In *Topics in Grammatical Inference*.
Jeffrey Heinz and Jose Sempere, eds.
Springer-Verlag. 2016.

Chapter 2
Efficiency in the Identification in the Limit Learning Paradigm

Rémi Eyraud, Jeffrey Heinz and Ryo Yoshinaka

Abstract The most widely used learning paradigm in Grammatical Inference was introduced in 1967 and is known as *identification in the limit*. An important issue that has been raised with respect to the original definition is the absence of efficiency bounds. Nearly fifty years after its introduction, it remains an open problem how to best incorporate a notion of efficiency and tractability into this framework. This chapter surveys the different refinements that have been developed and studied, and the challenges they face. Main results for each formalization, along with comparisons, are provided.

2.1 Introduction

2.1.1 The Importance of Efficiency in Learning

Gold [24] introduced in the 1960s a definition of learning called *identification in the limit*, which works as follows. An algorithm is fed with an infinite sequence of data exemplifying a target language. When a new element is given to the algorithm, it may output a hypothesis. The algorithm identifies the language in the limit if for any possible sequence of data for this language, there exists a moment from when the algorithm does not change its hypothesis, and this hypothesis is a correct representation of the target language. When a whole class of languages is considered, the algorithm identifies the class in the limit if it can identify all languages of the class.

R. Eyraud (✉)
QARMA Team, Laboratoire d'Informatique Fondamentale, Marseille, France
e-mail: remi.eyraud@lif.univ-mrs.fr

J. Heinz
University of Delaware, Newark, DE, USA
e-mail: heinz@udel.edu

R. Yoshinaka
Graduate School of Informatics, Kyoto University, Kyoto, Japan
e-mail: ry@i.kyoto-u.ac.jp

J. Heinz and J.M. Sempere (eds.), *Topics in Grammatical Inference*,
DOI 10.1007/978-3-662-48395-4_2

The fact that the convergence is required to hold whatever the sequence of data is what makes this paradigm adversarial [14]. This worst-case scenario principle strengthens the value of any algorithmic idea that yields an identification in the limit result for a class of languages [25].

However, Gold's formulation can be of little help for practical purposes, when one wants to study a learning idea with the aim of applying it to real-world data. This is mainly due to the fact that no efficiency property is required and thus one can assume infinite time and space. This is the reason why several refinements of Gold's model which add polynomial bounds to the requirements of the paradigm have been developed. The purpose of this chapter is to comprehensively review the proposed refinements and the challenges they face. Main results of each approach, along with comparisons, are provided.

Instead of augmenting the learning framework to incorporate a notion of efficiency, one response to this state of affairs could be to utilize a different learning framework altogether, preferably one which contains a built-in notion of efficiency, such as the Probably Approximately Correct framework [43]. Section 2.2 discusses some issues with PAC-learning of formal languages, which makes this option less attractive than it otherwise may appear at first.

Section 2.3 studies the limitations of the initial identification in the limit definition and previous attempts to overcome them. These include requirements based on the running time of the studied algorithm. Efficiency requirements depending on the incremental behavior of the algorithm, and a set-based refinement of Gold's paradigm are detailed in Sect. 2.4. Finally, Sect. 2.5 introduces two recent reformulations of the paradigm.

2.1.2 Preliminary Definitions

An *alphabet* Σ is a finite non-empty set of symbols called *letters*. A *string* w over Σ is a finite sequence $w = a_1 a_2 \dots a_n$ of letters. Let $|w|$ denote the length of w. Given a set of strings S, we denote $|S|$ its cardinality and $\|S\|$ its size, i.e. the sum of $|S|$ with the lengths of the strings S contains.[1] In the following, letters will be indicated by $a, b, c, \dots$, strings by $u, v, \dots, z$, and the empty string by λ. Let Σ^* be the set of all strings and Σ^+ the set $\Sigma^* \setminus \{\lambda\}$.

We assume a fixed but arbitrary total order $<$ on the letters of Σ. As usual, we extend $<$ to Σ^* by defining the *hierarchical order* [33], denoted by $\lhd$, as follows:

$$\forall w_1, w_2 \in \Sigma^*, w_1 \lhd w_2 \textit{ iff } \begin{cases} |w_1| < |w_2| \text{ or} \\ |w_1| = |w_2| \text{ and } \exists u, v_1, v_2 \in \Sigma^*, \exists a_1, a_2 \in \Sigma \\ s.t.\ w_1 = u a_1 v_1, w_2 = u a_2 v_2 \text{ and } a_1 < a_2. \end{cases}$$

[1] We define $\|S\| = |S| + \sum_{w \in S} |w|$ so that $\|\{a\}\| < \|\{\lambda, a\}\|$.

$\lhd$ is a total strict order over Σ^*, and if $\Sigma = \{a, b\}$ and $a < b$, then $\lambda \lhd a \lhd b \lhd aa \lhd ab \lhd ba \lhd bb \lhd aaa \lhd \dots$

We extend this order to non-empty finite sets of strings: $S_1 \lhd S_2$ iff $\|S_1\| < \|S_2\|$ or $\|S_1\| = \|S_2\|$ and $\exists w \in S_1 \setminus S_2$ such that $\forall w' \in S_2 \setminus S_1$, $w \lhd w'$. For instance $\{a\} \lhd \{\lambda, a\}$ and $\{a, b\} \lhd \{aaa\}$.

By a language L over Σ we mean any set $L \subseteq \Sigma^*$. Many classes of languages were investigated in the literature. In general, the definition of a class $\mathbb{L}$ relies on a class $\mathbb{R}$ of abstract machines,[2] here called *representations*, that characterize all and only the languages of $\mathbb{L}$. The relationship is given by the *naming function* $\mathscr{L} : \mathbb{R} \to \mathbb{L}$ such that: (1) $\forall R \in \mathbb{R}, \mathscr{L}(R) \in \mathbb{L}$ and (2) $\forall L \in \mathbb{L}, \exists R \in \mathbb{R}$ such that $\mathscr{L}(R) = L$. Two representations R_1 and R_2 are *equivalent iff* $\mathscr{L}(R_1) = \mathscr{L}(R_2)$.

Many different classes of representations have been studied in the literature. It is beyond the scope of this chapter to exhaustively list them. However, we introduce the following definition, which is a generalization of some well-known classes of grammars. We will mainly focus on the classes of representations whose characterization can be done in this context.

Definition 2.1 (*Generative grammar*) $G = \langle \Sigma, N, P, I \rangle$ where Σ is the alphabet of the language, N is a set of variables usually called non-terminals, $P \subset (N \cup \Sigma)^+ \times (N \cup \Sigma)^*$ is the set of generative (production) rules, I is the finite set of axioms, which are elements of $(\Sigma \cup N)^*$.

A generative rule (α, β) is usually denoted $\alpha \to \beta$. It allows the rewriting of elements of $(\Sigma \cup N)^*$ into elements of $(\Sigma \cup N)^*$. Given $\gamma \in (\Sigma \cup N)^*$ we say that a production rule $\alpha \to \beta$ applied to γ if it exists $\eta, \delta \in (\Sigma \cup N)^*$ such that $\gamma = \eta\alpha\delta$. The result of applying this rule on γ is $\eta\beta\delta$. We write $\gamma \Rightarrow \eta\beta\delta$. $\Rightarrow^*$ is the reflexive and transitive closure of $\Rightarrow$, and $\Rightarrow_P^*$ is the reflexive and transitive closure of $\Rightarrow$ restricted to the production rules in P.

We define the size of a generative grammar to be the size of the set of its rules, plus the size of its set of axioms: $\|G\| = \|I\| + |P| + \sum_{\alpha \to \beta \in P} (|\alpha\beta| + 1)$.

Definition 2.2 (*Generated language*) Let $G = \langle \Sigma, N, P, I \rangle$ be a generative grammar. $\mathscr{L}(G) = \{w \in \Sigma^* : \exists \alpha \in I \ s.t.\ \alpha \Rightarrow_P^* w\}$.

Example 2.1 The usual classes of the Chomsky hierarchy are classes of generative grammars. Regular grammars correspond to the restriction $P \subset N \times (\Sigma N \cup \{\lambda\})$, or $P \subset N \times (N\Sigma \cup \{\lambda\})$ by symmetry. The context-free grammars are the ones where $P \subset N \times (\Sigma \cup N)^*$ while the context-sensitive grammars are the ones such that if $\alpha \to \beta \in P$ then $\exists (\gamma, \delta, \eta) \in (\Sigma \cup N)^*$, $A \in N$: $\alpha = \delta A \eta$ and $\beta = \delta\gamma\eta$. If no restrictions are imposed on the rules of the grammar, then the resulting class of representations corresponds to that of the unrestricted grammars. All of these classes were formerly defined with a set of axioms reduced to one element of N [11].

[2] This is not strictly necessary: for instance, the substitutable languages [13] have no grammatical characterization.

Example 2.2 String Rewriting Systems (SRS) [9] are generative devices where $N = \emptyset$. A rule corresponds to an element of Σ^* rewritten into an element of Σ^* and the set of axioms is made of elements of Σ^*. The language represented by an SRS is the set of strings that can be rewritten using the rules from an element of I.

Some classes of representations that have been studied in grammatical inference are not covered by Definition 2.1. This is the case for instance for multiple context-free grammars [39], patterns [2], tree [17] and graph [37] grammars, etc. While it is not difficult to generalize the definition in order to cover these classes, we conduct our discussion in the context of the above definition for concreteness and due to its familiarity.

2.2 PAC Learning and Other Learning Paradigms

2.2.1 PAC Paradigm

The best known paradigm in machine learning is certainly the Probably Approximately Correct (PAC) criterium [43] and its refinements [29, 30]. Unlike the identification in the limit paradigm, the PAC framework comes with built-in efficiency requirements so PAC-learners are efficient in important senses. A natural question then is: Why modify the identification in the limit paradigm when the PAC framework can be utilized instead? We argue that the PAC paradigm is not well-adapted to learning formal languages, as even very simple and well-characterized classes of languages are not PAC-learnable [4]. Several theoretical reasons explain this inadequacy, and each of them relates to aspects of the formal grammars used to describe formal languages.

One of the main reasons is that the VC-dimension of even the simplest models of language representations, namely the finite state automata, is not bounded [28] which make them not learnable in the PAC sense [8]. Indeed, not even the class of finite languages has finite VC-dimension. This is closely related to the fact that the learning principle of empirical risk minimization [44], inherent in most approaches studied under the PAC framework, is of little use when formal languages are considered. Indeed, the number of representations *consistent* to a given set of data of a target language, that is to say representations that correctly explain all the data, is often infinite. It is then useless to reduce the hypothesis space to the ones that minimize the error on a given set of data.

Similarly, consider the fact that the PAC paradigm does not suffer from the main drawback of identification in the limit of being asymptotic. Unlike PAC learning, in identification in the limit, there is no guarantee provided about the quality of the hypothesis before the (exact) convergence happens. But this drawback seems to be inherent to the kind of representations of the learning targets considered. Even if two generative grammars have all but one of their rules in common, the languages of these two grammars can be as far apart as one wishes. This problem is inherent to the

nature of formal languages and their grammatical representations. This '*Gestalt*-like' property is unavoidable in the formalization of learning: the whole grammar is more than the sum of its rules. In our view, this mainly justifies the use of identification in the limit in the context of grammar learning.

Another reason is that a representation of a formal language is not only a classifier, that is to say a device that defines what is in the language and what is not, but it also gives *structural* information about the elements of the language.

Also, there are concerns that are more independent of the representations. Another particularity of language learning is that a lot of algorithms use only positive examples of a target concept, while the usual machine learning framework relies on labeled data.

Finally, the PAC paradigm is particularly pertinent in the case of statistical models, where the probability of making a mistake can be evaluated using the hypothesis. This particular attribute of the PAC paradigm is of less value when non-stochastic model learning is of interest. But even while grammatical inference is concerned with learning probability distributions over strings, the power of the considered models makes the paradigm inadequate: there are for instance infinitely many structurally different probabilistic context-free grammars that define the same set of distributions [26].

To be complete, some positive learning results exist in restrictive versions of the PAC-paradigm, mostly in the case where the target distribution is known to be drawn using a given class of stochastic grammars, and with additional restrictions that allow us to distinguish the different parts of the target from any sample (see [15, 36, 41] for examples).

2.2.2 Other Learning Paradigms

There are other less known learning frameworks which eschew identifiability in the limit in order to incorporate notions of computational efficiency. The aim here is not to give an exhaustive list of such paradigms: we just want to give pointers to the main ones.

The first that is worth mentioning is known as *query learning* in which the learner interacts with an oracle (see Chap. 3, *Learning Grammars and Automata with Queries*, de la Higuera). A wide range of types of queries have been investigated, from membership queries [31] where the oracle answers whether given strings belong to the language or not, to equivalence queries [3] that allow the learner to know if its current hypothesis is the target one, including correction queries [6] that correspond to membership queries where the oracle returns a 'close' element of the language if the submitted string is not part of the target (different definitions of string distance can be considered). In this approach, efficiency is measured by the number of queries the algorithm needs to converge to a hypothesis exactly equivalent to the target. Another learning paradigm derived from the former one requires access to a finite set of examples of the language and a membership oracle [16, 34].

Although these paradigms can be of practical interest (see the work on model checking for instance [27]), and though they can also be motivated by the study of first language acquisition [14], the need for an oracle clearly reduces the practical value of an algorithm investigated in this context.

Another learning paradigm that can be used to study algorithms in the context of grammatical inference is the one of *stochastic finite learning* [51]. In this framework, an algorithm is said to have learned a language if, from any infinite sequence of data of this language drawn from a probability distribution, it stops after having seen a finite number of elements and its hypothesis at that point is correct with high probability. The expected number of examples that the learner needs before convergence forms a measure of the algorithm's efficiency. This approach is similar in its aims to identification in the limit, but it can also be seen as a probably exactly correct paradigm. It is thus a tempting way to fill the gap between PAC-learning and identification in the limit. However, results in this paradigm are hard to obtain and even simple classes of languages are known to be not learnable. Many of the arguments of the previous section on the PAC-paradigm work can be used for this formalization. On the other hand, there are positive results for some classes of pattern languages [52].

We believe the reasons above, or some combination thereof, have led many scholars to seek a way to incorporate efficiency into the identification in the limit paradigm (as opposed to abandoning the paradigm altogether).

2.3 The Limits of Gold's Paradigm

2.3.1 *Identification in the Limit*

We now provide a detailed formalization of the identification in the limit paradigm.

A presentation P of a language L is an infinite sequence of data corresponding to L. We note $P[i]$ the ith element of P and P_i the set of the ith first elements of P. If the data contains only elements of L then the presentation is called a *text* of language L. A text T is a complete presentation of L iff for all $w \in L$ there exists $n \in \mathbb{N}$ such that $T[n] = w$. If data in the presentation are instead pairs (w, l), such that $w \in \Sigma^*$ and l is a Boolean valued TRUE if $w \in L$ and FALSE otherwise, then the presentation is called an *informant*. An informant I is a complete presentation of L iff for all $w \in \Sigma^*$ there exists $n \in \mathbb{N}$ such that $I[n] = (w, l)$. In the rest of the chapter, we will only consider complete presentations.

A learning algorithm in this context, sometimes called an *inductive inference machine*, is an algorithm that takes as input larger and larger initial segments of a presentation and outputs, after each input, a hypothesis from a pre-specified hypothesis space.

Definition 2.3 (*Identification in the limit* [24]) A class $\mathbb{L}$ of languages is *identifiable in the limit (IIL)* from text (resp. from informant) if and only if there exists a learning algorithm $\mathfrak{A}$ such that for all languages $L \in \mathbb{L}$, for all text T (resp. informant I) of L,

- there exists an index N such that $\forall n \geq N, \mathfrak{A}(T_n) = \mathfrak{A}(T_N)$ [resp. $\mathfrak{A}(I_n) = \mathfrak{A}(I_N)$]
- $\mathscr{L}(\mathfrak{A}(T_N)) = L$ [resp. $\mathscr{L}(\mathfrak{A}(I_N)) = L$]

Angluin [5] characterizes exactly those classes of languages that are identifiable in the limit from text. The central theorem in this work refers to the presence of 'telltale' finite subsets for each language in the class. Later, in Sect. 2.4.2, we will see an efficiency bound in terms of 'characteristic' finite subsets of languages (these are not exactly the same as Angluin's telltale subsets).

Gold [24] established three important results in this paradigm. The first is that the class of all finite languages is identifiable in the limit from text. The second is that no superfinite class of languages can be identified in the limit from text. Despite what the name may evoke, a class of languages is superfinite if it contains all finite languages and at least one infinite language (the class contains thus an infinite number of languages). The third is that any computably enumerable class whose uniform membership problem is decidable[3] is identifiable in the limit from an informant.

The proof of the second result relies on the fact that given a presentation of an infinite language L, there does not exist any index N from which a learner can distinguish the finite language made of the strings seen so far and the infinite language. If the algorithm converges to L on a complete text T for L at N then there is a text for the finite language containing all and only the strings in T_N for which the algorithm will *also* converge to L. Hence the algorithm fails to identify this finite language in the limit.

On the other hand, the learning algorithm for the third result (learning any computably enumerable class with informant) is really naive: it enumerates the elements of the class until it finds the first one consistent with the information so far. In other words, the algorithm always conjectures the first language in the enumeration that accepts all positive examples (labeled TRUE) and rejects all negative ones (labeled FALSE). If it is the correct hypothesis, the algorithm has converged. If not, then there will be an example later in the presentation that will be inconsistent with the current hypothesis and consequently the algorithm will move along down the enumeration to the next consistent language.

This third result, though of positive nature, is one of the main reasons that the identification in the limit paradigm needs to be refined to include a notion of tractability. 'Learning by enumeration' is clearly not tractable and thus is of little use. While it meets the letter of the definition of learning, it violates our intuitions of what learning should be like. At first glance, a natural way to exclude such learning 'solutions' is to add a tractability requirement to the definition in some way. However, as we now discuss, this is more difficult than it may initially appear.

[3] The uniform membership problem is the one where given a string and a representation one needs to determine whether the string belongs to the represented language.

For more on variations of Gold's original paradigm see Chap. 1, *Gold-Style Learning Theory* (Case).

2.3.2 Polynomial Time

Given the limitations of IIL shown in the previous section, designing requirements to add to the paradigm is needed to strengthen the validity of learning ideas. An intuitive way to deal with that is to constrain the time allowed for the algorithm to make its computations.

Limiting the overall running time appears inappropriate since languages may have infinite cardinality and concomitantly there is no bound on the length of the strings. Thus for any polynomial function p, infinite language L, and number n, there is a presentation P for L such that the first element of P is larger than $p(n)$. Stochastic finite learning [51] would be of great interest to readers concerned with this problem since it replaces this worst-case scenario with a learning framework that focuses on expected convergence (where presentations are drawn according to probability distributions).

A more consensual requirement is *update-time efficiency*. An algorithm is update-time efficient if it outputs a new hypothesis in time polynomial in the size of the data seen so far. This is reasonable as far it goes. Unfortunately, this requirement turns out to be no real restriction at all.

In a seminal paper [35], Leonard Pitt shows that update-efficiency is not sufficient to prove the validity of a learning approach. Indeed, using a method now known as Pitt's trick, he proves that any algorithm that can identify a class in the limit can be transformed into an algorithm that identifies the class in the limit and is update-time efficient.

Informally the proof relies on the fact that, given a presentation P, if a learner converges to a correct hypothesis on the initial sequence P_i, a variant can delay the computation of any interesting hypothesis until having seen P_j $(j > i)$ such that the computation time of the initial learner on P_i is polynomial in $\|P_j\|$. This variant of the learning algorithm then has an efficient update-time while also fulfilling the conditions of identification in the limit. Pitt's trick essentially trades time for data so that enforcing tractability in this way has no impact. The set of classes of languages identifiable in the limit without the update-time requirement is exactly the same as the set of classes of languages identifiable in the limit with it.

Pitt's trick reveals that algorithms may be able to efficiently output hypotheses, but convergence can only occur after non-reasonable amounts of *data* have been provided. This lessens the practical utility of the theoretical results when real data is taken into account.

One may wonder if one can prohibit Pitt's trick, which ignores the great part of the given data, by forcing a learner to respect all the given data. Case and Kötzing [10] show that apparently reasonable properties to force a learner to take all the examples into account are not strong enough to prevent Pitt's trick actually when learning from text.

2.3.3 Identification of a Language and the Size of a Target Representation

Despite the problem described in the previous section, the requirement to have polynomial update-time is still desirable. Efforts have been made to enrich the paradigm further such that Pitt-style delaying tricks are not possible.

Most additional requirements are based on the same method: they link the behavior of the algorithm to the size of a representation of the target language. Indeed, though the identification of the target language is required, a polynomial bound cannot be established with respect to the size of the language since non-trivial classes of languages often contain an infinite number of infinite languages. A representation of finite size of the target language is thus needed. Choosing a target representation also focuses the attention on the hypothesis space of the algorithm, which is relevant from a machine learning standpoint.

However, the choice of representations is not central at all in Gold's learning paradigm as a learner's hypotheses can converge to an arbitrary one among equivalent representations for the correct language. The apparent consequence is that the choice of a representation class for a target language class does matter when taking the representation size into account.

But this duality between the identification of a language and an efficiency bound on the size of a target representation has consequences that need to be handled carefully. For example, it is well known that a nondeterministic finite automaton can be exponentially smaller than the smallest deterministic finite automaton accepting the same language. A learning algorithm that behaves efficiently with respect to the size of deterministic finite automata may not be admitted as an efficient algorithm in terms of the size of nondeterministic finite automata. The reader is referred to Chap. 4, *On the Inference of Finite State Automata from Positive and Negative Data* (López and García), for details on this question.

In general, an inefficient learner can be seen as an efficient learner by choosing a class of redundant representations. Therefore, it is important to make clear under which class of representations the efficiency of a learner is discussed.

In principle, the choice of a representation class is arbitrary and seems hard to justify, but in practice there exist orthodox or natural representations for target language classes. For example, minimal deterministic (canonical) finite state automata are widely used to represent regular languages. Since they are uniquely determined based on an algebraic property of regular languages, there is no room to inflate the representation size.

An intuitive way to deal with the duality exposed above would be to define a paradigm where identification is on a target representation and not on a language. The formalization of this idea is known as *strong identification* [12]. However, this approach only makes sense for classes of representations where each language admits a unique representative: otherwise, it is impossible for any algorithm to distinguish between the different grammars generating the same language, and thus the identi-

fication cannot succeed. The use of canonical finite-state automata in the work on regular languages [33] is an example of such an approach.

2.4 First Refinements

2.4.1 Mind Changes and Implicit Errors of Prediction

One way to formalize the notion of convergence with a reasonable amount of data with respect to the size of the representation is to measure the number of *mind changes* [1, 7]. Another way is to measure the number of *implicit prediction errors* [35].

A *mind change* occurs when a learning algorithm replaces its current hypothesis with another. Then one adds to the identification in the limit paradigm the requirement that the number of mind changes made before convergence must be bounded by a polynomial function in the size of the representation.

However, Pitt [35] presents another trick where the algorithm postpones changing its mind solely to meet the requirements of the mind change bound. Consequently, the algorithm maintains untenable hypotheses (ones inconsistent with the data) until a sufficient amount data is seen so that a mind change can occur without violating the polynomial bound on the number of mind changes.

Measuring implicit predictions errors can get around this trick when learning from an informant. When the learner's current hypothesis is inconsistent with a new datum, it is called *an implicit error of prediction*. Then one adds to the identification in the limit paradigm the requirement that the number of times the current hypothesis is in contradiction with the new example has to be polynomial in the size of the target representation. More formally:

Definition 2.4 (*Identification in polynomial number of implicit errors*)

- Given a presentation P, an algorithm $\mathfrak{A}$ makes an implicit error of prediction at step n if the language of the hypothesized target $\mathfrak{A}(P_n)$ is in contradiction with $P[n+1]$.
- A class $\mathbb{G}$ of representations is polynomial-time identifiable in the limit in Pitt's sense if $\mathbb{G}$ admits a polynomial-time learning algorithm $\mathfrak{A}$ such that for any presentation of $\mathscr{L}(G)$ for $G \in \mathbb{G}$, $\mathfrak{A}$ makes implicit errors of prediction at most polynomial in $\|G\|$ [35].
- A class $\mathbb{G}$ of representations is polynomial-time identifiable in the limit in Yokomori's sense if $\mathbb{G}$ admits a polynomial-time learning algorithm $\mathfrak{A}$ such that for any presentation P of $\mathscr{L}(G)$ for $G \in \mathbb{G}$, for any natural number n, the number of implicit errors of prediction made by $\mathfrak{A}$ on the nth first examples is bounded by a polynomial in $m \cdot \|G\|$, where $m = \max\{|P[1]|, \ldots, |P[n]|\}$ [46].

Notice that Yokomori's formulation is a relaxed version of that of Pitt's.

However, if the presentation is a text, there is yet another unwanted Pitt-style delaying trick: the algorithm can output a representation for Σ^*, which will never

be in contradiction with the data. It can then wait to see enough examples before returning a pertinent hypothesis.

On the other hand, if the presentation is an informant, then the additional requirement limiting the number of implicit prediction errors is significant because there is no language like Σ^* which is consistent with both the positive and negative examples. Consequently, it can be shown that not all classes of languages identifiable in the limit in polynomial update time are identifiable in the limit in Pitt's sense or in Yokomori's sense: in the former paradigm, an algorithm working in polynomial time can change its hypothesis an exponential number of times before convergence, while in the latter paradigms this is not allowed and cannot be circumvented as in the case of texts. Note this is different from the mind-change requirement, where the delaying trick there works in both kinds of presentations: in that case, the algorithm can choose to not update its hypothesis when a new example contradicts it.

Another property of these requirements is that they are mainly designed for incremental algorithms. Indeed, these paradigms give a lot of importance to the sequence of data, in particular as the parts of two sequences that contain the same elements in a different order might not correspond to the same number of implicit errors (or mind changes). This forces the complexity analysis to consider particularly malevolent sequences of data. However, in many practical frameworks, for instance in Natural Language Processing or Bio-informatics, we are interested in algorithms that work from a finite *set* of data, where the order of presentation is irrelevant. From this perspective, the (inadvertent) focus on an incremental process appears to be a drawback.

The main positive learning results using this approach concerns the class of very simple languages [47, 49]: an algorithm has been designed that fulfills the requirements of Yokomori's formulation of the paradigm. This class of languages is incomparable with the class of regular languages and contains context-free languages.

2.4.2 Characteristic Sample

The most widely used definition of data efficiency relies on the notion of *characteristic sample*. The characteristic sample is a finite set of data from a language L that ensures the correct convergence of the algorithm on any presentation of L as soon as it is included in the data seen so far. For some, these characteristic samples evoke Angluin's telltale subsets [5], also of finite size, which were central to characterizing the nature of classes of formal languages identifiable from text.

In this learning paradigm [18], it is required that the algorithm needs a characteristic sample whose size[4] is polynomial in the size of the target representation. Formally:

Definition 2.5 (*Identification in the limit in polynomial time and data*) A class of languages $\mathbb{L}$ is *identifiable in the limit in polynomial time and data* from a class $\mathbb{R}$ of representations iff there exist a learning algorithm $\mathfrak{A}$ and a polynomial $p()$ such that for any language $L \in \mathbb{L}$, for any representation $R \in \mathbb{R}$ of L:

- $\mathfrak{A}$ has a polynomial update-time,
- there exists a set of data CS, called a characteristic sample, of size at most $p(\|R\|)$ such that for any presentation P of L, if $CS \subseteq P_n$ then $\mathfrak{A}(P_n)$ is equivalent to R, and for all $N > n$, $\mathfrak{A}(P_N) = \mathfrak{A}(P_n)$.

The idea underlying the paradigm is that if the data available to the algorithm so far does not contain enough information to distinguish the target from other potential targets then it is impossible to learn. This complexity requirement diverges from update-time requirements above in that incremental learning algorithms no longer sit at the core of the paradigm. Indeed, limiting the complexity in terms of the characteristic sample makes possible the set-based definition that we are developing below.

Definition 2.6 Let $\mathbb{L}$ be a class of languages represented by some class $\mathbb{R}$ of representations.

1. A *sample* S for a language $L \in \mathbb{L}$ is a finite set of data consistent with L. A *positive sample* for L is made only of elements of L. A positive and negative sample for L is made of pairs (w, l), where l is a boolean such that $l = \text{TRUE}$ if $w \in L$ and $l = \text{FALSE}$ otherwise. The *size* of a sample S is the sum of the size of all its elements plus $|S|$.
2. An $(\mathbb{L}, \mathbb{R})$-learning algorithm $\mathfrak{A}$ is a program that takes as input a sample for a language $L \in \mathbb{L}$ and outputs a representation from $\mathbb{R}$.

We can now formalize the notion of characteristic sample in the set-based approach.

Definition 2.7 (*Characteristic sample*) Given an $(\mathbb{L}, \mathbb{R})$-learning algorithm $\mathfrak{A}$, we say that a sample CS is a *characteristic sample* of a language $L \in \mathbb{L}$ if for all samples S such that $CS \subseteq S$, $\mathfrak{A}$ returns a representation R such that $\mathscr{L}(R) = L$.

Hopefully it is evident that the class of representations is especially relevant in this paradigm.

The learning paradigm can now be defined as follows.

[4]The size of a sample is the sum of the length of its elements: it has been shown [35] that its cardinality is not a relevant feature when efficiency is considered, as it creates a risk of collusion: one can delay an exponential computation on a given sample of data and wait for a sufficient number of examples to run the computation on the former sample in polynomial time in the size of the latter.

Definition 2.8 (*Set-based identification in polynomial time and data* [18]) A class $\mathbb{L}$ of languages is *identifiable in polynomial time and data (IPTD)* from a class $\mathbb{R}$ of representations if and only if there exists an $(\mathbb{L}, \mathbb{R})$-learning algorithm $\mathfrak{A}$ and two polynomials $p()$ and $q()$ such that:

1. Given a sample S of size m for $L \in \mathbb{L}$, $\mathfrak{A}$ returns a consistent hypothesis $H \in \mathbb{R}$ in $\mathcal{O}(p(m))$ time.
2. For each representation R of size k of a language $L \in \mathbb{L}$, there exists a characteristic sample of L of size at most $\mathcal{O}(q(k))$.

Notice that the first item is a reformulation of the polynomial update time requirement, which is now in terms of the size of the sample. The second item corresponds to the additional requirement that the amount of data needed to converge is computationally reasonable. By forcing the algorithm to converge to a correct hypothesis whenever a characteristic sample of reasonable size has been seen, this paradigm tackles the risk of collusion by forbidding Pitt's delaying tricks.

The main reason this unusual way to formalize identification is chosen is because by formalizing learning when a set of data is available it corresponds to the most common framework when real-world data is considered.

Furthermore, the set-based approach encompasses the incremental approach since any algorithm studied in the latter can easily be cast into a set-based one. In other words, any algorithm that learns a class of languages in the sense of Definition 2.5 also learns the class in the sense of Definition 2.8.

However, it is not easy to cast set-based learners into incremental ones. Naively one may believe that for any algorithm $\mathfrak{A}$ satisfying Definition 2.8, there exists an incremental algorithm which satisfies Definition 2.5. The idea would be, for each new data, to run $\mathfrak{A}$ on the set of data seen so far. However, as shown in Appendix, this simple approach will not always work. There is an algorithm for learning the substitutable context-free languages which satisfies Definition 2.8 for which this incremental construction fails. In Appendix, it is shown that unless this incremental algorithm $\mathfrak{A}$ is conservative,[5] $\mathfrak{A}$ will not converge to a single grammar. However, if $\mathfrak{A}$ is conservative then there is a presentation at a point at which the characteristic set is seen but $\mathfrak{A}$ has not yet converged to the correct grammar. It remains to be seen whether for every set-based learner satisfying Definition 2.8, there is an incremental learner satisfying Definition 2.5.

Main Results Many learning algorithms have been studied in the context of IPTD. The main positive results concerns approaches that used positive and negative examples as input. In this context, regular languages are learnable [18] using deterministic finite state automata, and so are deterministic even linear languages as the question of inferring these grammars can be reduced to that of inferring deterministic finite state automata [40]. Another related class of languages that have been positively investigated in this context is the deterministic linear one [20]. The algorithms is fed with positive and negative examples and outputs a deterministic linear grammar.

[5]An incremental learner is *conservative* if it changes its current hypothesis H if and only if the next datum is inconsistent with H.

Context-free languages that are representable by delimited, almost non-overlapping string rewriting systems are also IPTD-learnable [22] from positive and negative examples. Comparisons of this class with the previous ones are difficult since they are not defined using the same kind of representation.

The whole class of context-free languages is learnable in the IPTD sense [19, 38] from structural positive examples, that is to say derivation trees with no information on the internal nodes. Given a positive integer k, the target class of representation is that of k-reversible context-free grammars [32] and the elements of the sample have to correspond to derivation trees of these grammars.

Limitations We have already discussed one drawback to measuring the complexity of the learning problem in terms of the size of the representation. It can be unclear what counts as a 'reasonable' representation. Consequently, it may be possible to artificially inflate representations to allow learning. This is another kind of trick since the algorithm would be efficient according to the letter of the definition but not its spirit.

Identification in polynomial time and data also suffers from the opposite kind of drawback. As we will see, for non-regular languages, there can be exponentially *compact* representations of languages. For such cases, IPTD-learning appears to give the wrong results: classes which intuitively ought to count as tractably learnable (because they return a very compact representation of the target language) can in fact be shown to not be IPTD-learnable. As IPTD was developed and studied in the context of learning regular languages, neither of these problems arose since minimal deterministic finite-state automata are considered to be reasonable representations of regular languages.

Example 2.3 illustrates the problem for the IPTD-learning of non-regular languages. It proves that context-free languages cannot be learned under this criterion using context-free grammars. Indeed, the characteristic sample of any grammar of the series has to contain the only string in the language, but the length of this string is exponentially greater than the size of the grammar.

Example 2.3 [18] let $\mathbb{G}_1 = \cup_{n>0}\{G_n\}$ be the class of context-free grammars such that for any n, the unique axiom of G_n is N_0 and its production rules are $N_i \to N_{i+1}N_{i+1}$, for $0 \leq i < n$, and $N_n \to a$. The language of G_n is the singleton $L(G_n) = \{a^{2^n}\}$.

The reason why this example is not learnable does not come from the hardness of the languages: they are made of only one string. But the use of any class of representations that contains $\mathbb{G}_1$ is not identifiable in the limit.

It seems that in this case the problem comes from the definition of what learning means, that is to say from the learning criterion, rather than the properties of the language. From an information theory point of view, it is obviously interesting to have an algorithm that is able to find a model explaining the data it is fed with that is exponentially smaller than the data. This is actually a desired property in many fields of machine learning (see [23] for instance). Hence, the trouble here comes from the learning paradigm.

2.5 Recent Refinements

In this section, we review two contemporary approaches that develop a definition of efficient learning which can be applied to non-regular classes of languages. They are both refinements of the identification with polynomial time and data.

2.5.1 *Structurally Complete Set*

We first introduce the following definition:

Definition 2.9 (*Structurally complete set*) Given a generative grammar G, a structurally complete set (SCS) for G is a set of data SC such that for each production $\alpha \to \beta$, there exists an element $x \in SC$, an element $\gamma \in I$ and two elements $\eta, \tau \in (\Sigma \cup N)^*$ such that $\gamma \Rightarrow^* \eta\alpha\tau \Rightarrow \eta\beta\tau \Rightarrow^* x$. The smallest structurally complete set (SSCS) S for a grammar G is the sample such that for all SCS S' for G, $S \trianglelefteq S'$.

A notion of structurally complete sample has already been defined in the context of regular language learning [21]. However, this former definition relied on a particular representation, namely the finite state automaton, and it considered only the case of positive and negative examples. Definition 2.9 is more general as it does not depend on a particular representation and does not consider a particular type of data. Definition 2.9 is a generalization of the notion of *representative sample* [42] that has been introduced in the context of learning from membership queries and a sample of positive examples of a subclass of context-free languages named simple deterministic languages.

Definition 2.10 (*Polynomial structurally complete identification*) A class $\mathbb{L}$ of languages is *identifiable in polynomial time and structurally complete data (IPTscD)* for a class $\mathbb{R}$ of representations if and only if there exists an algorithm $\mathfrak{A}$ and two polynomials $p()$ and $q()$ such that:

1. Given a sample S for $L \in \mathbb{L}$ of size m, $\mathfrak{A}$ returns a consistent hypothesis $H \in \mathbb{R}$ in $\mathcal{O}(p(m))$ time.
2. For each representation R of a language $L \in \mathbb{L}$, there exists a *characteristic sample* CS whose size is in $\mathcal{O}(q(k))$, where k is the size of the smallest structurally complete set for R.

Notice that in the case where negative data is also available, the size of the characteristic sample has to be polynomial in the size of a SCS which contains only positive examples. This implies that the amount of negative evidence has to be polynomially related to that of the positive evidence.

This paradigm shifts the perspective considerably: the efficiency does not rely anymore directly on the size of the representation but instead on the kind of strings it can generate. This move is anticipated, and pursued in part, in the approach by Ryo Yoshinaka [48], discussed in Sect. 2.5.2.

Comparison with IPTD Consider the class of languages $\mathbb{L}_2 = \bigcup_{n\in\mathbb{N}} \{\{a^i : 0 \leq i \leq 2^n\}\}$. This class is identifiable in polynomial time and data from positive data only using the class of representations $\mathbb{G}_2 = \bigcup_{n\in\mathbb{N}} \{\{\{a\}, \{S, A\}, \{S \to A^{2^n}, A \to a|\lambda\}, \{S\}\}\}$. Indeed, given a target language, the simple algorithm that returns the only grammar consistent with a sample admits the characteristic sample $\{a^{2^n}\}$ which is linear in the size of the target. However, the smallest structurally complete set of any target grammar is $\{\lambda, a\}$ which is of size 2. As the size of the smallest SCS is constant and the class of languages infinite, $\mathbb{L}_2$ is not identifiable in polynomial time and structurally complete data.

On the other hand, let us consider the class of languages of Example 2.3: $\mathbb{L}_1 = \bigcup_{n\in\mathbb{N}} \{\{a^{2^n}\}\}$ and its class of representations $\mathbb{G}_1 = \bigcup_{n\in\mathbb{N}} \{\{\{a\}, N_n, P_n, \{N_0\}\}\}$, with $P_n = \{N_n \to a\} \cup_{0\leq i<n} \{N_i \to N_{i+1}N_{i+1}\}$. Given n, the characteristic sample is $\{a^{2^n}\}$ which is also the smallest structurally complete set for the target grammar. However, this sample is not polynomial in the size of the target grammar. Therefore $\mathbb{L}_1$ is identifiable in the limit in polynomial time and structurally complete data using $\mathbb{G}_1$ but not in polynomial time and data.

This shows that these two paradigms are thus non-comparable. However, most non-trivial language classes studied under the former paradigm are identifiable in polynomial time and structurally complete data. This is the case for instance for the regular languages from positive and negative examples and for all sub-regular classes studied in the context of grammatical inference: there is a linear link between the size of a regular grammar and what can be derived from any of its non-terminals.

2.5.2 Thickness

In a recent paper [48], Ryo Yoshinaka introduced the identification from a characteristic sample whose size is a polynomial in the size of the target grammar and of a measure called the thickness of the grammar.

Definition 2.11 (*Thickness*) Let $G = \langle \Sigma, N, P, I \rangle$ be a generative grammar. The *thickness* of G is $\tau_G = \max\{|\omega(\alpha)| : \exists \beta, \alpha \to \beta \in P\}$ where $\omega(\alpha) = \min_{\lhd}\{w \in \Sigma^* : \alpha \Rightarrow_G^* w\}$.

Informally, the thickness is the length of the longest string in the set of the smallest strings that can be generated from a left hand-side of a grammar rule.

This definition is an extended version of the one that was first introduced for context-free grammars in the context of model complexity [45]. Notice that it has nothing to do with the usual notion of thickness in learning theory.

Definition 2.12 (*Polynomial thick identification* [48]) A class $\mathbb{L}$ of languages is *identifiable in polynomial time and thick data (IPTtD)* for a class $\mathbb{R}$ of representations if and only if there exists an algorithm $\mathfrak{A}$ and two polynomials $p()$ and $q()$ such that:

1. Given a sample S for $L \in \mathbb{L}$ of size m, $\mathfrak{A}$ returns a consistent hypothesis $H \in \mathbb{R}$ in $\mathcal{O}(p(m))$ time.
2. For each representation R of a language $L \in \mathbb{L}$ of size k, there exists a *characteristic sample* CS whose size is in $\mathcal{O}(q(k, \tau_R))$.

IPTtD is clearly a refinement of IPTD since it simply adds the thickness as a parameter of the paradigm. It is however a fundamental move since it links the efficiency of the learning not only on the target representation but also to the kind of strings the grammar produces. This shift in perspective is a way to indirectly take into account the length of the strings in the language in the learning criterium. On the other hand, since it does not go so far as to remove the requirement that the characteristic sample be polynomial in the size of the grammar, it is still susceptible to inflation tricks.

Learnable Classes Since the size of the representation is used in Definition 2.12, it is clear that every class of languages that is IPTD is also IPTtD.

However, the converse is not true. Consider the grammars of Example 2.3: The thickness of any $G_n \in \mathbb{G}_1$ is 2^n.

More interesting examples are the classes of languages that have been investigated in the context of what is called distributional learning (see Chap. 6, *Distributed hearing of contest-free and multiple contest free Grammars*, Clark and Yoshinaka). For instance, a context-free language is *substitutable* if whenever two substrings appear once in the same context, then they always appear in the same context in the language [13].

There exists a polynomial-time algorithm that identifies the class of context-free substitutable languages from positive examples only, in the sense of Definition 2.8, but the exhibited characteristic sample might be of size exponential in the size of the target representation (this is the case for the languages of Example 2.3, which are substitutable). Thus, this algorithm is not IPTD. On the other hand, it is easy to see that this characteristic sample is polynomial in the size *and the thickness* of the target grammar, so the algorithm is IPTtD. This result can be extended to the more complex classes that have been studied in the context of distributional learning from positive examples only (see for instance [48, 50]).

2.5.3 Comparison of the Two Refinements

Since the IPTD and IPTscD classes are incomparable and every IPTD class is IPTtD, clearly there is an IPTtD class which is not IPTscD (this is the case for instance of the class $\mathbb{L}_2$ introduced at the end of Sect. 2.5.1). However, one can show that every

IPTtD class of unambiguous CFGs is IPTscD. Also, it is easy to see that every IPTscD class of context-free languages is IPTtD using the same class of representations.

The two refinements of polynomial identification share a basic idea – to measure the complexity by the size of the simplest strings that a grammar generates, rather than the description size of it. Indeed one can show that the size of the smallest SCS of G is polynomially bounded by $\tau_G \|G\|$. That is, if a language class is IPTscD for a class of context-free grammars, then it is also IPTtD.

However, the converse is not necessarily true. The following discussion illustrates a particularly difficult problem for IPTscD learning: ambiguity. Let G_n consists of the following rules:

$$P_n = \{ A \to a,\ A \to b,\ B \to b \} \cup \{ S \to X_1 \dots X_n \mid X_i \in \{A, B\} \},$$

which generates $L(G_n) = \{a, b\}^n$. Then the set $\{a^n, b^n\}$, whose size is $2n + 2$, is the smallest SCS for G_n. On the other hand, $\|G_n\| \in \mathcal{O}(n2^n)$ and $\tau_{G_n} = n$. When learning the class $\mathbb{L} = \bigcup_{n \in \mathbb{N}} \mathbb{L}_n$ where $\mathbb{L}_n = \{ L \subseteq \{a, b\}^n \} \setminus \{\emptyset\}$ with a positive sample, the only possible characteristic sample of $L(G_n)$ is $L(G_n)$ itself for any learning algorithm. Therefore, $\mathbb{L}$ is not IPTscD for any representation class. One can easily see that $\mathbb{L}$ is IPTtD for a reasonable class of grammars where G_n is the unique grammar for $\{a, b\}^n$.

The grammar G_n is very redundant and highly ambiguous—there are 2^n ways to derive b^n. If the redundancy is removed from G_n by deleting the nonterminal B and the rules involving B, the size of the grammar is now $O(n)$ and it is not IPTtD any more. In fact, one can show that $\tau_G \|G\|$ is polynomially bounded by the size of the smallest SCS when only unambiguous context-free grammars are considered.

2.6 Conclusion

The purpose of this chapter was to address the problem of efficiently learning formal languages and grammars. We argued that the PAC framework is not the best suited one even though its efficiency requirements are well-designed. On the other hand, we argued in favor of identification in the limit paradigms provided they are adequately modified to include efficiency requirements. This survey showed doing so is more challenging than anyone may have anticipated. We discussed the challenges that have been encountered by different attempts. For regular languages, de la Higuera's [18] solution is satisfactory due to the canonical representation given by the smallest deterministic acceptors. For non-regular languages, challenges remain. We discussed two promising paths forward to address efficient learning in the identification in the limit paradigm in the realm of non-regular languages. One was based on the notion of a structurally complete sample, and the other was based on the 'thickness' of strings generated by production rules. Both are measuring efficiency at least partly

in terms of the size of particular strings generated by grammars. We believe further developments along these lines will help shape future directions in grammatical inference.

Appendix

Here we present an example that shows that a learning result in a set-based approach (that of IPTtD) may not yield to a learning result in the incremental approach.

A characteristic sample has been exhibited for a set-based polynomial-time learning algorithm[6] for the class of substitutable context-free languages [13]. The size of this characteristic sample is polynomial in the size of the target grammar and its thickness [48]. From any superset of this set, the algorithm returns a representation that generates the target language. Therefore, one can state that the algorithm learns the class of substitutable context-free languages in a set-based approach.

A particularity of this algorithm is that from two different supersets of the characteristic sample, it may returns two different equivalent grammars, and the number of such pairs of samples is infinite (this is due to the infinite number of congruence classes that a context-free language defines). Consider the incremental version of the algorithm that computes a new grammar for every new example. It therefore does not fit the conditions of identification in the limit since there does not exist a moment after which the algorithm always returns the same hypothesis, though there exists a point after which the generated language will always be the target one.[7]

An intuitive solution is then to make the algorithm conservative: the incremental version of the algorithm has to change its hypothesis only if the new example is not recognized. However, this is not working as is shown with the following example.

Consider the language $a(\{b, c\}\{b, c\})^*$, which is substitutable. It is also context-free as it can be generated by the grammar whose rules are $S \to a|SBB$ and $B \to b|c$, with S being the only axiom.

As defined in the previously cited papers, the characteristic sample is the following set: $CS = \{lur \in \Sigma^* : \exists N \to \alpha, (l, r) = C(A) \text{ and } u = \omega(\alpha)\}$, where $C(A)$ is the smallest context in which the non-terminal A can appear in a derivation, and $\omega(\alpha)$ is the smallest element of Σ^* that can be derived from α in the grammar.

If we assume $a < b < c$ and $(ab, \lambda) < (a, b)$, the characteristic sample is then $CS = \{a, abb, abc\}$.

Suppose the learner gets examples $a, abb, abbbc$ in this order. As the letter c is new, the conjecture has to be updated at this point. The new conjecture is then the string rewriting system $\{a \to abb, a \to abc, b \to bbc\}$ with a being the only axiom.

[6] Notice that the algorithm was originally presented in an incremental paradigm. However, its study was (mostly) done in a set-based framework and, as is shown in this appendix, the proofs are valid only in this context.

[7] This is known as *behaviorally correct* identification in the limit.

It generates every sentence in the characteristic sample.[8] However the hypothesis is not correct since for example *acc* is in the target language but not in the current one. Therefore, if the next example is the missing string of the characteristic sample, *abc*, the algorithm will not change its hypothesis: though all elements of the characteristic sample are available, the current hypothesis is not correct. Once an element of the language that is not generated by the hypothesis is seen, the hypothesis will be updated using a set containing a characteristic sample and thus the new conjecture will correspond to a correct representation of the language.

References

1. A. Ambainis, S. Jain, and A. Sharma. Ordinal mind change complexity of language identification. *Theoretical Computer Science*, pages 323–343, 1999.
2. D. Angluin. Finding patterns common to a set of strings. *Journal of Computer and System Sciences*, 21:46–62, 1980.
3. D. Angluin. Queries and concept learning. *Machine Learning*, 2(4):319–342, 1987.
4. D. Angluin, J. Aspnes, and A. Kontorovich. On the learnability of shuffle ideals. In *Proceedings of the Algorithmic Learning Theory Conference*, pages 111–123, 2012.
5. Dana Angluin. Inductive inference of formal languages from positive data. *Information and Control*, 45:117–135, 1980.
6. L. Becerra-Bonache, A. Dediu, and C. Tirnăucă. Learning DFA from correction and equivalence queries. In *Proceedings of the International Colloquium on Grammatical Inference*, pages 281–292, 2006.
7. L. E. Blum and M. Blum. Toward a mathematical theory of inductive inference. *Information and Control*, 28(2):125–155, 1975.
8. A. Blumer, A. Ehrenfeucht, D. Haussler, and M. Warmuth. Learnability and the Vapnik-Chervonenkis dimension. *Journal of the ACM*, 36(4):929–965, 1989.
9. R. Book and F. Otto. *String-Rewriting Systems*. Springer Verlag, 1993.
10. J. Case and T. Kötzing. Difficulties in forcing fairness of polynomial time inductive inference. In *Proceedings of the Algorithmic Learning Theory Conference*, pages 263–277, 2009.
11. N. Chomsky. Three models for the description of language. *IRE Transactions on Information Theory*, 2:113–124, 1956.
12. A. Clark. Learning trees from strings: A strong learning algorithm for some context-free grammars. *Journal of Machine Learning Research*, 14:3537–3559, 2014.
13. A. Clark and R. Eyraud. Polynomial identification in the limit of substitutable context-free languages. *Journal of Machine Learning Research*, 8:1725–1745, 2007.
14. A. Clark and S. Lappin. *Linguistic Nativism and the Poverty of the Stimulus*. Wiley-Blackwell, 2011.
15. A. Clark and F. Thollard. PAC-learnability of probabilistic deterministic finite state automata. *Journal of Machine Learning Research*, 5:473–497, 2004.
16. A. Clark and R. Yoshinaka. Distributional learning of parallel multiple context-free grammars. *Machine Learning*, 96:5–31, 2014.
17. H. Comon, M. Dauchet, R. Gilleron, C. Löding, F. Jacquemard, D. Lugiez, S. Tison, and M. Tommasi. Tree automata techniques and applications. Available on: http://tata.gforge.inria.fr/, 2007.

[8] The algorithm is consistent which implies that the two first elements of the characteristic sample are in the conjectured language and we have the rule $a \rightarrow abc$ that generates the third element of CS from the axiom.

18. C. de la Higuera. Characteristic sets for polynomial grammatical inference. *Machine Learning*, 27:125–138, 1997.
19. C. de la Higuera. *Grammatical inference: learning automata and grammars*. Cambridge University Press, 2010.
20. C. de la Higuera and J. Oncina. Learning deterministic linear languages. In *Proceedings of Conference on Learning Theory*, pages 185–200, 2002.
21. P. Dupont, L. Miclet, and E. Vidal. What is the search space of the regular inference? In *Proceedings of the International Colloquium on Grammatical Inference*, pages 25–37, 1994.
22. R. Eyraud, C. de la Higuera, and J.-C. Janodet. LARS: A learning algorithm for rewriting systems. *Machine Learning*, 66(1):7–31, 2007.
23. F. Girosi. An equivalence between sparse approximation and support vector machines. *Neural Comput.*, 10(6):1455–1480, 1998.
24. E. M. Gold. Language identification in the limit. *Information and Control*, 10(5):447–474, 1967.
25. J. Heinz. Computational theories of learning and developmental psycholinguistics. In J. Lidz, W. Synder, and J. Pater, editors, *The Oxford Handbook of Developmental Linguistics*. Cambridge University Press, in press
26. D. Hsu, S. M. Kakade, and P. Liang. Identifiability and unmixing of latent parse trees. In *Advances in Neural Information Processing Systems (NIPS)*, pages 1520–1528, 2013.
27. M. Isberner, F. Howar, and B. Steffen. Learning register automata: from languages to program structures. *Machine Learning*, 96:65–98, 2014.
28. Y. Ishigami and S. Tani. VC-dimensions of finite automata and commutative finite automata with k letters and n states. *Discrete Applied Mathematics*, 74:123–134, 1997.
29. J. Langford. Tutorial on practical prediction theory for classification. *Journal of Machine Learning Research*, 6:273–306, December 2005.
30. M. Li and P. Vitanyi. Learning simple concepts under simple distributions. *SIAM Journal of Computing*, 20:911–935, 1991.
31. E. Moore. Gedanken-experiments on sequential machines. In Claude Shannon and John McCarthy, editors, *Automata Studies*, pages 129–153. Princeton University Press, 1956.
32. T. Oates, D. Desai, and V. Bhat. Learning k-reversible context-free grammars from positive structural examples. In *Proceedings of the International Conference in Machine Learning*, pages 459–465, 2002.
33. J. Oncina and P. García. Identifying regular languages in polynomial time. In *Advances in Structural and Syntactic Pattern Recognition*, volume 5 of *Series in Machine Perception and Artificial Intelligence*, pages 99–108. 1992.
34. T.-W. Pao and J. Carr III. A solution of the syntactical induction-inference problem for regular languages. *Computer Languages*, 3(1):53 – 64, 1978.
35. L. Pitt. Inductive inference, DFA's, and computational complexity. In *Analogical and Inductive Inference*, number 397 in LNAI, pages 18–44. Springer-Verlag, 1989.
36. D. Ron, Y. Singer, and N. Tishby. On the learnability and usage of acyclic probabilistic finite automata. In *Proceedings of the Conference on Learning Theory*, pages 31–40, 1995.
37. G. Rozenberg, editor. *Handbook of Graph Grammars and Computing by Graph Transformation: Volume I. Foundations*. World Scientific, 1997.
38. Y. Sakakibara. Efficient learning of context-free grammars from positive structural examples. *Information and Computation*, 97:23–60, 1992.
39. Hiroyuki Seki, Takashi Matsumura, Mamoru Fujii, and Tadao Kasami. On multiple context-free grammars. *Theoretical Computer Science*, 88(2):191–229, 1991.
40. J. M. Sempere and P. García. A characterization of even linear languages and its application to the learning problem. In *Proceedings of the International Colloquium in Grammatical Inference*, pages 38–44, 1994.
41. C. Shibata and R. Yoshinaka. PAC-learning of some subclasses of context-free grammars with basic distributional properties from positive data. In *Proceedings of the Algorithmic Learning Theory conference*, pages 143–157, 2013.

42. Y. Tajima, E. Tomita, M. Wakatsuki, and M. Terada. Polynomial time learning of simple deterministic languages via queries and a representative sample. *Theoretical Computer Science*, 329(1-3):203 – 221, 2004.
43. L. G. Valiant. A theory of the learnable. *Communications of the Association for Computing Machinery*, 27(11):1134–1142, 1984.
44. V. Vapnik. *The nature of statistical learning theory*. Springer, 1995.
45. M. Wakatsuki and E. Tomita. A fast algorithm for checking the inclusion for very simple deterministic pushdown automata. *IEICE TRANSACTIONS on Information and Systems*, VE76-D(10):1224–1233, 1993.
46. T. Yokomori. On polynomial-time learnability in the limit of strictly deterministic automata. *Machine Learning*, 19:153–179, 1995.
47. T. Yokomori. Polynomial-time identification of very simple grammars from positive data. *Theoretical Computer Science*, 1(298):179–206, 2003.
48. R. Yoshinaka. Identification in the limit of k, l-substitutable context-free languages. In *Proceedings of the International Colloquium in Grammatical Inference*, pages 266–279, 2008.
49. R. Yoshinaka. Learning efficiency of very simple grammars from positive data. *Theoretical Computer Science*, 410(19):1807–1825, 2009.
50. R. Yoshinaka. Efficient learning of multiple context-free languages with multidimensional substitutability from positive data. *Theoretical Computer Science*, 412:1821–1831, 2011.
51. T. Zeugmann. Can learning in the limit be done efficiently? In *Proceedings of the Algorithmic Learning Theory conference*, pages 17–38, 2003.
52. T. Zeugmann. From learning in the limit to stochastic finite learning. *Theoretical Computer Science*, 364(1):77–97, 2006.

Chapter 3
Learning Grammars and Automata with Queries

Colin de la Higuera

Abstract When learning languages or grammars, an attractive alternative to using a large corpus is to learn by interacting with the environment. This can allow us to deal with situations where data is scarce or expensive, but testing or experimenting is possible. The situation, which arises in a number of fields, is formalised in a setting called active learning or query learning. By controlling better the information to which one has access, this setting provides us with a better understanding of the hardness of learning tasks. But the setting also allows us to solve practical learning situations, for which new algorithms are needed.

3.1 Introduction

Grammatical inference deals with the question of learning grammatical models, such as automata, grammars or transducers, given information about a language [32]. The most general setting is that of learning from examples over which the learner has no say, often allowing us to reformulate the learning problem as a combinatorial question: find the smallest automaton such that some condition is met. An alternative introduced by Angluin is to allow the learning algorithm to have access to its environment, and question it about the language for which a grammar is sought [29, 44, 50].

Query learning [5] can be described as a game where the learner can ask questions (queries) to an Oracle (teacher) about the target language. The game ends when the learner guesses the target. Of course, the learning results strongly depend on the sort of queries that the learner is allowed to make.

This work was partially supported by the IST Programme of the European Community, under the PASCAL 2 Network of Excellence, IST–2007-216886. The author also acknowledges support by the Région Pays de Loire. This publication only reflects the author's views.

C. de la Higuera (✉)
LINA, University of Nantes, Nantes, France
e-mail: cdlh@univ-nantes.fr

J. Heinz and J.M. Sempere (eds.), *Topics in Grammatical Inference*,
DOI 10.1007/978-3-662-48395-4_3