llorens, D., Marzal, A., Prat, F., Vidal, E., & Vilar, J. M. (1997). Error Correcting Parsing for Text-to-text Machine Translation using Finite State Models. In *Proceedings of the Seventh International Conference on Theoretical and Methodological Issues in Machine Translation*, Santa Fe, U.S.A., pp. 135–142.

Amengual, J. C., Benedí, J. M., Casacuberta, F., Castaño, A., Castellanos, A., Llorens, D., Marzal, A., Prat, F., Vidal, E., & Vilar, J. M. (1997). Using Categories in the EuTrans System. In *Proceedings of the Workshop on Spoken Language Translation*, S. Krauwer, D. Arnold, W. Kasper, M. Rayner and H. Somers (eds.), Association for Computational Linguistics and European Network in Language and Speech, Madrid, Spain, pp. 44–53.

Amengual, J. C., Benedí, J. M., Castaño, A., Marzal, A., Prat, F., Vidal, E., Vilar, J. M., Delogu, C., Di Carlo, A., Ney, H., & Vogel, S. (1996). Definition of a

Machine Translation Task and Generation of Corpora. Technical Report D1, EuTrans IT-LTR-OS-20268 (Restricted).

Amengual, J. C., & Vidal, E. (1995). Fast Viterbi decoding with Error Correction. In *Preprints of the VI Spanish Symposium on Pattern Recognition and Image Analysis*, A. Calvo and R. Molina (eds.), Córdoba, Spain, pp. 218–226.

Angluin, D., & Smith, C. H. (1983). Inductive Inference: Theory and Methods. *Computing Surveys*, **15**, 237–269.

Berstel, J. (1979). *Transductions and Context-Free Languages.* Teubner. Stuttgart, Germany.

Castellanos, A., Galiano, I., & Vidal, E. (1994). Application of OSTIA to Machine Translation Tasks. In *Lecture Notes in Artificial Intelligence (862): Grammatical Inference and Applications*, R.C. Carrasco and J. Oncina (eds.), pp. 93–105. Springer-Verlag. Berlin, Germany.

Castellanos, A., Vidal, E., Varó, M. A., & Oncina, J. (1996). Language Understanding and Subsequential Transducer Learning. Technical Report, DSIC II/25/96. *Dpto. Sistemas Informáticos y Computación*, Universidad Politécnica de Valencia. Valencia, Spain.

Della Pietra, S., Epstein, M., Roukos, S., & Ward, T. (1996). Fertility Models for Statistical Natural Language Understanding. In *Proceedings of the 35th Annual Meeting of the Association for Computational Linguistics*, Madrid, Spain, pp. 168-173.

Epstein, M., Papineni, K., Roukos, S., Ward, T., & Della Pietra, S. (1996). Statistical natural language understanding using hidden clumpings. In *Proceedings of the 1996 International Conference on Acoustics, Speech and Signal Processing*, Atlanta, Georgia, U.S.A., pp. 176-179.

Feldman, J. A., Lakoff, G., Stolcke, A., & Weber, S. H. (1990). Miniature Language Acquisition: A touchstone for cognitive science. Technical Report, TR-90-009. *International Computer Science Institute.* Berkeley, California, U.S.A.

Forney, G, D. (1973). The Viterbi algorithm. *Proceedings IEEE* **61**, 268–278.

Gansner, E. R., Koutsofios, E., North, S. C., & Vo, K. P. (1993). A Technique for Drawing Directed Graphs. *IEEE Trans. on Software Engineering* **19**, 214–230.

García, P., & Vidal, E. (1990). Inference of k-testable languages in the strict sense and applications to syntactic pattern recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence* **12**, 920–925.

Gold, E. M. (1967). Language Identification in the Limit. *Information and Control*, **10**, 447–474.

Gorin, A. L. (1995). On automated language acquisition. *Journal of the Acoustical Society of America* **97**, 3441–3461.

Gorin, A. L., Levinson, S. E., Gertner, A. N., & Goldman, E. (1991). Adaptive acquisition of language. *Computer Speech and Language* **5**, 101–132.

Hopcroft, J. E., & Ullman, J. D. (1979). *Introduction to Automata Theory, Languages and Computation.* Addison-Wesley. Massachusetts, U.S.A.

Jelinek, F. (1976). Continuous Speech Recognition by Statistical Methods. *Proceedings of IEEE* **64**, 532–556.

Jiménez, V. M., Vidal, E., Oncina, J., Castellanos, A., Rulot, H., & Sánchez, J. A. (1994). Spoken-Language Machine Translation in Limited-Domain Tasks. In *Proceedings in Artificial Intelligence: CRIM/FORWISS Workshop on Progress and Prospects of Speech Research and Technology*, H. Niemann, R. de Mori and G. Hanrieder (eds.), pp. 262–265. Infix.

Jiménez, V. M., Castellanos, A., & Vidal, E. (1995). Some Results with a Trainable Speech Translation and Understanding System. In *Proceedings of the 1995 International Conference on Acoustics, Speech and Signal Processing*, Detroit, U.S.A., pp. 113-116.

Miclet, L. (1990). Grammatical Inference. In *Syntactic and Structural Pattern Recognition: Theory and Applications*, H. Bunke and A. Sanfeliu (eds.), pp. 237–290. World Scientific.

Oncina, J. (1991). *Aprendizaje de Lenguajes Regulares y Funciones Subsecuenciales* (in Spanish). Ph. D. dissertation, Universidad Politécnica de Valencia. Valencia, Spain.

Oncina, J., & García, P. (1991). Inductive Learning of Subsequential Functions. Technical Report, DSIC II/34/91. *Dpto. Sistemas Informáticos y Computación*, Universidad Politécnica de Valencia. Valencia, Spain.

Oncina, J., & García, P. (1992). Inferring Regular Languages in Polynomial Updated Time. In *Pattern Recognition and Image Analysis*, N. Perez de la Blanca, A. Sanfeliu and E. Vidal (eds.), pp. 49–61. World Scientific Pub.

Oncina, J., García, P., & Vidal, E. (1992). Transducer Learning in Pattern Recognition. In *Proceedings of the 11th IAPR International Conference on Pattern Recognition*, The Hague, The Netherlands, Vol. II, pp. 299–302.

Oncina, J., García, P., & Vidal, E. (1993). Learning Subsequential Transducers for Pattern Recognition Interpretation Tasks. *IEEE Transactions on Pattern Analysis and Machine Intelligence* **15**, 448–458.

Oncina, J., Castellanos, A., Vidal, E., & Jiménez, V. M. (1994). Corpus-Based Machine Translation through Subsequential Transducers. In *Proceedings of the*

Third International Conference on the Cognitive Science of Natural Language Processing, Dublin, Ireland.

Oncina, J., & Varó, M. A. (1996). Using domain information during the learning of a Subsequential Transducer. In *Lecture Notes in Artificial Intelligence (1147): Grammatical Inference. Learning Syntax from Sentences*, L. Miclet and C. de la Higuera (eds.), pp. 301–312. Springer-Verlag. Berlin, Germany.

Pieraccini, R., Levin, E., & Vidal, E. (1993). Learning How To Understand Language. In *Proceedings of the 3rd European Conference on Speech Communication and Technology*, Berlin, Germany, pp. 1407–1412.

Raudys, S. J., & Jain, A. K. (1991). Small Sample Size Effects in Statistical Pattern Recognition: Recommendations for Practitioners. *IEEE Transactions on Pattern Analysis and Machine Intelligence* **13**, 252–264.

Sankoff, D., & Kruskal, J. B. (1983). *Time Warps, String Edits and Macromolecules: the Theory and Practice of Sequence Comparison.* Addison-Wesley. Massachusetts, U.S.A.

Stolcke, A. (1990). Learning Feature-based Semantics with Simple Recurrent Networks. Technical Report, TR-90-015. *International Computer Science Institute.* Berkeley, California, U.S.A.

Vidal, E., García, P., & Segarra, E. (1990). Inductive Learning of Finite-State Transducers for the Interpretation of Unidimensional Objects. In *Structural Pattern Analysis*, R. Mohr, T. Pavlidis, and A. Sanfeliu (eds.), pp. 17–36. World Scientific.

Vidal, E. (1994). Language Learning, Understanding and Translation. In *Proceedings in Artificial Intelligence: CRIM/FORWISS Workshop on Progress and*

Prospects of Speech Research and Technology, H. Niemann, R. de Mori and G. Hanrieder (eds.), pp. 131–140. Infix.

Vilar, J. M., Marzal, A., & Vidal, E. (1995). Learning Language Translation in Limited Domains using Finite-State Models: some Extensions and Improvements. In *Proceedings of the 4th European Conference on Speech Communication and Technology*, Madrid, Spain, pp. 1231–1234.

Footnotes

[1] In the following figures, the name of a state will only be displayed within the state if required.

[2] The concept of tail of a state in a DFA is defined similarly as the concept of tail of a state in a subsequential transducer, which was defined in section 2: given a state p of a DFA D, $T_D(p) = L(D')$, where $D' = (Q_D, X, \delta_D, p, F_D)$.

Table I: Examples of input descriptive sentences accompanied with their transduction in each one of the three semantic languages.

input:	a medium square and a large light triangle are far above a dark circle
output L1:	(M(x) & S(x) & La(y) & Li(y) & T(y)) FA (D(z) & C(z))
output L2:	M(x) & S(x) & La(y) & Li(y) & T(y) & FA(x,z) & FA(y,z) & D(z) & C(z)
output L3:	M(x) & S(x) & La(y) & Li(y) & T(y) & D(z) & C(z) & FA(x,z) & FA(y,z)

input:	a small triangle touches a medium light circle and a large square
output L1:	(Sm(x) & T(x)) To (M(z) & Li(z) & C(z) & La(w) & S(w))
output L2:	Sm(x) & T(x) & To(x,z) & M(z) & Li(z) & C(z) & To(x,w) & La(w) & S(w)
output L3:	Sm(x) & T(x) & M(z) & Li(z) & C(z) & La(w) & S(w) & To(x,z) & To(x,w)

Table II: Examples of English sentences that have been increasingly distorted through an insertion-deletion-substitution error model (D = degree of distortion). These distorted sentences can not be generated by the grammar. Thus, they can properly be considered negative with respect to the input language.

Original :
a dark square touches a large light circle and a large circle
$D = 10\%$:
is dark square touches a large light circle touch and a large circle
$D = 20\%$:
a dark square far touches a large medium circle and a large circle
$D = 30\%$:
square dark square touches a large light circle and a dark large circle
$D = 40\%$:
are square to touches circle large light circle circle light large circle
$D = 50\%$:
dark left square light touches dark a light circle and circle a large circle

Table III: Averaged results for the three semantic languages on six training-sets of $20,000$ samples with different transducers learned with OSTIA and OSTIA-DR, using several combinations of minimized k-TS automata of domain and/or range, with independent test-sets of $100,000$ positive samples and with a test-set of $100,000$ negative samples.

	Learned Transducer		Size: States	Size: Edges	Transduction Error (%)	Negative Error (%)
L1	Original		4	56	0.00	30.74
	$k = 4$	Domain	29	95	0.00	0.00
		Range	21	104	0.01	1.39
		Domain & Range	38	110	0.01	0.00
L2	Original		22	254	0.02	29.40
	$k = 4$	Domain	199	613	0.60	0.02
		Range	126	638	0.40	2.89
		Domain & Range	236	737	0.63	0.00
L3	Original		39	407	0.02	28.35
	$k = 2$	Domain	232	814	0.32	0.03
		Range	59	525	0.32	14.65
		Domain & Range	231	811	0.31	0.02
	$k = 3$	Domain	232	814	0.32	0.03
		Range	144	811	0.32	4.80
		Domain & Range	245	808	0.30	0.00
	$k = 4$	Domain	221	750	0.10	0.02
		Range	167	1052	0.72	5.93
		Domain & Range	271	906	0.27	0.00

Table IV: Examples of L3 semantic sentences obtained by parsing the distorted English sentences of Table II.

Target L3 sentence:			
D(x) & S(x) & La(z) & Li(z) & C(z) & La(w) & C(w) & To(x,z) & To(x,w)			
L3 sentences obtained by a transducer learned by OSTIA:			
D(x) & S(x)	& La(z) & Li(z) & C(z) &	La(w) & C(w) & A(x,z) & A(x,w)	
D(x) & S(x)	& & La(z) & M(z) & C(z) &	La(w) & C(w) & To(x,z) & To(x,w)	
S(x) & D(x) & S(x)	& La(z) & Li(z) & C(z) & D(w) &	La(w) & C(w) & To(x,z) & To(x,w)	
& S(x) & C(x) S(x)	& Li(x) & C(x) & C(x) & Li(x) &	La(x) & C(x)	
D(x) & S(x)	Li(x) & & D(x) & Li(y) & C(y) C(x)	La(z) & C(z) & To(x,z)	
L3 sentences obtained by a transducer learned by OSTIA-DR with minimized 4-testable automata of domain and range:			
D(x) & S(x) & La(z) & Li(z) &	C(z) & La(w)	& C(w) & To(x,z) & To(x,w)	
D(x) & S(x) & La(z) &	C(z) & La(w)	& C(w) & To(x,z) & To(x,w)	
D(x) & S(x) & La(z) & Li(z) &	C(z) & D(w)	& C(w) & To(x,z) & To(x,w)	
S(x) & La(z) & Li(z) &	C(z) & La(w)	& C(w) & To(x,z) & To(x,w)	
S(x) & Li(z) &	C(z) & La(w)	& C(w) & To(x,z) & To(x,w)	

$$t = \{((a^i b^j)^k, (0^i 1^j)^k) | i, j, k \geq 0\}$$

(a)

$$\begin{aligned} t' = \ & \{(\lambda, \lambda)\} \cup \\ & \{((a^i b^j)^k a, (0^i 1^j)^k 0A) | i, j, k \geq 0\} \cup \\ & \{((a^i b^j)^k b, (0^i 1^j)^k 1B) | i, j, k \geq 0\} \end{aligned}$$

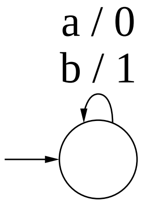
(d)

$$\begin{gathered} t = \{(\lambda, \lambda), (a, 0), (b, 1), (aa, 00), \\ (ab, 01), (ba, 10), (bb, 11), \\ (aaa, 000), (aab, 001), \ldots\} \end{gathered}$$

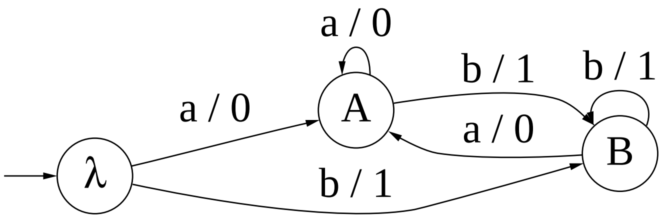
(b)

$$\begin{gathered} t' = \{(\lambda, \lambda), (a, 0A), (b, 1B), (aa, 00A), \\ (ab, 01B), (ba, 10A), (bb, 11B), \\ (aaa, 000A), (aab, 001B), \ldots\} \end{gathered}$$

(e)

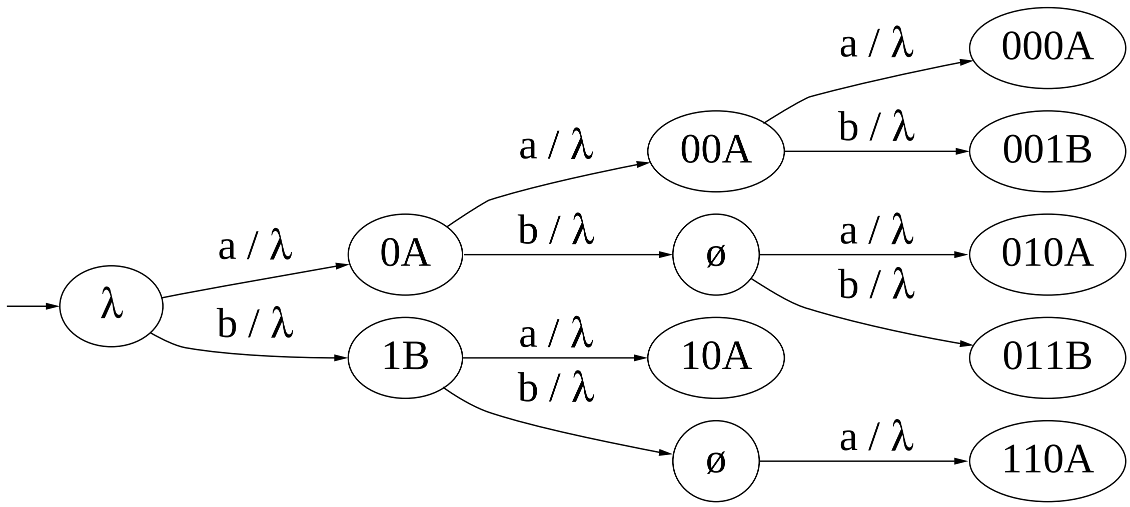


(c)

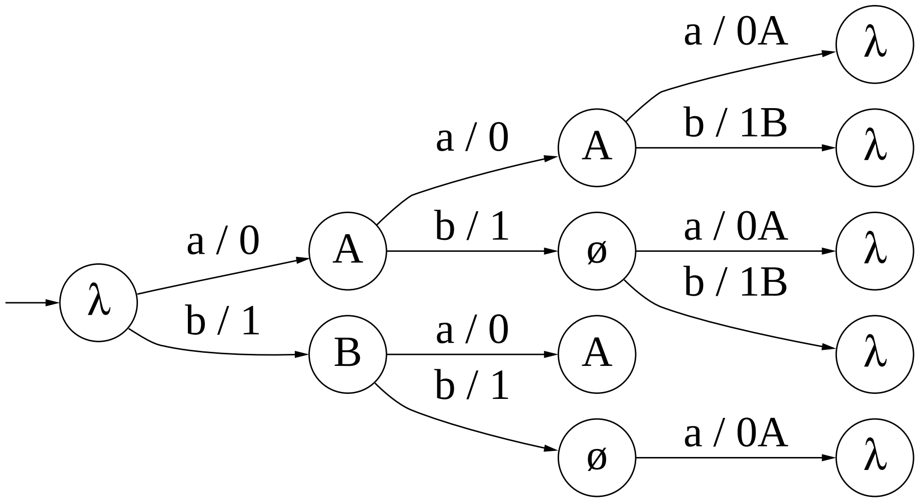


(f)

Figure 1: Examples of sequential and subsequential transductions: (a) A sequential transduction $t : \{a, b\}^* \rightarrow \{0, 1\}^*$; (b) Some pairs of the relation defined by t; (c) A sequential transducer implementing t; (d) A subsequential transduction $t' : \{a, b\}^* \rightarrow \{0, 1, A, B\}^*$; (e) Some pairs of the relation defined by t'; (f) A subsequential transducer implementing t'. Within each state, its associated output symbol is displayed (not the name of the state).



(a)



(b)

Figure 2: (a) Tree Subsequential Transducer (TST) and (b) Onward Tree Subsequential Transducer (OTST) that represent the set of input-output samples $T = \{(\lambda, \lambda), (a, 0A), (b, 1B), (aa, 00A), (ba, 10A), (aaa, 000A), (aab, 001B), (aba, 010A), (abb, 011B), (bba, 110A)\}$. The output string associated to each state is displayed within the state (not the name of the state).

```
Algorithm OSTIA
   INPUT: Single-valued finite set of input-output pairs, T ⊂ (X* × Y*)
   OUTPUT: Onward Subsequential Transducer τ consistent with T
   τ:=OTST(T);
   q:=first(τ);
   while q < last(τ) do
      q:=next(τ,q); q':=first(τ);
      while q' < q do
         if σ(q') = σ(q) or σ(q') = ∅ or σ(q) = ∅ then
            τ':=τ;
            merge(τ,q',q);
            while ¬subsequential(τ) do
               let (r,a,v,s), (r,a,v',s') be two edges of τ
                  that violate the subsequential condition, with s' < s;
               if s' < q and v' ∉ Pr(v) then exitwhile endif
               u:=lcp(v',v);
               push_back(τ,u^{-1}v', (r,a,v',s'));
               push_back(τ,u^{-1}v, (r,a,v,s));
               if σ(s') = σ(s) or σ(s') = ∅ or σ(s) = ∅ then merge(τ,s',s)
               else exitwhile endif
            endwhile //¬subsequential(τ)//
            if ¬subsequential(τ) then τ:=τ' else exitwhile endif
         endif //σ(q') = σ(q)//
         q':=next(τ,q');
      endwhile //q' < q//
   endwhile //q < last(τ)//
end //OSTIA//
```

Figure 3: The Onward Subsequential Transducer Inference Algorithm.

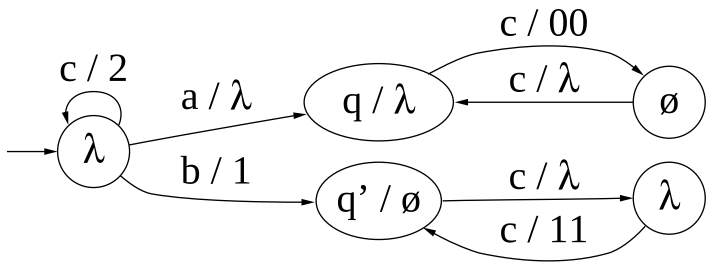
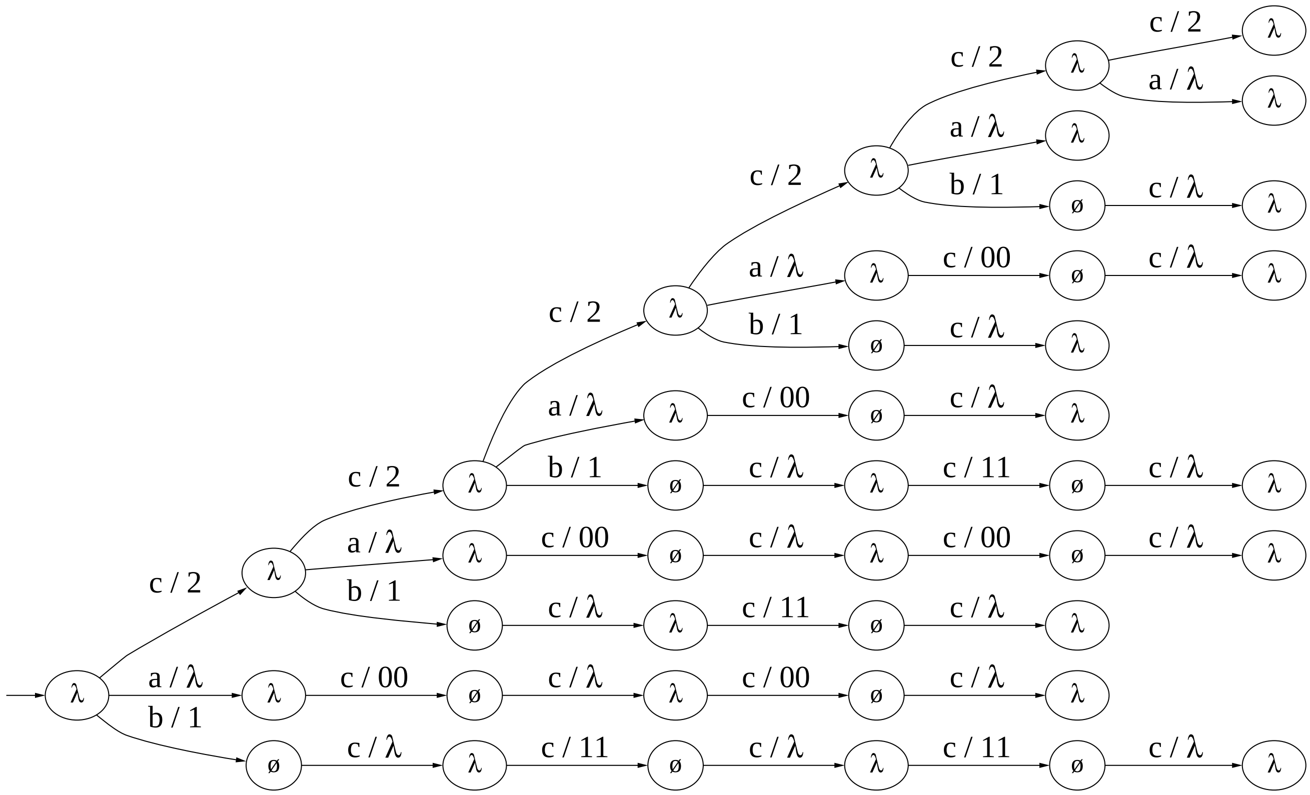
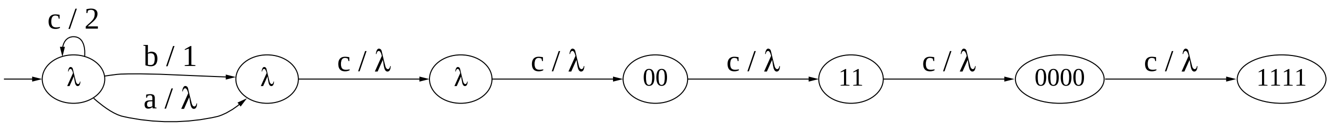


Figure 4: A subsequential transducer for the function of Example 1. No transduction example can help distinguish the two states q and q'.

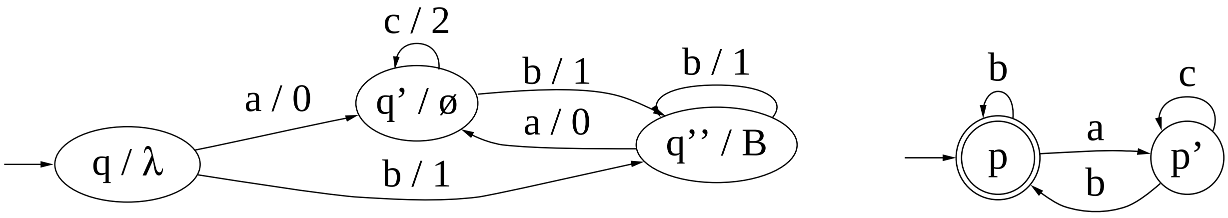


(a)



(b)

Figure 5: (a) OTST of a sample T of the function of Example 1 which contains all the input-output pairs that can be generated up to an input length of 6. (b) Transducer yield by OSTIA from this OTST.



(a) (b)

Figure 6: (a) Canonical OST implementing the partial subsequential transduction of Example 2. (b) Minimum DFA describing the domain of the partial subsequential transduction of Example 2.

```
Algorithm OSTIA-D
  INPUT: Single-valued finite set of input-output pairs, T ⊂ (X* × Y*)
            Deterministic Finite Automaton modeling the Domain language, D
  OUTPUT: Onward Subsequential Transducer τ consistent with T and D
  τ:=OTST(T);
  q:=first(τ);
  while q < last(τ) do
    q:=next(τ,q); q':=first(τ);
    while q' < q do
      if δ_D(p_0, input_prefix(q')) = δ_D(p_0, input_prefix(q)) then
        if σ(q') = σ(q) or σ(q') = ∅ or σ(q) = ∅ then
          τ':=τ;
          merge(τ,q',q);
          while ¬subsequential(τ) do
            let (r,a,v,s), (r,a,v',s') be two edges of τ
              that violate the subsequential condition, with s' < s;
            if s' < q and v' ∉ Pr(v) then exitwhile endif
            u:=lcp(v',v);
            push_back(τ,u^-1 v', (r,a,v',s'));
            push_back(τ,u^-1 v, (r,a,v,s));
            if σ(s') = σ(s) or σ(s') = ∅ or σ(s) = ∅ then merge(τ,s',s)
            else exitwhile endif
          endwhile //¬subsequential(τ)//
          if ¬subsequential(τ) then τ:=τ' else exitwhile endif
        endif //σ(q') = σ(q)//
      endif //δ_D//
      q':=next(τ,q');
    endwhile //q' < q//
  endwhile //q < last(τ)//
end //OSTIA-D//
```

Figure 7: Onward Subsequential Transducer Inference Algorithm with Domain structural information.

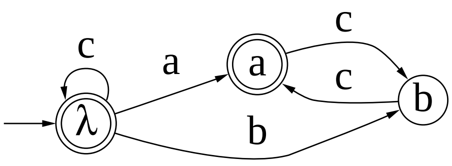


Figure 8: Minimum DFA describing the domain of the partial subsequential transduction of Example 1. Each state is named with the shortest string that reaches it. These names are used for the labels associated to the states of the subsequential transducers of Fig. 9.

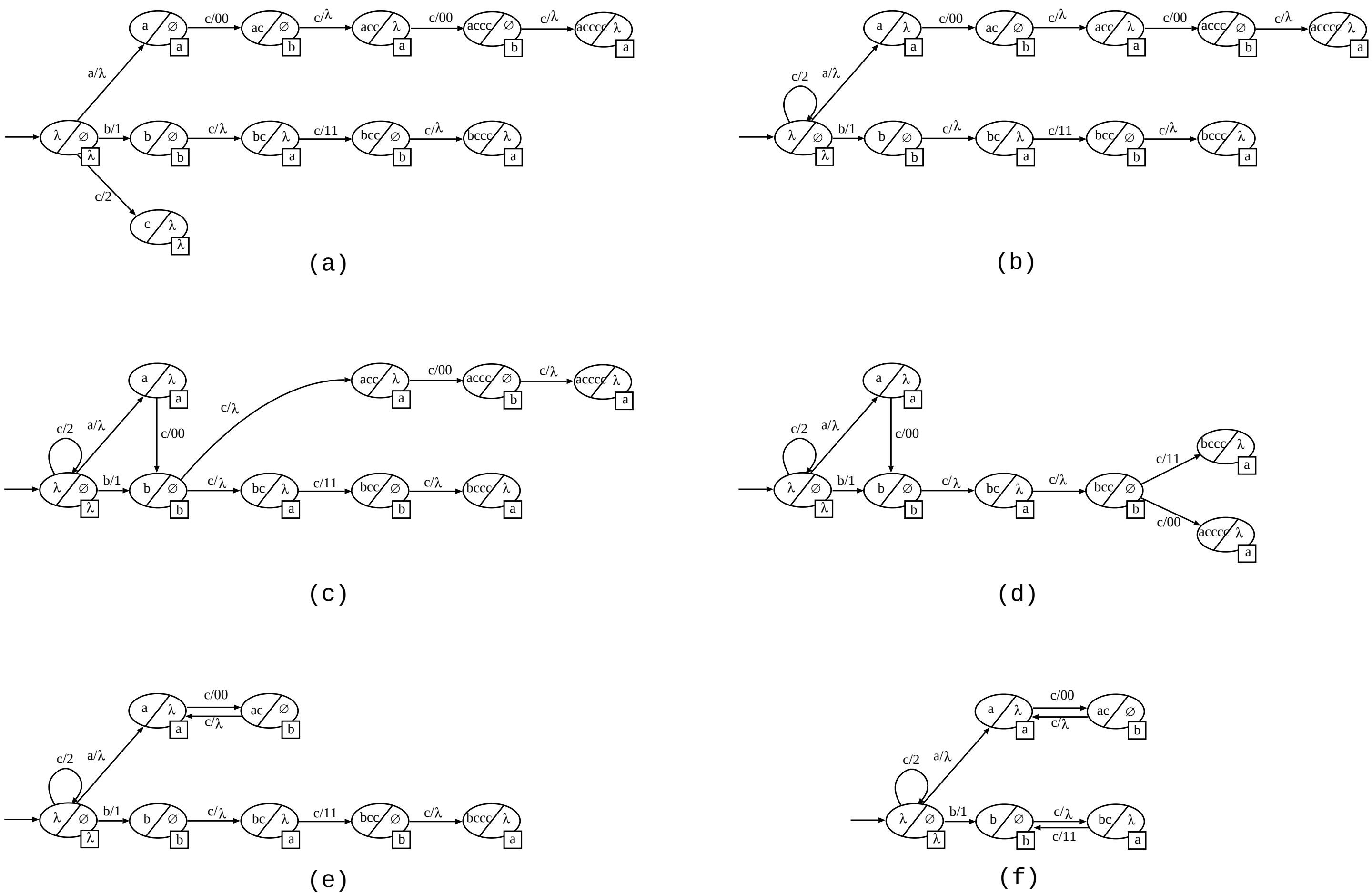


Figure 9: Key steps of the OSTIA-D algorithm as applied to the training set $T = \{(a, \lambda), (acc, 00), (acccc, 0000), (bc, 1), (bccc, 111), (c, 2)\}$.

Algorithm OSTIA-R

INPUT: Single-valued finite set of input-output pairs, $T \subset (X^* \times Y^*)$
Deterministic Finite Automaton modeling the Range language, R

OUTPUT: Onward Subsequential Transducer τ consistent with T and R

τ:=OTST(T);

q:=$first(\tau)$;

while q $<$ $last(\tau)$ **do**

  q:=$next(\tau$,q); q':=$first(\tau)$;

  while q' $<$ q **do**

    if $\delta_R(\mathrm{p}_0, output\_prefix(\mathrm{q'})) = \delta_R(\mathrm{p}_0, output\_prefix(\mathrm{q}))$ **then**

      if σ(q') = σ(q) **or** σ(q') = $\emptyset$ **or** σ(q) = $\emptyset$ **then**

        τ':=τ;

        $merge(\tau$,q',q);

        while $\neg subsequential(\tau)$ **do**

          let (r,a,v,s), (r,a,v',s') be two edges of τ

            that violate the *subsequential* condition, with s' $<$ s;

          if s' $<$ q **and** v' $\notin Pr$(v) **then exitwhile endif**

          u:=lcp(v',v);

          $push\_back(\tau$,u^{-1}v', (r,a,v',s'));

          $push\_back(\tau$,u^{-1}v, (r,a,v,s));

          if σ(s') = σ(s) **or** σ(s') = $\emptyset$ **or** σ(s) = $\emptyset$ **then** $merge(\tau$,s',s)

          else exitwhile endif

        endwhile //$\neg subsequential(\tau)$//

        if $\neg subsequential(\tau)$ **then** τ:=τ' **else exitwhile endif**

      endif //σ(q') = σ(q)//

    endif //δ_R//

    q':=$next(\tau$,q');

  endwhile //q' $<$ q//

endwhile //q $<$ $last(\tau)$//

end //OSTIA-R//

Figure 10: Onward Subsequential Transducer Inference Algorithm with Range structural information.

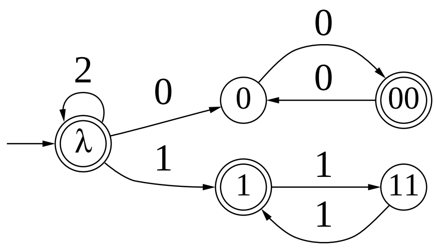


Figure 11: Minimum DFA describing the range of the partial subsequential transduction of the Example 1. Each state is named with the shortest string that reaches it. These names are used for the labels associated to the states of the subsequential transducers of Fig. 12.

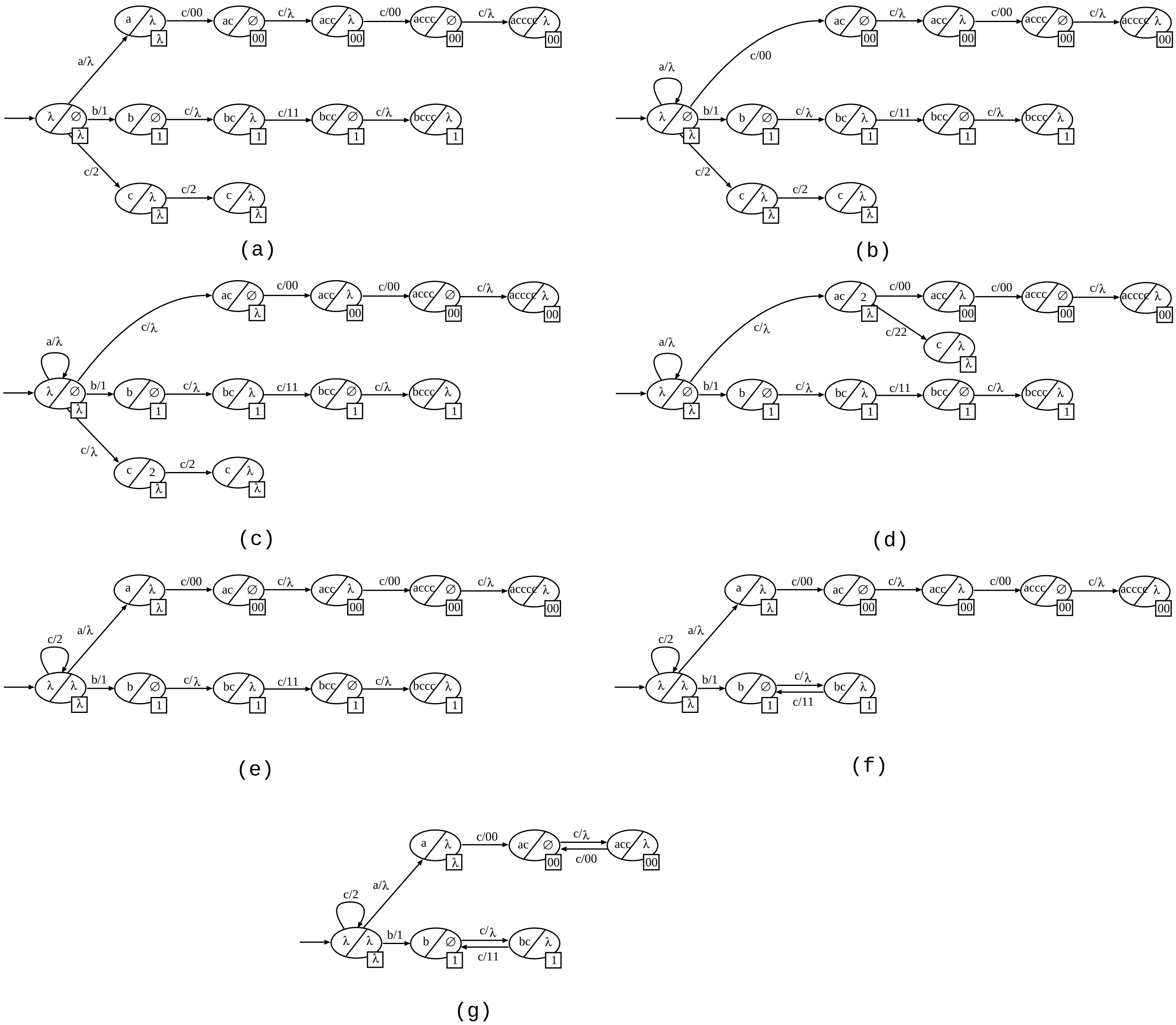


Figure 12: Key steps of the OSTIA-R algorithm as applied to the training set $T = \{(a, \lambda), (acc, 00), (acccc, 0000), (bc, 1), (bccc, 111), (c, 2), (cc, 22)\}$.

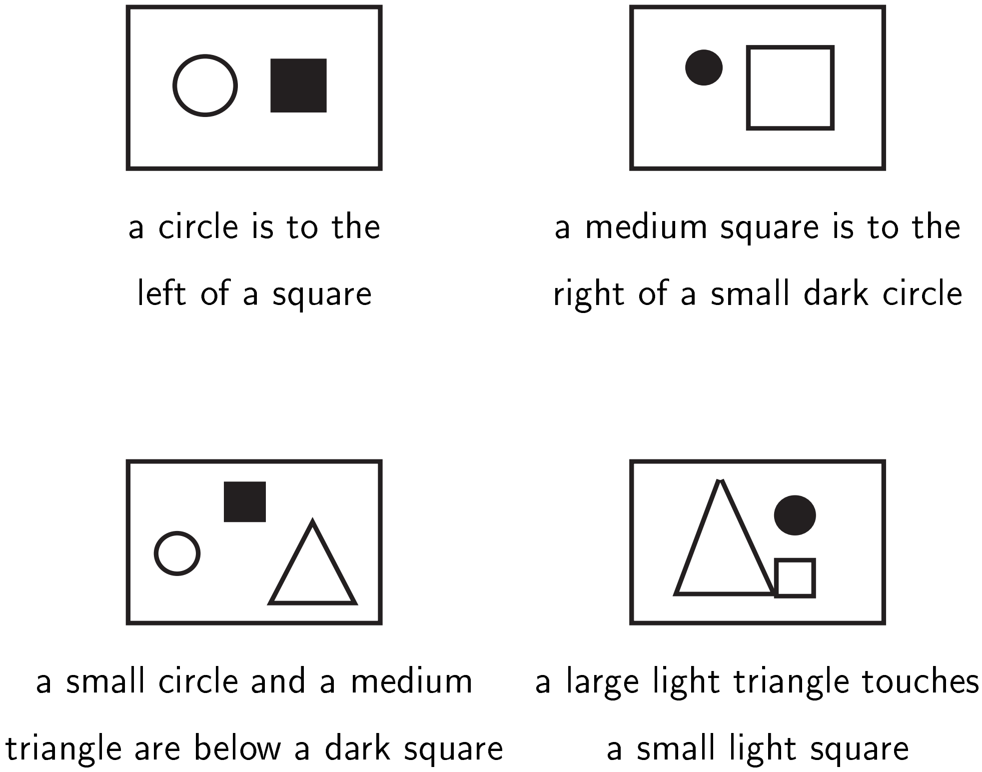


Figure 13: Some scenes and descriptive sentences of the VSD task.

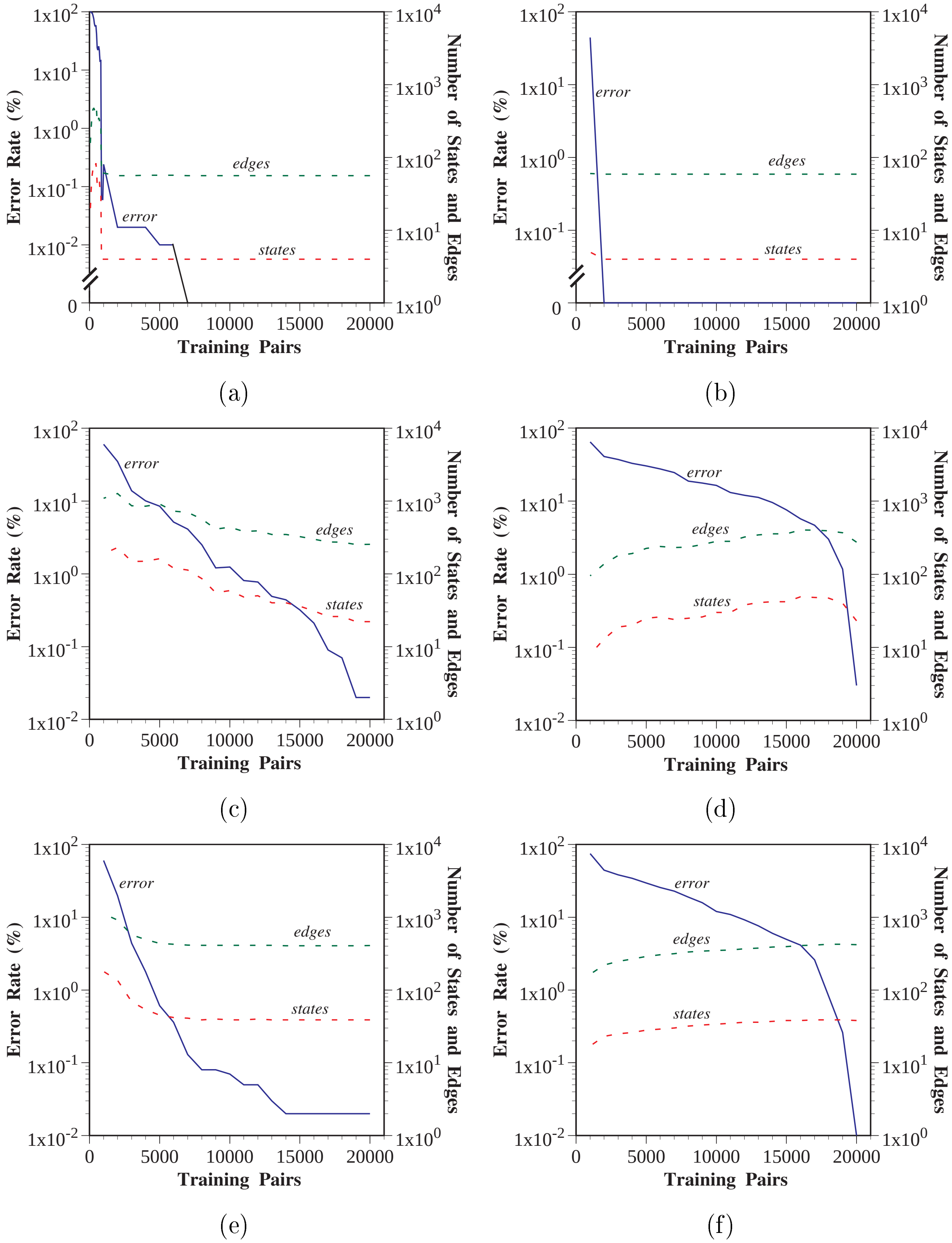


Figure 14: Evolution of the average error rate, number of edges and states of OSTIA learned transducers for the three semantic coding schemes. Left charts: Random presentation of the training. Right charts: Length sorted presentation of the training. From top to bottom: semantic languages L1, L2 and L3.

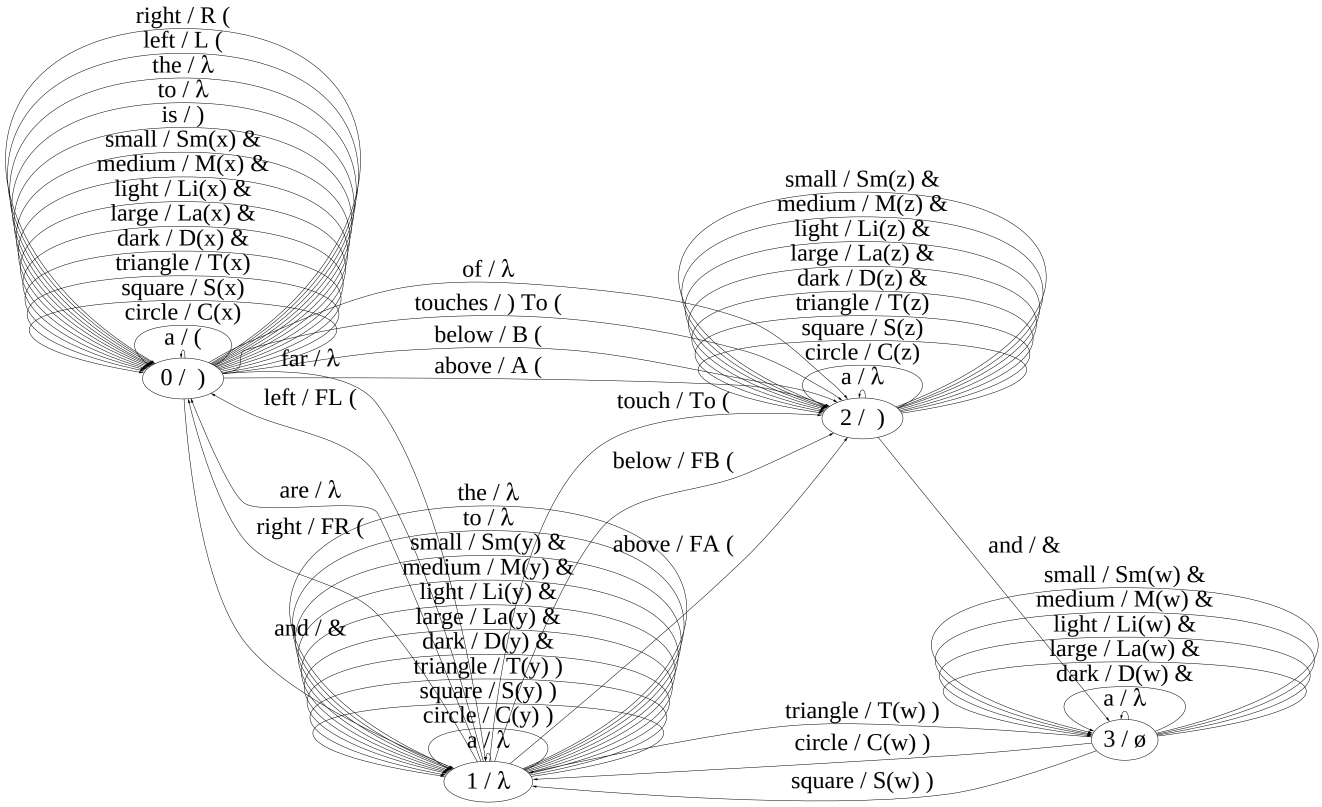


Figure 15: An example of a transducer learned by OSTIA for the VSD task with L1 coding scheme.

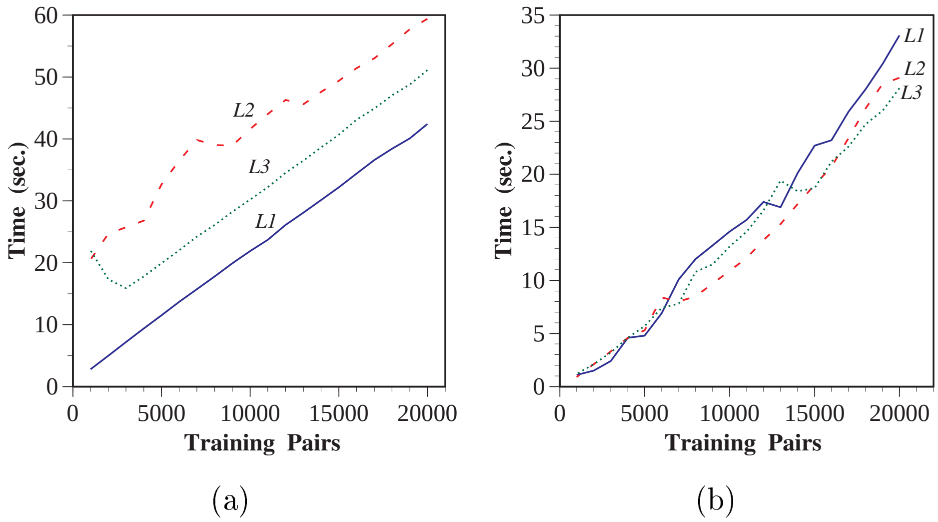


Figure 16: Evolution of the average time required by OSTIA for learning the three semantic coding schemes. (a) Random presentation of the training. (b) Length sorted presentation of the training.

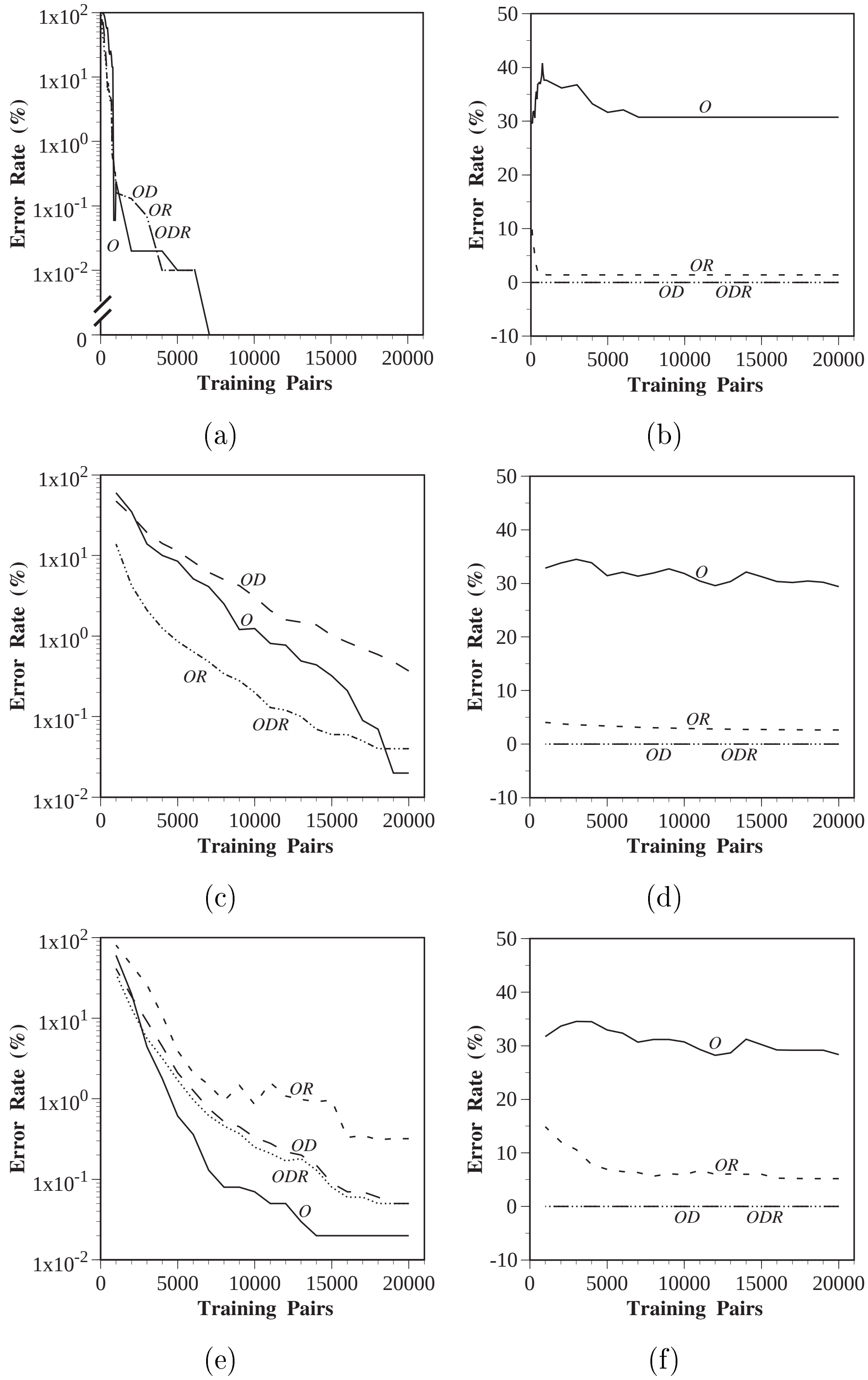
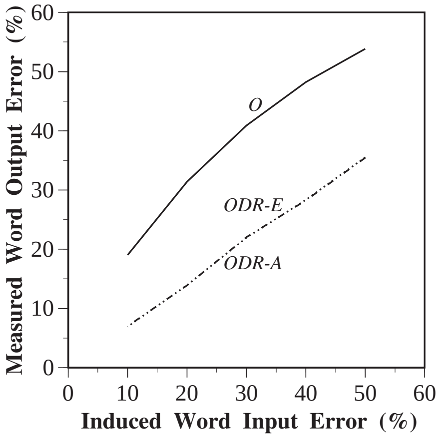
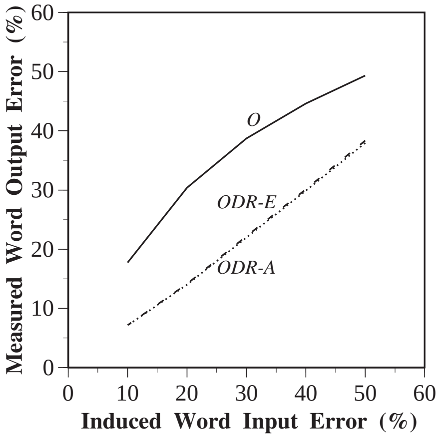


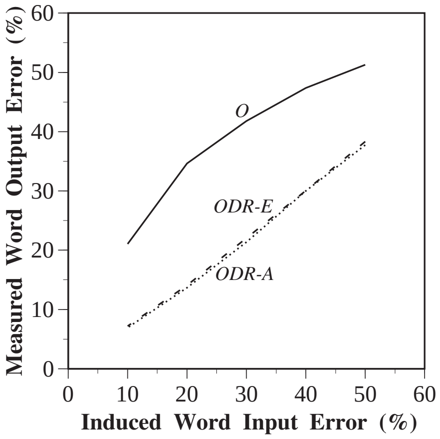
Figure 17: Evolution of the average error rates in the three semantic languages for transducers learned by OSTIA (O) and by OSTIA-DR with exact DFAs of domain (OD), range (OR) and domain and range (ODR). Left charts: Positive transduction error rates. Right charts: Negative recognition error rates. From top to bottom: semantic languages L1, L2 and L3.



(L1)



(L2)



(L3)

Figure 18: Comparative behavior between a transducer learned by OSTIA (O) and another two learned by OSTIA-DR with exact DFAs (ODR-E) and with approximate minimized 4-testable automata (ODR-A) of domain and range in preserving the word error rate of the transductions when they recognize incorrect sentences through an Error Correcting parsing.