Chapter 1
Gold-Style Learning Theory

A Selection of Highlights Since Gold

John Case

Abstract This chapter is a tutorial addressed to, among other audiences, the grammatical inference community. It is on the computational learning theory initiated by E. Mark Gold's seminal 1967 paper. One of Gold's important motivations was to present a formal model of child language learning. This chapter introduces Gold's model and also presents an introduction to some selected highlights of the theory appearing since Gold 1967. Since both Gold and the present author had or have cognitive science motivations, many of the results discussed herein also have implications for, and were motivated by, questions regarding the human cognitive ability to learn. For this chapter some prior knowledge of computability theory would be helpful to the reader, as would some prior acquaintance with Gold's identification/learning in the limit concept. The first section concentrates on results which are independent of computational complexity considerations. Covered are: Gold's model of child language learning (including some reasonable, post-Gold criteria of successful learning and some critical discussion); some severe constraints on successful learning; some characterization results and important examples of successful learning; discussion and results on potential child insensitivities to data presentation order; and empirical U-shaped learning with some corresponding formal results interpreted for their cognitive science significance. The second section concentrates on relevant complexity issues. The issues considered are: large database size and corresponding *memory-limited learners*; unavoidable cases of complexity and information deficiencies *of the programs learned*; and the complexity of *learner updates*. In this section a few, seemingly difficult, open mathematical questions are indicated. Some of them are important for cognitive science.

J. Case (✉)
Computer and Information Sciences Department, University of Delaware,
Newark, DE 19716, USA
e-mail: case@udel.edu

J. Heinz and J.M. Sempere (eds.), *Topics in Grammatical Inference*,
DOI 10.1007/978-3-662-48395-4_1

1.1 Language Learnability: Gold 1967 and Beyond

In this section, covered are: Gold's model of child language learning (Sect. 1.1.1); criteria of success (Sect. 1.1.2); some severe constraints on successful learning (Sect. 1.1.3); some characterization results and important examples of successful learning (Sect. 1.1.4); discussion and results on potential child insensitivities to data presentation order (Sect. 1.1.5); and empirical U-shaped learning (Sect. 1.1.6) with some corresponding formal results interpreted for their cognitive science significance (Sect. 1.1.7).

1.1.1 Gold's 1967 Model of Child Language Learning

M, below in Fig. 1.1, is an algorithmic device (machine), and, for $t = 0, 1, 2, \ldots$, at "time" t, M reacts only to the finite sequence of utterances $u_0, u_1, \ldots, u_t$, with utterance u_i arriving from, for example, the Mother, at time i; with the (formal language) L to be learned $= \{u_0, u_1, \ldots\}$; and with g_t the child's t-th internal grammar *computed* by its M on input utterance sequence $u_0, u_1, \ldots, u_t$.

This leaves open for the moment what constitutes *successful* learning of a formal language L. In the next section (Sect. 1.1.2) we begin to take up this interesting topic. Further below, in the second main section entitled Complexity Considerations, i.e., Sect. 1.2, additional criteria of success are explored.

As for the plausibility of Gold's model of child language learning above, Gold [37] argued from the psycholinguistic literature (e.g., [47]) that children react to and need only *positive* data regarding languages L they learn—and that they really need *no* data regarding $\overline{L}$.

While this model of Gold's is clearly deficient regarding inputs to the child of semantic/denotational information and social reinforcers, I'll, nonetheless present below what I consider to be some resultant insights for cognitive science.

My view, argued in [19] (and to some extent in [18]) is that reality, *including cognitive reality*, has algorithmic *expected* behavior.

Hence, restricting learners to being algorithmic *from this point of view* is perfectly reasonable. Furthermore, it is a standard assumption of the field of cognitive science that cognition is algorithmic.

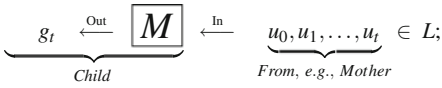

Fig. 1.1 Gold's model

1.1.2 Some Criteria of Successful Learning

Definition 1.1

- T is a *text* for $L \stackrel{\text{def}}{\Leftrightarrow} \{T(0), T(1), \ldots\} = L$.[1]
- Depicted just below is the I/O behavior of learning machine M receiving successive text elements and outputting corresponding successive grammars.

$$g_0, g_1, \ldots, g_t, \ldots \stackrel{\text{Out}}{\longleftarrow} M \stackrel{\text{In}}{\longleftarrow} T(0), T(1), \ldots, T(t), \ldots.$$

- Below are criteria from [17, 27, 37, 53] for some machine M *successfully* learning *every* language L in a class of languages $\mathscr{L}$ (see also [40])—where the g_is are from the previous bullet.
  Suppose: $\mathbb{N} = \{0, 1, 2, \ldots\}$; $\mathbb{N}^+ = \{1, 2, \ldots\}$; $b \in (\mathbb{N}^+ \cup \{*\})$; and $x \leq *$ means $x < \infty$.
  - $\mathscr{L} \in \mathbf{TxtFex}_b \Leftrightarrow (\exists$ suitably clever $M)(\forall L \in \mathscr{L})(\forall T$ for $L)(\exists t)$ $[g_t, g_{t+1}, \ldots$ each generates $L \wedge$ card$(\{g_t, g_{t+1}, \ldots\}) \leq b]$.
    We say M $\mathbf{TxtFex}_b$*-learns* (each) $L \in \mathscr{L}$. $\mathbf{TxtEx} \stackrel{\text{def}}{=} \mathbf{TxtFex}_1$, *Gold's criterion*! For example, the class $\mathscr{F}$ of all finite languages $\in \mathbf{TxtEx}$ [37].
  - $\mathscr{L} \in \mathbf{TxtBc} \Leftrightarrow (\exists M)(\forall L \in \mathscr{L})(\forall T$ for $L)(\exists t)$ $[g_t, g_{t+1}, \ldots$ each generates $L]$. We say M $\mathbf{TxtBc}$*-learns* (each) $L \in \mathscr{L}$. For example, $\mathscr{K} = \{K \cup \{x\} \mid x \in \mathbb{N}\} \in (\mathbf{TxtBc} - \mathbf{TxtFex}_*)$ [17], where K is the diagonal halting problem from [58].

1.1.2.1 $\mathbf{TxtFex}_b$-Hierarchy

Definition 1.2 Let $W_g \stackrel{\text{def}}{=}$ the language generated or enumerated by grammar or program g [58]. Informally: W_g can be thought of as the [summary of the] *behavior* of g.

I like the use of machine self-reference arguments [16], and in the next theorem (Theorem 1.3), the classes mentioned for witnessing that one $\mathbf{TxtFex}_b$ criterion has more learning power than another are each (finitarily) self-referential. The proof that they work is omitted but can be found in [17].

Theorem 1.3 [17] *Let* $\langle \cdot, \cdot \rangle$ *computably map* $\mathbb{N} \times \mathbb{N}$ *1–1 onto* $\mathbb{N}$ *[58]. Here and below* $\stackrel{\infty}{\forall} z$ *means: for all but finitely many* $z \in \mathbb{N}$ *[9]. Suppose* $n \in \mathbb{N}^+$. *Let* $\mathscr{L}_n =$ *the set of all* infinite L *such that*

$$(\exists e_1, \ldots, e_n)[W_{e_1} = \cdots = W_{e_n} = L \wedge$$

$$(\stackrel{\infty}{\forall} \langle x, y \rangle \in L)[y \in \{e_1, \ldots, e_n\}]].$$

[1] Technically, a text T for L is a mapping from $\mathbb{N}$ onto L. A text for L is then a sequence of utterances of all and only the elements of L.

Let $\mathscr{L}_* = \cup_{n \in \mathbb{N}^+} \mathscr{L}_n$. *Then* $\mathscr{L}_{n+1} \in (\mathbf{TxtFex}_{n+1} - \mathbf{TxtFex}_n) \wedge \mathscr{L}_* \in (\mathbf{TxtFex}_* - \cup_{n \in \mathbb{N}^+} \mathbf{TxtFex}_n)$.

Does, say, Noam Chomsky, for each natural language he learns, eventually vacillate between up to 42 correct grammars in his head (but for some languages no fewer)? I.e., does he need $\mathbf{TxtFex}_{42}$-learning? The problem is that we can*not* (yet) see grammars inside peoples' heads. So, we don't know if humans employ **TxtEx**-learning, $\mathbf{TxtFex}_2$-learning, $\mathbf{TxtFex}_3$-learning, …. Hence, for now, $\mathbf{TxtFex}_n$-learning, for non-astronomical $n \in \mathbb{N}$, are all not unreasonable for modeling human behavior.

1.1.2.2 Discussion

In [49] a well-documented body of *experimental* evidence indicates Mothers' utterances to young children are *not* calibrated by increasing syntactic complexity to teach children gradually (but, instead, to fit the limited attention span and processing powers of children).

These considerations are *partly* formally mirrored in the success criteria by the requirement that the learners eventually correctly learn a language L *no matter in what order the input is presented*, i.e., ($\forall T$ for L)—as long as that input contains all and only the correct sentences of the language. At this point it is important to note that ($\forall T$ for L) includes both computable and also *un*computable texts T for L! It is noted in [52] that, since the utterances of children's caretakers depend heavily on external environmental events, such influences might introduce a random component into naturally occurring texts. Whence comes the interest at all in non-computable texts. As we shall see below in Sect. 1.1.5.2, *for many important criteria of success*, learning power is nicely *un*affected by whether we allow or disallow uncomputable texts!

Children *may* be *in*sensitive to some aspects of the *order* of data presentation. In Sect. 1.1.5 several *possibilities* are considered, and some corresponding theoretical results (for some learning criteria) will be presented.

First, though, in the next section (Sect. 1.1.3), the topic of constraints on successful learning are considered.

1.1.3 Constraints on Learnability

Angluin [2] introduced the following Subset Principle for **TxtEx**-learning. It places a severe constraint on the learnable.

Theorem 1.4 (Subset Principle [2, 17]) *Suppose* $\mathbf{C} \in \{\mathbf{TxtFex}_b, \mathbf{TxtBc}\}$ *and* M $\mathbf{C}$*-learns* L.
Then ($\exists$ finite $S \subseteq L$)($\forall L' \subset L \mid S \subseteq L'$))[$M$ *does* not $\mathbf{C}$*-identify* L'].

N.B. The above Subset Principle *and* its depressing corollary just below (Corollary 1.5) hold *even if* M *is not algorithmic*! The proofs essentially depend on Baire Category (or Banach-Mazur Games) from Topology [51, 52] and *not* on algorithmicity! This is possible *only because* M in general can*not* infer data about $\overline{L}$. Case and Kötzing [26] studies which learning theory results depend only on topology, which on algorithmicity.

Angluin connected the Subset Principle to the machine learning problem of *overgeneralization*. From the proof: M *must* overgeneralize on (some Ts for) L', but it does not overgeneralize on L itself.

See [41, 68] for discussion regarding the possible connection between this subset principle and a more traditionally linguistically oriented one in [45].

Corollary 1.5 [37, 53] *If* $\mathscr{L}$ *contains an* infinite *language together with* all *its* finite *sublanguages, then* $\mathscr{L} \notin$ **TxtBc**.
Hence, for example, the class of regular languages $\notin$ **TxtBc***!*

Definition 1.6 Consider the variant of **TxtBc**-learning called **TxtBc***-learning. **TxtBc***-learning is just like **TxtBc**-learning *except* the final grammars or programs, instead of each being perfectly correct, may each make *finitely many mistakes*.

Actually:

Remark 1.1 [53] While **TxtBc*** learners can learn more than **TxtBc**-learners, the class of regular languages $\notin$ **TxtBc***!

In [17] there are variants of Theorems 1.3 and 1.4 for criteria which allow a few mistakes in final programs. In [5] there are variants of Theorem 1.7 and Remark 1.2 for criteria which allow a few mistakes in final programs. Success criteria allowing a few mistakes in the final programs are also considered below in Sect. 1.2.2.

1.1.4 Characterizations and Pattern Languages

In this section presented are some characterization results (Sect. 1.1.4.1) and important positive learnability results based on pattern languages (Sect. 1.1.4.2).

1.1.4.1 Characterizations

Program p taking two inputs i, x is a *uniform decision procedure* for a class $\mathscr{U}$ of computably decidable languages iff

$$\mathscr{U} = \{U_i \mid (\forall x)[p \text{ on input } i, x \text{ decides whether or not } x \in U_i]\}.$$

Such a $\mathscr{U}$ is called *uniformly decidable*. Important examples are all the Chomsky Hierarchy classes, Regular, …, Context-Sensitive [39], and, in the next section

(Sect. 1.1.4.2), Angluin's important class of Pattern Languages [1]. The following important characterization of Angluin *extends* the Subset Principle for the cases of *uniformly decidable* classes.

Theorem 1.7 (Angluin [2]) *Suppose* $\mathscr{U} = \{U_i \mid i \in \mathbb{N}\}$ *is uniformly decidable as above. Then* $\mathscr{U} \in$ **TxtEx** *iff there is a* computably enumerable sequence of enumerating programs *for* finite *sets* $S_0, S_1, \ldots$ *(*tell tales*) such that* $(\forall i)[S_i \subseteq U_i \ \wedge \ (\forall j \mid S_i \subseteq U_j)[U_j \not\subset U_i]]$. *The output programs of a witnessing* M can *be decision procedures.*

Remark 1.2 From [5], uniformly decidable $\mathscr{U}$ as just above is $\in$ **TxtBc** iff the associated tell tales $S_0, S_1, \ldots$ exist *but* do *not* have to be computably enumerable. In this context, the output programs of a witnessing M can*not*, in general, be decision procedures.

Remark 1.3 Angluin [2] exhibited a *uniformly decidable* $\mathscr{U}$ *with* tell tales but with *no* computably enumerable tell tales. Hence, from [5], her $\mathscr{U}$ is a *uniformly decidable class* $\in$ (**TxtBc** $-$ **TxtEx**).

1.1.4.2 Pattern Languages

Next is an ostensive definition of Angluin's important class of *Pattern Languages*.

Definition 1.8 (*Angluin* [1]) A *pattern language* is one generated by all and only the *positive* length substitutions for variables (in upper case letter alphabet) of strings (over a lower case letter alphabet) in a pattern, such as, for example, `abXYcbbZXa`.

Angluin [1] showed the class of pattern languages to be **TxtEx**-learnable, and through further papers we have the following.

Theorem 1.9 [1, 63, 71] *For each* $n \in \mathbb{N}^+$, *the uniformly decidable class of unions of* n *pattern languages* $\in$ **TxtEx***!*

These classes are not rendered unlearnable by the severe constraint of Theorem 1.4 as are the classes in the Chomsky Hierarchy (Corollary 1.5). This, in part, is because they *crosscut* the classes in the Chomsky Hierarchy. Perhaps the (somewhat ill-defined) class of natural languages is like that too. For example, most linguists consider each natural language to be infinite.

Applications of unions of n pattern languages, ranging from learning in molecular biology to more general machine learning, appear in, for example, [3, 4, 11, 50, 61, 64].

1.1.5 Insensitivities to Presentation Order

In this section we first consider some possible child insensitivities to order of presentation (Sect. 1.1.5.1); then corresponding formal results are presented (Sect. 1.1.5.2).

1.1.5.1 Order Insensitivities' Definitions

Children *may* be sensitive to the order or timing of data presentation (texts). First we present two *local* and, then, two *global* (formal) *in*sensitivities.

Definition 1.10

- M is *partly set-driven* [34, 35, 60] iff, on *sequence* $u_0, \ldots, u_t$, it reacts only to the *set* $\{u_0, u_1, \ldots, u_t\}$ of utterances *and* the *length* $t+1$ of the utterance sequence—*not* to the *order* of the *sequence*. In effect, M reacts a little to timing (but not to order) of utterances.
- M is *set-driven* [69] iff, when shown utterance *sequence* $u_0, u_1, \ldots, u_t$, it reacts only to the corresponding *set* $\{u_0, u_1, \ldots, u_t\}$— *not* to the sequence's order and length.
- M is *weakly b-ary order independent* [17] iff, for each language L on which, for *some* T for L, M converges in the limit to a finite set of grammars, there is, corresponding to L, a finite set of grammars G of cardinality $\leq b$ such that M converges to a *subset* of this *same* G for *each* T for L.
- M is *b-ary order independent* [17] iff M is weakly so, *but*, instead of converging to a *subset* of G, it converges to *exactly* G. For $b = 1$, these two notions coincide and are essentially from [8, 52].

1.1.5.2 Order Insensitivities' Results

Results regarding (partly) set-driveness for **TxtEx** (the $b = 1$ case of $\mathbf{TxtFex}_b$) are from [34, 35, 60]. For example, *set-driveness strictly limits learning power for* **TxtEx**. That, for **TxtEx**, (weakly) 1-ary order independence is without loss of learning power is essentially from [8, 34, 35]. I found the $b > 1$ cases harder to prove than the $b = 1$ case.

Theorem 1.11 [17] *Any M can be algorithmically transformed into an M' so that M' is both* partly set-driven and weakly b-ary independent *and M'* $\mathbf{TxtFex}_b$*-learns* all *the languages M does (*even if *M only learns for* computable *texts).*

As noted above in Sect. 1.1.2.2, [52] argues that, since the utterances of children's caretakers depend heavily on external environmental events, such influences might introduce a random component into naturally occurring texts. This is whence comes the interest at all in non-computable texts. The just above theorem and the results below in this section (Sect. 1.1.5.2) imply that *for the important criteria considered in this section*, learning power is nicely *un*affected by whether texts are or are not allowed to be uncomputable.

Remark 1.4 [17] The preceding theorem holds with partly set-driven *replaced by set-driven but* with $\mathbf{TxtFex}_b$-identification *restricted to only infinite languages*.

Theorem 1.12 [17] *Any M can be algorithmically transformed into an M' so that M' is b-*ary order independent *and M'* **TxtFex**$_b$*-learns* all *the languages M does (*even if *M only learns for* computable *texts).*

It is not known (it's hard to tease out experimentally) whether children exhibit any of these insensitivities, but the formal results tell us something of how they affect learning power. It has not been investigated how the results in this section would be affected if complexity considerations were taken into account.

1.1.6 Empirical U-Shaped Learning

U-Shaped Learning follows the sequence *Learn, Unlearn, Relearn*. It occurs in child development [10, 46, 66, 67], e.g., verb regularization and understanding of various (Piaget-like) conservation principles, such as temperature and weight conservation and interaction between object tracking and object permanence.

Here is an example of U-shaped learning from irregular English past tense verbs. A child first uses *spoke*, the correct past tense of the irregular verb *to speak*. Then the child ostensibly overregularizes, incorrectly using *speaked*. Lastly, the child returns to using *spoke*. The major concern of the prior cognitive science literature on U-shaped learning is in *how* one *models* U-shaped learning. For example, for language learning, by general rules or tables of exceptions [10, 46, 54, 55]? With neural nets [59] and statistical regularities or statistical irregularities?

My own concern regarding U-shaped learning is whether it is an *un*necessary and harmless accident of human evolution *or* whether U-shaped learning is advantageous in that some classes of tasks *can* be learned in the U-shaped way, but *not* otherwise?

1.1.7 Formal U-Shaped Learning

In the interest of studying whether U shapes are necessary for full learning power, it is mathematically useful to define alternatives to success criteria, including those above but in which U shapes are *forbidden* on the way to success.

Definition 1.13

- Depicted just below is the I/O behavior of learning machine M receiving successive text elements and outputting corresponding successive grammars.

$$g_0, g_1, \ldots, g_t, \ldots \overset{\text{Out}}{\longleftarrow} M \overset{\text{In}}{\longleftarrow} T(0), T(1), \ldots, T(t), \ldots .$$

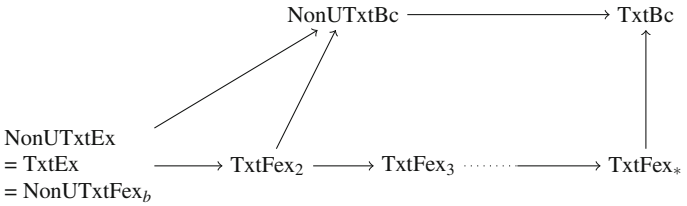

Fig. 1.2 Diagram of results

- Suppose **C** is a learning success criterion, for example, one $\in \{\mathbf{TxtFex}_b, \mathbf{TxtBc}\}$. Then, where the g_is are from the first bullet of this definition, $\mathscr{L} \in \mathbf{NonUC} \Leftrightarrow (\exists M \text{ witnessing } \mathscr{L} \in \mathbf{C})(\forall L \in \mathscr{L})(\forall T \text{ for } L)(\forall i, j, k \mid i < j < k)[W_{g_i} = W_{g_k} = L \Rightarrow W_{g_j} = L]$. *Non-U-shaped* learners of $\mathscr{L}$ *never abandon correct behaviors output on texts for* $L \in \mathscr{L}$ *and subsequently return to them.*

1.1.8 Results/Question Regarding U-Shaped Learning

The results are presented first as a diagram (Sect. 1.1.8.1) and, then, as a verbal summary of key points—with an important cognitive science question at the end (Sect. 1.1.8.2).

1.1.8.1 Diagram of Some Results

The arrow $\longrightarrow$ above denotes class *inclusion*. The transitive closure of the inclusions in Fig. 1.2 below hold *and* no other inclusions hold [6, 12, 14, 15]. For *example*, from Fig. 1.2, we have $\mathbf{TxtFex}_2$ included in $\mathbf{TxtFex}_3$ *and* properly so—regarding the latter, the transitive closure of the inclusions of Fig. 1.2 has *no* arrow from $\mathbf{TxtFex}_3$ to $\mathbf{TxtFex}_2$.

For example, from the above, there is some $\mathscr{L} \in (\mathbf{TxtFex}_3 - \mathbf{NonUTxtBc})$! This same $\mathscr{L}$ then can*not* be $\in \mathbf{NonUTxtFex}_*$—else, it would, then, be in **NonUTxtBc**. The proof regarding this $\mathscr{L}$ *does* employ an interplay between *general rules* and (finite) sets of *exceptions* [12, 14]. As mentioned above in Sect. 1.1.6, some cognitive scientists, e.g., [10, 46, 54, 55], believe this interplay underpins human U-shaped learning.

1.1.8.2 Main Results and a Question

In the present section we summarize key results from the previous section (Sect. 1.1.8.1) and, at the end, pose and discuss a difficult question for cognitive science and the evolution of human cognition.

Remark 1.5

- *Main results*:
  - From **NonUTxtBc** $\longrightarrow$ **TxtBc**, U-shaped learning *is* needed for some class in **TxtBc**.
  - From **NonUTxtEx** $=$ **TxtEx**, U-shaped learning is *not* needed for **TxtEx** learning, i.e., for learning *one* successful grammar in the limit.
  - From **NonUTxtFex**$_*$ $\longrightarrow$ **TxtFex**$_2$, U-shaped learning *is* needed for some class in **TxtFex**$_2$ even if finitely many $(*)$ grammars are allowed in the limit *but*, from **TxtFex**$_2$ $\longrightarrow$ **NonUTxtBc**, it is *not* needed if we allow infinitely many grammars in the limit.
  - From the reasoning after the previous section's (Sect. 1.1.8.1's) diagram, there *exists* $\mathscr{L} \in$ (**TxtFex**$_3$ $-$ (**NonUTxtFex**$_*$ $\cup$ **NonUTxtBc**)); in particular, U-shaped learning *is* needed for this $\mathscr{L} \in$ **TxtFex**$_3$—even if allow infinitely many grammars in the limit!
- *Question*: Does the class of tasks humans must learn to be competitive in the genetic marketplace, like this latter $\mathscr{L}$, *necessitate* U-shaped learning?
  Of course we have not yet modeled human cognition and its evolution sufficiently to answer this question. As pointed out in [12], on the *formal* modeling level, the pattern emerges that, for parameterized, cognitively relevant learning criteria, beyond very few initial parameter values, U shapes are *necessary* for full learning power! This is seen in the results just above as well as in Sect. 1.2.1.3. This latter section has, though, some important open questions very relevant to the emerging pattern.

1.2 Complexity Considerations

This section concentrates on computational complexity issues. Considered are: large database size and corresponding memory-limited learners (Sect. 1.2.1); unavoidable cases of complexity and information deficiencies of the programs learned (Sect. 1.2.2); and the complexity of learner updates (Sect. 1.2.3). In the present section a few, seemingly difficult, open questions are indicated. Some of them are important for cognitive science (Sect. 1.2.1.3); one pertains to Turing machine complexity (Sect. 1.2.3.6).

1.2.1 Hulking Databases and Memory-Limited Learners

The *Database (DB)* of utterances, $u_0, u_1, \ldots, u_{t-1}$, *prior* to time t, can, for *large* t, become an *Incredible Hulk (IH) un*pleasant to handle and query. I think of this

Fig. 1.3 The incredible hulk (IH) (©2013 Marvel and Subs)

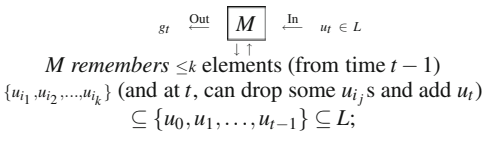

Fig. 1.4 $\leq k$-bounded example-memory model

Marvel Comics character (Fig. 1.3) as metaphorically representing the problem of DBs that are too large.

Furthermore, humans (including children) may not remember, even subconsciously, every utterance they've ever heard.

In Sects. 1.2.1.1 and 1.2.1.2 we begin to introduce some *Memory-Limited* criteria of learning success to begin to deal with these *too large DB problems*.

Then Sect. 1.2.1.3 presents corresponding results. It indicates some results of relevance to cognitive science—as well as important, cognitive science-relevant, seemingly hard open questions.

1.2.1.1 One Kind of Memory-Limited Learning Criteria

The $\leq k$-Bounded Example-Memory Model [22, 36, 52] is *without access to the complete IH DB* and is depicted at time t below in Fig. 1.4.[2] At time t, M reacts only to: u_t, g_{t-1} and its memory from time $(t-1)$ of $\leq k$ of the prior utterances from L.

We say M **TxtBem$_k$**-learns L iff, for any text of all/only the utterances of L, at some time t, g_t above in Fig. 1.4 is a grammar for L and $g_t = g_{t+1} = g_{t+2} = g_{t+3} = \ldots$.

1.2.1.2 A Second Kind of Memory-Limited Learning Criteria

For **The $\leq k$-Queries Feedback Model** [22] ($k = 0, 1$ cases: [44, 70]) the learning machine M asks *little per hypothesis update of the complete IH DB*, and it is depicted at time t below in Fig. 1.5. In this model, at time t, M reacts only to: u_t, g_{t-1} and the answers to only $\leq k$ queries it generates regarding membership in the eventually large IH DB of the past inputs $\{u_0, u_1, \ldots, u_{t-1}\}$ up through time $(t-1)$.

[2] In Fig. 1.4, for $t = 0$, M's memory of prior data at time $(t-1)$ is empty; furthermore, in Fig. 1.4 and elsewhere below, for $t = 0$, g_{t-1} can be taken to be a fixed grammar, e.g., for the empty language.

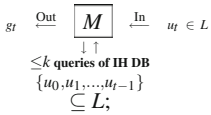

Fig. 1.5 $\leq k$-queries feedback model

M is said to **TxtFb**$_k$-learn L iff, for any text of all/only the utterances of L, at some time t, g_t above is a grammar for L and $g_t = g_{t+1} = g_{t+2} = g_{t+3} = \cdots$.

For $k = 0$ the two models coincide and the resultant learning criterion is called *iterative* or **TxtIt**-*learning* [69, 70].

1.2.1.3 Results About the Just Prior Two Models

Theorem 1.14 [22] *In* each *model, one can learn more with strictly larger k.*

Theorem 1.15 [22] *For* each $k > 0$*, for each model, there are language classes where that model is successful and the other is not!*

For each $k > 0$, for each of the two models, here is a *hard open problem*: does *forbidding* U shapes decrease learning power? For $k = 0$, the answer is *no* [28].

Even for $k = 0$, it is a hard open problem: does the case of adding the *counter* t to M's inputs u_t, g_{t-1} affect whether U shapes are necessary?

Remark 1.6 It is *known* [25], for example, for *each* $k \geq 3$, for the variant of **TxtBem**$_k$, where g_{t-1} is *not* available to M and where the memory of $\leq k$ utterances is *replaced* by memory of *any one of k-objects* (i.e., $\log_2(k)$-bits), that *forbidding* U shapes *decreases* learnability.

For $k = 1, 2$ here, U shapes are *ir*relevant [13].

Theorem 1.16 [22] *For each* $n > 0$*, the class of all* unions of n pattern languages $\in$ **TxtIt***!*

1.2.2 *Deficiencies of Programs Learned*

Independently of other complexity issues is the problem of the reasonableness of the programs eventually successfully found by a learner. This is explored in the present section for the learning of programs for total characteristic/decision functions $\mathbb{N} \to \{0, 1\}$ from *complete* data about their graphs. In prior sections we explored the mathematically different vehicle of learning programs for generating formal languages from *positive information only about those languages*. For any *total* function, once one has an enumeration of its graph, one can algorithmically generate all that is and only what is in the complement of its graph. We restrict our attention in the present

section to data presentations of such functions f in the order: $f(0), f(1), f(2), \ldots$— since from any presentation order of the graph of f (i.e., $\{(x, y) \mid f(x) = y\}$) one can algorithmically generate the presentation order $f(0), f(1), f(2), \ldots$.

In Sects. 1.2.2.1–1.2.2.3, the basic background material is presented. Then, in Sect. 1.2.2.4, results about inherent *computational complexity deficiencies* in programs learned are presented. Finally, in Sect. 1.2.2.5, some unexpected, inherent *informational deficiencies* of programs learned are laid out.

1.2.2.1 Successful Learning on Complete Data Regarding Total Functions

Suppose $\mathscr{S} \subseteq \mathscr{R}_{0,1}$, *the class of all (total) computable characteristic functions of subsets of* N. Defined just below are criteria of success analogs of **TxtEx** and **TxtBc** — but regarding learning programs for characteristic functions instead of grammars for languages.

Definition 1.17

- Depicted below is the I/O behavior of a learning machine M receiving successive values of a function f and outputting corresponding successive programs. Some later items in this definition refer to this depiction.

$$p_0, p_1, \ldots, p_t, \ldots \overset{\text{Out}}{\longleftarrow} M \overset{\text{In}}{\longleftarrow} f(0), f(1), \ldots, f(t), \ldots .$$

- Suppose $a \in \mathbb{N} \cup \{*\}$. a is for *a*nomaly count. For $a = *$, a stands for *finitely many*.[3]
- $\mathscr{S} \in \mathbf{Ex}^a \Leftrightarrow (\exists \text{ suitably clever } M)(\forall f \in \mathscr{S})$ [M, fed $f(0), f(1), \ldots$, outputs $p_0, p_1, \ldots \wedge (\exists t)[p_t = p_{t+1} = \cdots \wedge p_t$ *computes* f*—except at up to* a *inputs*]].
- $\mathscr{S} \in \mathbf{Bc}^a \Leftrightarrow (\exists M)(\forall f \in \mathscr{S})$ [M, fed $f(0), f(1), \ldots$, outputs $p_0, p_1, \ldots \wedge (\exists t)[p_t, p_{t+1}, \ldots$ each computes f—except at up to a inputs]].

1.2.2.2 Examples

Below are some important example classes together with some known results about them which will help us understand the results in Sects. 1.2.2.4 and 1.2.2.5 on the deficiencies of programs learned.

[3] Case and Smith [29, 30] motivate by anomalous dispersion from physical optics the presence of $a > 0$ anomalies in "successful" final programs. We omit herein details about that.

Remark 1.7

- For $k \geq 1$, let $\mathscr{P}^k =$ the class all 0–1 valued functions computable by (multi-tape) TMs in $O(n^k)$ time, with $n =$ input length. Let $\mathscr{P} = \bigcup \mathscr{P}^k$, the class of polynomial time computable characteristic functions.
- Let slow be a *fixed* slow-growing unbounded function $\in \mathscr{P}^1$, e.g., $\leq$ Ackermann^{-1} [31]. Let $\mathscr{Q}^k =$ the class all $\{0, 1\}$-valued functions computable in $O(n^k \cdot \log(n) \cdot \text{slow}(n))$ time, again with $n =$ input length. We have,

$$\underbrace{\mathscr{P}^k \subset \mathscr{Q}^k}_{\text{Tightest known separation [38, 39]}} \subset \mathscr{P}^{k+1}.$$

- $\mathscr{P} \in \mathbf{Ex}^0$. $\mathscr{P}^k \in \mathbf{Ex}^0$ too (*with each output conjecture running in k-degree poly time*); $\mathscr{CF}$, the class all *characteristic functions of* context-free languages, $\in \mathbf{Ex}^0$ [37].
- $\mathbf{Ex}^0 \subset \mathbf{Ex}^1 \subset \mathbf{Ex}^2 \subset \cdots \subset \mathbf{Ex}^* \subset \mathbf{Bc}^0 \subset \mathbf{Bc}^1 \subset \cdots \subset \mathbf{Bc}^*$ [29]. Hence, more anomalies tolerated in final successful programs entails strictly more learning power.
- Harrington in [30]: $\mathscr{R}_{0,1} \in \mathbf{Bc}^*$.

1.2.2.3 Basic Notation

Next is some basic terminology important to the statements of results in Sects. 1.2.2.4 and 1.2.2.5.

Definition 1.18

- $\mathscr{C}\text{of} = \{f \in \mathscr{R}_{0,1} \mid (\overset{\infty}{\forall} x)[f(x) = 1]\}$ ($\subset \mathscr{P}^1$ and $\mathscr{REG}$, the class all *characteristic functions of* regular languages). As will be seen, $\mathscr{C}\text{of}$ is important as a nice example of a particularly trivial, easily learnable class of (characteristic) functions.
- $\varphi_p^{\text{TM}} =$ the partial computable function $\mathbb{N} \to \mathbb{N}$ computed by (multi-tape) Turing machine program (number) p.
- $\Phi_p^{\text{TM}}(x) =$ the *runtime* of Turing machine program (number) p on input x, if p halts on x, and is undefined otherwise.
- $\Phi_p^{\text{WS}}(x) =$ the tape work space used by Turing machine program (number) p on input x, if p halts on x, and is undefined otherwise.
- $f[m] =$ the *sequence* $f(0), \ldots, f(m-1)$; for $m = 0$, $f[m]$ is the empty sequence.
- $M(f[m]) = M$'s output based on $f[m]$. For the results herein, we may and will suppose without loss of generality that $M(f[m])$ is always defined.
- $M(f)$ denotes M's *final* output program on input f, *if any*; else, it is undefined.

1.2.2.4 Complexity Deficiencies of Programs Learned

First is a simple positive result which, in effect, is part of the third bullet in Remark 1.7 in Sect. 1.2.2.2. This result will contrast nicely with the theorem just after it.

Proposition 1.19 [20] *For each* $k \geq 1$, $\exists M$ *witnessing* $\mathscr{P}^k \in \mathbf{Ex}^0$ *such that* $(\forall f)(\forall m)$ $[\Phi^{\mathrm{TM}}_{M(f[m])}(x) \in O(|x|^k)]$.

Theorem 1.20 ([20], tightens [65]) *Suppose* $k \geq 1$ *and that* M *witnesses either* $\mathscr{Q}^k \in \mathbf{Ex}^*$ *or* $\mathscr{Q}^k \in \mathbf{Bc}^0$ *(*special case*: M witnesses* $\mathscr{Q}^k \in \mathbf{Ex}^0$*).*
Then: $(\exists f \in \mathscr{C}\text{of})(\forall k\text{-}$*degree polynomials* $p)$
$(\overset{\infty}{\forall} n)(\overset{\infty}{\forall} x)[\Phi^{\mathrm{TM}}_{M(f[n])}(x) > p(|x|)]$.

Idea: if one ups the generality of a learner M from $\mathscr{P}^k$ to $\mathscr{Q}^k$, then the run-times of M's successful outputs on some trivial $f \in \mathscr{C}\text{of}$ is *worse* than *any* preassigned k-degree polynomial bound, a complexity deficiency.

Of course, since $\mathscr{REG}$ is defined in terms of finite automata [32, 39], *no* Work Space is needed to compute the functions in $\mathscr{REG}$ (which includes the trivial functions in $\mathscr{C}\text{of}$), but:

Theorem 1.21 [20] *For* $k \geq 1$ *and* M *witnessing* $\mathscr{CF} \in \mathbf{Ex}^*$ *(*special case*: M witnesses* $\mathscr{CF} \in \mathbf{Ex}^0$*),* $(\exists f \in \mathscr{C}\text{of})(\exists x)[\Phi^{\mathrm{WS}}_{M(f)}(x) \geq k]$.

Idea: learning $\mathscr{CF}$ instead of $\mathscr{REG}$ produces some final programs on some trivial functions in $\mathscr{C}\text{of}$ which *do* require some work space, another complexity deficiency.

1.2.2.5 Information Deficiencies of Programs Learned

First is a positive result which will contrast nicely with the theorem just after it.

Theorem 1.22 [20] $\exists M$ outputting only total poly-time conjectures *and witnessing* both $\mathscr{P} \in \mathbf{Bc}^*$ and $\mathscr{P}^1 \in \mathbf{Ex}^0$ *such that* $(\forall f \in \mathscr{P}^1)[\Phi^{\mathrm{TM}}_{M(f)}(x) \in O(|x|)]$.

In the next theorem f.o. **PA** is the version of Peano Arithmetic expressed within first order logic [48, 58]. **PA** is a well-known formal theory in which one can express and prove all the theorems in an *elementary* number theory book.

For a sentence E expressible in **PA**, $\ll E \gg$ is some natural translation of E into the language of **PA**.

In the next theorem, the example

$$E = \varphi^{\mathrm{TM}}_{M(f[m])} \text{ is computable}^* \text{ in } O(|x|^k)\text{time}$$

contains what would, without explanation, be a mysterious $*$. This E is meant to be an abbreviation of the clearer, but longer sentence: any total, finite variant of the function computed by the TM-program output by M on $f[m]$ runs in time $O(n^k)$, where $n =$ the length of this program's input.

Theorem 1.23 [20]
Suppose theory **T** *is any* true, computably axiomatized *extension of f.o.* **PA** *(so,* **T** *is a* safe, algorithmic *information extractor). Suppose* $k \geq 1$ *and M witnesses* $\mathscr{Q}^k \in \mathbf{Bc}^*$. *Then:* $(\exists f \in \mathscr{C}\text{of})(\overset{\infty}{\forall} m)[\mathbf{T} \not\vdash \ll \varphi^{\text{TM}}_{M(f[m])}$ *is computable* in* $O(|x|^k)$ *time* $\gg]$.

It is particularly interesting to apply the preceding theorem (Theorem 1.23) to the case of a learner M from Theorem 1.22 earlier: while the programs learned by *this* M for functions in all of $\mathscr{C}$of are perfectly correct and *do* have excellent, linear-time run-times, some of these programs learned will be *informationally deficient*—since we cannot prove in **T** from *them* even considerably weaker upper bounds on the run-times of *any* total finite variants of the functions they compute!

1.2.3 Complexity of Learner Updates

This section begins with a general discussion of the important and well-investigated subject of learner update complexity (Sect. 1.2.3.1). Then an automata-theory-based paradigm is chosen for both illustration and since it may be new to the Grammatical Inference community as well as interesting to it (Sects. 1.2.3.2–1.2.3.6).

1.2.3.1 Complexity-Restricted Updates of Learners Generally

There has been tremendous interest in the Grammatical Inference community, e.g., [33], in *polynomial time updating of each learned output conjecture*, where the polynomials are typically in some measure of the size of data used to obtain the conjecture.

For example, the Pattern Languages are polytime **TxtIt**-learnable [43]—but at the interesting, apparently necessary cost of intermediate conjectures which do not generate some of the data on which they are based!

Pitt [56] notes that it's possible to *cheat* and always obtain (meaningless) polytime updates: Suppose q is a polynomial, and M on finite data sequence σ delivers its correspondingly conjectured program within time $q(|\sigma|)$. Pitt notes M can put off outputting conjectures based on σ until it has seen much larger data τ so that $q(|\tau|)$ is enough time for M to work on σ as long as it needs. He notes that this delaying trick is *un*fair—since it allows as much to be learned as in the case of no time bound on the learner updates.

Finding mathematical conditions to *guarantee no Pitt delaying tricks can be used* seems very difficult [24]. For a time I believed polytime $\mathbf{TxtFb}_k$-learning is fair—until Frank Stephan provided me a counterexample. I currently believe polytime $\mathbf{TxtBem}_k$-learning is reasonably fair—provided output conjectures are not padded up too much to carry over extra information from one update to the next. Reasonably fair such padding was employed for **TxtIt**-learning of some mildly context-sensitive classes in [7]. Most published polytime update learning algorithms I've seen seem

quite fair, e.g., the **TxtEx**-learning of somewhat different mildly context-sensitive classes in [72] and, of course, the polytime **TxtIt**-learnability of the Pattern Languages mentioned above.

Next are presented, for example, some recent results about the use of learning functions whose graphs are regular (i.e., finite automata-acceptable). These are called *automatic* learning functions and are defined more formally later in Sect. 1.2.3.3.

Perhaps this material will be new to the Grammatical Inference community and of some interest to them. We believe finite automata are not smart enough to do Pitt tricks.

1.2.3.2 Automatic Structures

The study of automatic structures (defined below) began with the Program of Khoussainov and Nerode [42]: replace TMs by *finite automata* in computable model theory.

Below we provide some basic definitions regarding automatic structures which will lead, for example, to Remark 1.8 in Sect. 1.2.3.3 in which we are able to define automatic learning function and present a few results.

Definition 1.24

- The *automatic 1-ary relations* are the *regular (i.e., finite automata accepted) languages* $\subseteq \Sigma^*$, for some finite (non-empty) alphabet Σ.
- For our next defining *automatic binary relations* R over a finite alphabet $\Upsilon (\supseteq \Sigma)$, we need be able to *submit* a pair $(\alpha, \beta) \in (\Upsilon^* \times \Upsilon^*)$ to a finite automaton.
- *If* we feed α and, subsequently, β, the finite automaton will have trouble remembering much about α upon receiving β, *so* we use *convolution* to submit α and β *together*—details next.
- Suppose $\sqcup$ is a "blank" symbol $\notin \Upsilon$. We provide next an ostensive definition of $\text{conv}(\alpha, \beta)$.
  Example: $\text{conv}(ab, bba) = (a, b)(b, b)(\sqcup, a)$. Idea: these *pairs* are each new *single alphabet symbols* to be *sequentially* read by a finite automaton.
- We say a *binary* relation R is *automatic* iff $\{\text{conv}(\alpha, \beta) \mid R(\alpha, \beta)\}$ is *regular*—over the alphabet $((\Upsilon \cup \{\sqcup\}) \times (\Upsilon \cup \{\sqcup\}))$.
  The concept obviously generalizes to k-ary relations, and we consider it to be so generalized.

1.2.3.3 Automatic Classes and Functions

Remark 1.8

- $\mathscr{L}$ is said to be an *automatic class* iff each $L \in \mathscr{L}$ is a subset of Σ^*, and, for some *regular* index domain I and some *automatic* $S \subseteq (I \times \Sigma^*)$,
  $\mathscr{L} = \{L_\alpha \mid \alpha \in I\}$, where (the then regular) $L_\alpha = \{x \mid (\alpha, x) \in S\}$.
  Idea: such a $\mathscr{L}$ is *uniformly* regular.

- Automatic classes are the *automatic analog* of the *uniformly decidable classes*, e.g., from Angluin's characterization theorem, Theorem 1.7 from Sect. 1.1.4.1. The analog of the index domain above for the uniformly decidable classes is just $\mathbb{N}$.
- An *automatic function* M is an automatic, single-valued relation. I.e., a function M is *automatic* iff the relation $\{(\alpha, \beta) \mid M(\alpha) = \beta\}$ is automatic.
- In the later, learnability portion of this section, we are interested in modeling *learners* as *automatic and, also, as more general functions* M. Although more general functions can learn more, and
- Case et al. [21] notes there are *significant* automatic classes based on *erasing regular* patterns which are learnable by mere automatic learners.[4]

1.2.3.4 TM Input/Output of Automatic Functions

A finite automaton *accepting (the convolutions of) all/only the ordered pairs in the graph of an automatic function* M is *not* at all the same as *computing the value of* $M(x)$ *from each* x *in domain*(M). However:

Theorem 1.25 [23] *Suppose* M *is an* automatic function. *Then there is a* linear-time bounded, one-tape, deterministic TM *which* computes M *where each input* x *is given on the tape starting from the* marked left end *and output* $M(x)$ *starts from this* same left end.

This theorem has a strong converse as follows.

Theorem 1.26 [23] *Suppose a* linear-time bounded, one-tape, non-deterministic TM computes M *such that each input* x *is given on the tape starting from the* marked left end*; on each non-deterministic path the output is* $M(x)$ *or?, where at least one path has output* $M(x)$*; and outputs* $M(x)$ *or? start from this* same left end. *Then* M *is automatic. (Left end I/O is provably crucial.)*

1.2.3.5 Relevant General Learnability Definitions

We are next interested in the learnability (general or otherwise) of *automatic classes*. The next definition begins to explore how to handle this.

Definition 1.27

- As above, a *text* T for $L \subseteq \Sigma^*$ (in automatic class) $\mathscr{L}$ is a sequence of all and only the elements of L.
- A learner employs output *hypotheses* $\text{hyp}_t \in I$ (I, the corresponding index set) and a sequence of long-term memories mem_t (each $\in \Gamma^*$).

[4] Above, in Sect. 1.1.4.2, we defined Angluin's pattern languages based on patterns, and *there* only *positive* length substitutions are allowed. The provably hard to learn *erasing* pattern languages [57] also allow empty substitutions. The *regular* ones [62] require that each variable in the associated defining pattern be present only once.

- A learner has initial long-term memory mem_0 (and initial hypothesis hyp_0). Think of each $t = 0, 1, \ldots$ as a time/cycle. Then *learner* $M : (T(t), \text{mem}_t) \mapsto (hyp_{t+1}, \text{mem}_{t+1})$.
- $\mathscr{L}$ is *learnable* by M: $(\forall L \in \mathscr{L})(\forall T \text{ for } L)(\exists t)(\forall t' > t)[\text{hyp}_{t'} = \text{hyp}_t \ \wedge \ \text{hyp}_t$ is correct for $L]$.

Remark 1.9

- With *un*restricted memory and *M*s *TM computable*, the preceding *learnability* criterion (*for automatic classes*) is equivalent to **TxtEx** (but with hypotheses restricted to I).
- Curiously: learnability *of automatic classes* does *not* change even if the *M*s can *also* be *un*computable, and Angluin's Characterization can *in this case* omit computable enumerability of tell tales.

1.2.3.6 Linear Time Learners Suffice!

We explore next how efficiently the general learning *of automatic classes* can be done. First we introduce a relevant TM model.

Definition 1.28 Our $(k + 1)$-Tape TM Model:

- Tape 0 (base tape): At *the beginning of cycle t*, it contains $\text{conv}(T(t), \text{mem}_t)$ (with $|\text{mem}_t| \leq$ longest text datum seen so far + a constant).
  At *end of cycle t*, it contains $\text{conv}(\text{hyp}_{t+1}, \text{mem}_{t+1})$ (with $|\text{mem}_{t+1}| \leq$ longest text datum seen so far + a constant). *Marked left end I/O* too.
- Additional Tapes $1, 2, \ldots, k$: *normal work-tapes*, with contents and head position *not* modified during *change* of cycle.
- *Each cycle* of the machine runs in the *linear-time* in the length of the longest text datum seen so far.

Theorem 1.29 [23] *Every* learnable automatic class *has* such *a* linear-time TM learner *employing only* $k =$ two *additional* work-tapes.
The two work tapes can be replaced by two stacks *or, instead,* one queue—a queue with non-overtaking, one-way heads to operate each end*!*

Open problem: does only $k =$ *one* additional *work tape* suffice?

1.3 Summary Including Open Problems

We briefly summarize what's been done in the present chapter.

As noted above, the chapter is about Gold's 1967 model of child language learning and selected highlights of the theory appearing since Gold's 1967 paper. It is divided into two major sections. The first concentrates on results that are independent of computational complexity considerations; the second concentrates on relevant complexity issues.

In the first major section are treated: Gold's Model; criteria of successful learning (with hierarchy results); reasonableness and weaknesses of the general model; provable severe constraints on learnability; related characterizations and the important example of Angluin's Pattern Languages; two local and two global formal insensitivities of learning to order of presentation of data with corresponding formal results, with no available conclusions about how these do or do not apply to human children; and the cognitive science empirically observed phenomenon of U-shaped learning with mathematically defined formal analogs, and corresponding interpreted formal results that suggest the U shape may be necessary for full human learning power. Also presented are some hard questions for the state of the art in empirical cognitive science: do some humans, after successfully learning at least some natural languages, vacillate between multiple grammars. Is U-shaped learning necessary for those humans who succeed in the genetic marketplace?

In the second major section on complexity considerations are treated: the problem of learning based on infeasibly large amounts of data, and presented are two formal models of data memory-limited learners, accompanied by corresponding formal hierarchy results; surprising results showing that some slight increases in learning generality inexorably lead to both complexity and information deficiencies in the programs that are learned for some very simple objects to be learned, but with no such Deficiencies appearing for the less general learning cases; and the important feasibly computable learner updates where it is noted that sometimes the feasibility of updates can be a cheat, and detailed results are given about both the use of finite automata accepted learning functions (where it is ostensibly difficult to cheat) and the learnability of uniformly finite automata accepted classes. Also presented are some seemingly difficult open mathematical questions (the first two are of relevance to cognitive science, the third to complexity theory): For each $k > 0$, for each of the two data memory-limited models of Sect. 1.2.1, does forbidding U shapes decrease learning power? For each $k \geq 0$, for these two models, does the case of supplying the counter t to M in addition to its inputs u_t, g_{t-1} affect whether U shapes are necessary? For Theorem 1.29 in Sect. 1.2.3.6, does only $k = 1$ additional work tape suffice?

Acknowledgments My thanks go to Sanjay Jain and Frank Stephan for catching a mistake in an earlier draft. I'm also grateful to an anonymous referee for excellent suggestions which I hope I've been able to follow sufficiently well in this chapter.

References

1. Angluin, D.: Finding patterns common to a set of strings. Journal of Computer and System Sciences **21**, 46–62 (1980)
2. Angluin, D.: Inductive inference of formal languages from positive data. Information and Control **45**, 117–135 (1980)
3. Arikawa, S., Miyano, S., Shinohara, A., Kuhara, S., Mukouchi, Y., Shinohara, T.: A machine discovery from amino-acid-sequences by decision trees over regular patterns. New Generation

Computing **11**, 361–375 (1993)
4. Arikawa, S., Shinohara, T., Yamamoto, A.: Learning elementary formal systems. Theoretical Computer Science **95**, 97–113 (1992)
5. Baliga, G., Case, J., Jain, S.: The synthesis of language learners. Information and Computation **152**, 16–43 (1999)
6. Baliga, G., Case, J., Merkle, W., Stephan, F., Wiehagen, W.: When unlearning helps. Information and Computation **206**, 694–709 (2008)
7. Becerra-Bonache, L., Case, J., Jain, S., Stephan, F.: Iterative learning of simple external contextual languages. Theoretical Computer Science **411**, 2741–2756 (2010). Special Issue for *ALT'08*
8. Blum, L., Blum, M.: Toward a mathematical theory of inductive inference. Information and Control **28**, 125–155 (1975)
9. Blum, M.: A machine independent theory of the complexity of recursive functions. Journal of the ACM **14**, 322–336 (1967)
10. Bower, M.: Starting to talk worse: Clues to language development from children's late speech errors. In: S. Strauss, R. Stavy (eds.) U-Shaped Behavioral Growth, Developmental Psychology Series. Academic Press, NY (1982)
11. Brazma, A., Jonassen, I., Eidhammer, I., Gilbert, D.: Approaches to the automatic discovery of patterns in biosequences. Journal of Computational Biology **5**(2), 279–305 (1998)
12. Carlucci, L., Case, J.: On the necessity of U-shaped learning. Topics in Cognitive Science (2012). Invited for Special Issue on Formal Learning Theory
13. Carlucci, L., Case, J., Jain, S., Stephan, F.: Memory-limited U-shaped learning. Information and Computation **205**, 1551–1573 (2007)
14. Carlucci, L., Case, J., Jain, S., Stephan, F.: Non U-shaped vacillatory and team learning. Journal of Computer and System Sciences **74**, 409–430 (2008). Special issue in memory of Carl Smith
15. Carlucci, L., Jain, S., Kinber, E., Stephan, F.: Variations on U-shaped learning. Information and Computation **204**(8), 1264–1294 (2006)
16. Case, J.: Infinitary self-reference in learning theory. Journal of Experimental and Theoretical Artificial Intelligence **6**, 3–16 (1994)
17. Case, J.: The power of vacillation in language learning. SIAM Journal on Computing **28**(6), 1941–1969 (1999)
18. Case, J.: Directions for computability theory beyond pure mathematical. In: D. Gabbay, S. Goncharov, M. Zakharyaschev (eds.) Mathematical Problems from Applied Logic II. New Logics for the XXIst Century, International Mathematical Series, Vol. 5, pp. 53–98. Springer (2007). Invited book chapter
19. Case, J.: Algorithmic scientific inference: Within our computable expected reality. International Journal of Unconventional Computing **8**(3), 192–206 (2012). Invited journal expansion of an invited talk and paper at the *3rd International Workshop on Physics and Computation 2010*
20. Case, J., Chen, K., Jain, S., Merkle, W., Royer, J.: Generality's price: Inescapable deficiencies in machine-learned programs. Annals of Pure and Applied Logic **139**, 303–326 (2006)
21. Case, J., Jain, S., , Le, T., Ong, Y., Semukhin, P., Stephan, F.: Automatic learning of subclasses of pattern languages. Information and Computation **218**, 17–35 (2012)
22. Case, J., Jain, S., Lange, S., Zeugmann, T.: Incremental concept learning for bounded data mining. Information and Computation **152**, 74–110 (1999)
23. Case, J., Jain, S., Seah, S., Stephan, F.: Automatic functions, linear time and learning. In: S. Cooper, A. Dawar, B. Löwe (eds.) How the World Computes - Turing Centenary Conference and Eighth Conference on Computability in Europe (CiE 2012), Proceedings, *Lecture Notes In Computer Science*, vol. 7318, pp. 96–106. Springer, Berlin (2012)
24. Case, J., Kötzing, T.: Difficulties in forcing fairness of polynomial time inductive inference. In: 20th International Conference on Algorithmic Learning Theory (ALT'09), *Lecture Notes in Artificial Intelligence*, vol. 5809, pp. 263–277 (2009)
25. Case, J., Kötzing, T.: Memory-limited non-U-shaped learning with solved open problems. Theoretical Computer Science (2012). In press online at doi:10.1016/j.tcs.2012.10.010, Special Issue for selected *ALT'10* papers

26. Case, J., Kötzing, T.: Topological separations in inductive inference. In: S. Jain, et al. (eds.) 24nd International Conference on Algorithmic Learning Theory (ALT'13), *Lecture Notes in Artificial Intelligence*, vol. 8139, pp. 128–142 (2013)
27. Case, J., Lynes, C.: Machine inductive inference and language identification. In: M. Nielsen, E. Schmidt (eds.) Proceedings of the 9th International Colloquium on Automata, Languages and Programming, *Lecture Notes in Computer Science*, vol. 140, pp. 107–115. Springer-Verlag, Berlin (1982)
28. Case, J., Moelius, S.: U-shaped, iterative, and iterative-with-counter learning. Machine Learning **72**, 63–88 (2008). Special issue for *COLT'07*
29. Case, J., Smith, C.: Anomaly hierarchies of mechanized inductive inference. In: Symposium on the Theory of Computation, pp. 314–319 (1978)
30. Case, J., Smith, C.: Comparison of identification criteria for machine inductive inference. Theoretical Computer Science **25**, 193–220 (1983)
31. Cormen, T., Leiserson, C., Rivest, R., Stein, C.: Introduction to Algorithms, second edn. MIT Press (2001)
32. Davis, M., Sigal, R., Weyuker, E.: Computability, Complexity, and Languages, second edn. Academic Press (1994)
33. de la Higuera, C.: Grammatical Inference: Learning Automata and Grammars. Cambridge University Press (2010)
34. Fulk, M.: A study of inductive inference machines. Ph.D. thesis, SUNY at Buffalo (1985)
35. Fulk, M.: Prudence and other conditions on formal language learning. Information and Computation **85**, 1–11 (1990)
36. Fulk, M., Jain, S., Osherson, D.: Open problems in Systems That Learn. Journal of Computer and System Sciences **49**(3), 589–604 (1994)
37. Gold, E.: Language identification in the limit. Information and Control **10**, 447–474 (1967)
38. Hennie, F., Stearns, R.: Two-tape simulation of multitape Turing machines. J. ACM **13**, 433–446 (1966)
39. Hopcroft, J., Ullman, J.: Introduction to Automata Theory, Languages, and Computation. Addison-Wesley Publishing Company (1979)
40. Jain, S., Osherson, D., Royer, J., Sharma, A.: Systems that Learn: An Introduction to Learning Theory, second edn. MIT Press, Cambridge, Mass. (1999)
41. Kapur, S., Lust, B., Harbert, W., Martohardjono, G.: Universal grammar and learnability theory: The case of binding domains and the 'subset principle'. In: E. Reuland, W. Abraham (eds.) Knowledge and Language, vol. I, pp. 185–216. Kluwer (1993)
42. Khoussainov, B., Nerode, A.: Automatic presentations of structures. In: Logical and Computational Complexity (International Workshop LCC 1994), *LNCS*, vol. 960, pp. 367–392. Springer (1995)
43. Lange, S., Wiehagen, R.: Polynomial time inference of arbitrary pattern languages. New Generation Computing **8**, 361–370 (1991)
44. Lange, S., Zeugmann, T.: Incremental learning from positive data. Journal of Computer and System Sciences **53**, 88–103 (1996)
45. Manzini, R., Wexler, K.: Parameters, binding theory and learnability. Linguistic Inquiry **18**, 413–444 (1987)
46. Marcus, G., Pinker, S., Ullman, M., Hollander, M., Rosen, T., Xu, F.: Overregularization in Language Acquisition. Monographs of the Society for Research in Child Development, vol. 57, no. 4. University of Chicago Press (1992). Includes commentary by H. Clahsen
47. McNeill, D.: Developmental psycholinguistics. In: F. Smith, G.A. Miller (eds.) The Genesis of Language, pp. 15–84. MIT Press (1966)
48. Mendelson, E.: Introduction to Mathematical Logic, fifth edn. Chapman & Hall, London (2009)
49. Newport, E., Gleitman, L., Gleitman, H.: Mother I'd rather do it myself: Some effects and noneffects of maternal speech style. In: C. Snow, C. Ferguson (eds.) Talking to children: Language input and acquisition, pp. 109–150. Cambridge University Press (1977)
50. Nix, R.: Editing by examples. Tech. Rep. 280, Department of Computer Science, Yale University, New Haven, CT, USA (1983)

51. Osherson, D., Stob, M., Weinstein, S.: Note on a central lemma of learning theory. Journal of Mathematical Psychology **27**, 86–92 (1983)
52. Osherson, D., Stob, M., Weinstein, S.: Systems that Learn: An Introduction to Learning Theory for Cognitive and Computer Scientists. MIT Press, Cambridge, Mass. (1986)
53. Osherson, D., Weinstein, S.: Criteria of language learning. Information and Control **52**, 123–138 (1982)
54. Pinker, S.: Language Learnability and Language Development. Harvard University Press (1984)
55. Pinker, S.: Rules of language. Science **253**, 530–535 (1991)
56. Pitt, L.: Inductive inference, DFAs, and computational complexity. In: Analogical and Inductive Inference, Proceedings of the Second International Workshop (AII'89), *Lecture Notes in Artificial Intelligence*, vol. 397, pp. 18–44. Springer-Verlag, Berlin (1989)
57. Reidenbach, D.: A negative result on inductive inference of extended pattern languages. In: N. Cesa-Bianchi, M. Numao (eds.) Proceedings of The 13th International Conference on Algorithmic Learning Theory (ALT'02), Lecture Notes in Artificial Intelligence, pp. 308–320. Springer (Lübeck, Germany, November, 2002)
58. Rogers, H.: Theory of Recursive Functions and Effective Computability. McGraw-Hill, New York (1967). Reprinted, MIT Press, 1987
59. Rogers, T.T., Rakinson, D.H., McClelland, J.L.: U-shaped curves in development: a PDP approach. Journal of Cognition and Development **5**(1), 137–145 (2004)
60. Schäfer-Richter, G.: Über eingabeabhängigkeit und komplexität von inferenzstrategien. Ph.D. thesis, RWTH Aachen (1984)
61. Shimozono, S., Shinohara, A., Shinohara, T., Miyano, S., Kuhara, S., Arikawa, S.: Knowledge acquisition from amino acid sequences by machine learning system BONSAI. Trans. Information Processing Society of Japan **35**, 2009–2018 (1994)
62. Shinohara, T.: Polynomial time inference of extended regular pattern languages. In: RIMS Symposia on Software Science and Engineering, Kyoto, Japan, *Lecture Notes in Computer Science*, vol. 147, pp. 115–127. Springer-Verlag (1982)
63. Shinohara, T.: Inferring unions of two pattern languages. Bulletin of Informatics and Cybernetics **20**, 83–88. (1983)
64. Shinohara, T., Arikawa, A.: Pattern inference. In: K.P. Jantke, S. Lange (eds.) Algorithmic Learning for Knowledge-Based Systems, *Lecture Notes in Artificial Intelligence*, vol. 961, pp. 259–291. Springer-Verlag (1995)
65. Sipser, M.: (1978). Private communication
66. Strauss, S., Stavy, R. (eds.): U-Shaped Behavioral Growth. Developmental Psychology Series. Academic Press, NY (1982)
67. Strauss, S., Stavy, R., Orpaz, N.: The child's development of the concept of temperature (1977). Unpublished manuscript, Tel-Aviv University
68. Wexler, K.: The subset principle is an intensional principle. In: E. Reuland, W. Abraham (eds.) Knowledge and Language, vol. I, pp. 217–239. Kluwer (1993)
69. Wexler, K., Culicover, P.: Formal Principles of Language Acquisition. MIT Press, Cambridge, Mass (1980)
70. Wiehagen, R.: Limes-Erkennung rekursiver Funktionen durch spezielle Strategien. Elektronische Informationverarbeitung und Kybernetik **12**, 93–99 (1976)
71. Wright, K.: Identification of unions of languages drawn from an identifiable class. In: R. Rivest, D. Haussler, M. Warmuth (eds.) Proceedings of the Second Annual Workshop on Computational Learning Theory, Santa Cruz, California, pp. 328–333. Morgan Kaufmann Publishers, Inc. (1989)
72. Yoshinaka, R.: Learning multiple context-free languages with multidimensional substitutability from positive data. Information and Computation **412**, 1821–1831 (2011)