Synthesizing Learners Tolerating Computable Noisy Data

John Case
Department of CIS
University of Delaware
Newark, DE 19716, USA
Email: case@cis.udel.edu

Sanjay Jain
School of Computing
National University of Singapore
Singapore 119260
Email: sanjay@comp.nus.edu.sg

Proposed Running Head: Synthesizing Learners

Send all Correspondence to:
Sanjay Jain
School of Computing
National University of Singapore
Singapore 119260
Republic of Singapore
Email: sanjay@comp.nus.edu.sg

Abstract

An *index for an r.e. class of languages* (by definition) generates a sequence of grammars defining the class. An *index for an indexed family of recursive languages* (by definition) generates a sequence of decision procedures defining the family.

F. Stephan's model of noisy data is employed, in which, roughly, correct data crops up infinitely often, and incorrect data only finitely often.

In a computable universe, all data sequences, even noisy ones, are computable. New to the present paper is the restriction that noisy data sequences be, nonetheless, computable. This restriction is interesting since we may live in a computable universe.

Studied, then, is the synthesis *from indices for r.e. classes and for indexed families of recursive languages* of various kinds of *noise-tolerant* language-learners for the corresponding classes or families indexed, *where the noisy input data sequences are restricted to being computable.*

Many positive results, as well as some negative results, are presented regarding the existence of such synthesizers.

The main positive result is: grammars for *each* indexed family *can* be learned behaviorally correctly from *computable*, noisy, positive data. The proof of another positive synthesis result yields, as a pleasant corollary, a strict subset-principle or tell-tale style characterization, for the *computable* noise-tolerant behaviorally correct learnability of grammars from positive and negative data, of the corresponding families indexed.

1 Introduction

Consider the scenario in which a subject is attempting to learn its environment. At any given time, the subject receives a finite piece of data about its environment, and based on this finite information, conjectures an explanation about the environment. The subject is said to *learn* its environment just in case the explanations conjectured by the subject become fixed over time, and this fixed explanation is a correct representation of the subject's environment. Inductive Inference, a subfield of computational learning theory, provides a framework for the study of the above scenario when the subject is an algorithmic device. The above model of learning is based on the work initiated by Gold [Gol67] and has been used in inductive inference of both functions and languages. This model is often refered to as **Ex**-learning.[1] We refer the reader to [AS83, BB75, CS83, JORS99, KW80] for background material in this field.

For function learning, there is a learner-synthesizer algorithm **lsyn** so that, if **lsyn** is fed any procedure that lists programs for some (possibly infinite) class $\mathcal{S}$ of (total) functions, then **lsyn** outputs an **Ex**-learner successful on $\mathcal{S}$ [Gol67]. The learners so synthesized are called *enumeration techniques* [BB75, Ful90]. These enumeration techniques yield many positive learnability results, for example, that the class of all functions computable in time polynomial in the length of input is **Ex**-learnable.[2]

For this paper, as is the practice in inductive inference literature, we consider a Turing Machine *index* for accepting/generating a language and *grammar* for the language as synonyms. For language learning *from positive data and with learners outputting grammars*, [OSW88] provided an amazingly negative result: there is *no* learner-synthesizer algorithm **lsyn** so that, if **lsyn** is fed any *pair* of grammars g_1, g_2 for any language class $\mathcal{L} = \{L_1, L_2\}$, then **lsyn** outputs an **Ex**-learner successful, from positive data, on $\mathcal{L}$.[3] [BCJ99] showed how to circumvent *some* of the sting of this [OSW88] result by resorting to more general learners than **Ex**. In particular, they used **Bc**-*learners*, which, when successful on an object input, (by definition) find a final (possibly infinite) *sequence* of correct programs for that object after at most finitely many trial and error attempts [$\mathrm{B\bar{7}4}$, CS83].[4] Of course, if suitable learner-synthesizer algorithm **lsyn** is fed procedures for listing *decision procedures* (instead of mere grammars), one also has more success at synthesizing learners. In fact the inductive inference community has shown considerable interest (spanning at least from [Gol67] to [ZL95]) in language classes defined by r.e. listings of decision procedures. These classes are called *uniformly decidable* or *indexed families of recursive languages* (or just indexed families). As is essentially pointed out in [Ang80], all of the formal language style example classes *are* indexed families. Note that Gold's result on learning enumerable class of functions does not apply to learning of indexed family of recursive languages, since for learning indexed families of recursive languages, the learner gets only positive data as input. A sample result from [BCJ99] is: there *is* a learner-synthesizer algorithm **lsyn** so that, if **lsyn** is fed any procedure that lists decision procedures defining some indexed family $\mathcal{L}$ of recursive languages *which can be* **Bc**-*learned from positive data with the learner outputting grammars*, then **lsyn** outputs a **Bc**-learner successful, from positive data, on $\mathcal{L}$. The proof of this positive result yielded the surprising characterization [BCJ99]: for indexed families $\mathcal{L}$ of recursive languages, $\mathcal{L}$ can

[1] **Ex** is short for *explanatory.*

[2] The reader is referred to Jantke [Jan79a, Jan79b] for a discussion of synthesizing learners for classes of computable functions that are not necessarily recursively enumerable.

[3] Also for language learning from positive data and with learners outputting grammars, a somewhat related negative result is provided by Kapur [Kap91]. He shows that one cannot algorithmically find an **Ex**-learning machine for **Ex**-*learnable* indexed families of recursive languages from an index of the class. This is a bit weaker than a closely related negative result from [BCJ99].

[4] **Bc** is short for *behaviorally correct.*

be **Bc**-learned from positive data with the learner outputting grammars iff

$$(\forall L \in \mathcal{L})(\exists S \subseteq L \mid S \text{ is finite})(\forall L' \in \mathcal{L} \mid S \subseteq L')[L' \not\subset L]. \tag{1}$$

(1) is Angluin's important Condition 2 from [Ang80], and it is referred to as the *subset principle*, in general a necessary condition for preventing overgeneralization in learning from positive data [Ang80, Ber85, ZLK95, KB92, Cas99].

In this paper we consider the effect of inaccuracies on the ability to synthesize learning machines. In the real world one always finds inaccuracies in the input data, and learning is often achieved despite the presence of inaccuracies in the data. For example, in the context of linguistic development, children likely receive ungrammatical sentences and may not receive some sentences. However, these inaccuracies do not seem to influence the outcome of linguistic development. Similarly, in the context of scientific discovery, the business of science progresses despite experimental errors and unfeasibility of performing certain experiments. We refer the reader to [FJ96, Jai96, SR85, Ste95] for some background on learning from inaccurate information.

[CJS99] considered language learning from both *noisy* texts (only positive data) and from *noisy* informants (both positive and negative data), and adopted, as does the present paper, Stephan's [Ste95, CJS00] noise model. Roughly, in this model correct information about an object occurs infinitely often while incorrect information occurs only finitely often. Hence, this model has the advantage that noisy data about an object nonetheless uniquely specifies that object.[5]

In the context of [CJS99], *where the noisy data sequences can be uncomputable*, the presence of noise plays havoc with the learnability of many concrete classes that can be learned without noise. For example, the well-known class of *pattern languages* [Ang80][6] can be **Ex**-learned from texts but cannot be **Bc**-learned from unrestricted noisy texts even if we allow the final grammars each to make finitely many mistakes. While it is possible to **Ex**-learn the pattern languages *from informants* in the presence of noise, a mind-change complexity price must be paid: any **Ex**-learner succeeding on the pattern languages from an unrestricted noisy informant must change its mind an unbounded finite number of times about the final grammar. However, some learner can succeed on the pattern languages from noise-*free* informants *and on its first guess as to a correct grammar* (see [LZK96]). The class of languages formed by taking the union of two pattern languages can be **Ex**-learned from texts [Shi83]; however, this class cannot be **Bc**-learned from unrestricted noisy informants even if we allow the final grammars each to make finitely many mistakes.

In [CJS99], the proofs of most of the positive results providing *existence* of learner-synthesizers which synthesize *noise-tolerant* learners also yielded pleasant characterizations which look like strict versions of the subset principle (1).[7] Here is an example. If $\mathcal{L}$ is an indexed family of recursive

[5] Less roughly: in the case of noisy informant each false item may occur a finite number of times; in the case of text, it is mathematically more interesting to require, as we do, that the *total* amount of false information has to be finite. The alternative of allowing *each* false item in a text to occur finitely often is too restrictive; it would, then, be impossible to learn even the class of all singleton sets [Ste95] (see also Theorem 14).

[6] [Nix83] as well as [SA95] outline interesting applications of pattern inference algorithms. For example, pattern language learning algorithms have been successfully applied for solving problems in molecular biology (see [SSS+94, SA95]). Pattern languages and finite unions of pattern languages [Shi83, Wri89, KMU95, CJLZ99] turn out to be subclasses of Smullyan's [Smu61] Elementary Formal Systems (EFSs). [ASY92] show that the EFSs can also be treated as a logic programming language over strings. The techniques for learning finite unions of pattern languages have been extended to show the learnability of various subclasses of EFSs [Shi91]. Investigations of the learnability of subclasses of EFSs are important because they yield corresponding results about the learnability of subclasses of logic programs. [AS94] use the insight gained from the learnability of EFSs subclasses to show that a class of linearly covering logic programs with local variables is **TxtEx**-learnable. These results have consequences which should be of interest for Inductive Logic Programming [MR94, LD94].

[7] For $\mathcal{L}$ either an indexed family or defined by some r.e. listing of grammars, the prior literature has many interesting

languages, then: $\mathcal{L}$ can be noise-tolerantly **Ex**-learned from positive data with the learner outputting grammars (iff $\mathcal{L}$ can be noise-tolerantly **Bc**-learned from positive data with the learner outputting grammars) iff

$$(\forall L, L' \in \mathcal{L})[L \subseteq L' \Rightarrow L = L']. \tag{2}$$

(2) is easily checkable (as is (1) above, but, (2) is more restrictive, as we saw in the just previous paragraph).

In a computable universe, all data sequences, even noisy ones, are computable.[8] In the present paper, we are concerned with learner-synthesizer algorithms which operate on procedures that list either grammars or decision procedures but, significantly, *we restrict the noisy data sequences to being computable.*

Herein, our main and surprising result (Theorem 13 in Section 4.1 below) is: there *is* a learner-synthesizer algorithm **lsyn** so that, if **lsyn** is fed any procedure that lists decision procedures defining *any* indexed family $\mathcal{L}$ of recursive languages, then **lsyn** outputs a learner which, from *computable*, noisy, positive data on any $L \in \mathcal{L}$, outputs a sequence of grammars eventually all correct for L. This result has the following corollary (Corollary 1 in Section 4.1 below): for *every* indexed family $\mathcal{L}$ of recursive languages, there *is* a machine for **Bc**-learning $\mathcal{L}$, where the machine outputs grammars and the input is *computable* noisy positive data. Essentially Theorem 13 is a constructive version of this corollary: not only can each indexed family be **Bc**-learned (outputting grammars on computable noisy positive data), but one can *algorithmically find* a corresponding **Bc**-learner (of this kind) from an *index* for any indexed family. As a corollary to Theorem 13 we have that the class of finite unions of pattern languages *is* **Bc**-learnable from *computable* noisy texts, where the machine outputs grammars (this contrasts sharply with the negative result mentioned above from [CJS99] that even the class of pattern languages is not learnable from *unrestricted* noisy texts).

Another main positive result of the present paper is Corollary 3 in Section 4.1 below. It says that an indexed family $\mathcal{L}$ can be **Bc**-learned from *computable* noisy *informant* data by outputting grammars iff

$$(\forall L \in \mathcal{L})(\exists z)(\forall L' \in \mathcal{L} \mid \{x \leq z \mid x \in L\} = \{x \leq z \mid x \in L'\})[L' \subseteq L]. \tag{3}$$

Corollary 2 in the same section is the constructive version of Corollary 3 and says one can algorithmically find such a learner from an index for any indexed family so learnable. (3) is easy to check too and intriguingly differs slightly from the characterization in [CJS99] of the same learning criterion applied to indexed families *but with the noisy data sequences unrestricted*:

$$(\forall L \in \mathcal{L})(\exists z)(\forall L' \in \mathcal{L} \mid \{x \leq z \mid x \in L\} = \{x \leq z \mid x \in L'\})[L' = L]. \tag{4}$$

Let N denote the set of natural numbers. Then $\{L \mid \mathrm{card}(N - L) \text{ is finite }\}$ satisfies (3), but not (4).[9]

As might be expected, for several learning criteria considered here and in previous papers on synthesis, the restriction to *computable* noisy data sequences may, in some cases, reduce a criterion to one previously studied, but, in other cases (e.g., the one mentioned at the end of the just previous

---

characterizations of $\mathcal{L}$ being **Ex**-learnable from noise-free positive data, with and without extra restrictions. See, for example, [Ang80, Muk92, LZK96, dJK96].

[8] In a computable universe (which ours might be), only computable data sequences are available to be presented to learning machines. That the universe may be discrete and computable is taken seriously, for example, in [Zus69, Tof77, TM87, Fey82, Cas92, Cas86, CRS94, Cas99]. Note that in a discrete, *random* universe with only computable probability distributions for its behavior (e.g., a discrete, quantum mechanical universe), the *expected* behavior will still be computable [dMSS56] (and constructively so [Gil72, Gil77]).

[9] However, $\mathcal{L}$ = the class of all unions of two pattern languages satisfies *neither* (3) nor (4).

paragraph), not. Section 3 below, then, contains many of the comparisons of the criteria of this paper to those of previous papers.

As we indicated above, Section 4.1 below contains the main results of the present paper, and, in general, the results of this section are about synthesis from indices for indexed families and, when appropriate, corresponding characterizations. Section 4.2 below contains our positive and negative results on synthesis from r.e. indices for r.e. classes.

As we noted above, in a computable universe, all data sequences, even noisy ones, are computable. One of the motivations for considering possibly non-computable data sequences is that, in the case of child language learning, the utterances the child hears (as its data) may, in part, be determined by uncomputable processes [OSW86] perhaps external to the utterance generators (e.g., the parents). The *limit recursive* functions are in between the computable and the arbitrarily uncomputable. Here is the idea. Informally, they are (by definition) the functions computed by *limit-programs*, programs which do not give correct output until after some unspecified but finite number of trial outputs [Sha71]. They "change their minds" finitely many times about each output before getting it right.[10] In Section 5 we consider briefly what would happen if the world provided *limit recursive* data sequences (instead of computable or unrestricted ones). The main result of this section, Corollary 7, is that, for **Bc**-learning of grammars from positive data, learning from limit recursive data sequences is (constructively) the same as learning from unrestricted data sequences. Importantly, the same proof yields this equivalence also in the case of noisy data sequences.

Finally Section 6 gives some directions for further research.

2 Preliminaries

2.1 Notation and Identification Criteria

The recursion theoretic notions are from the books of Odifreddi [Odi89] and Soare [Soa87]. $N = \{0, 1, 2, \ldots\}$ is the set of all natural numbers, and this paper considers r.e. subsets L of N. $N^+ = \{1, 2, 3, \ldots\}$, the set of all positive integers. All conventions regarding range of variables apply, with or without decorations[11], unless otherwise specified. We let c, e, i, j, k, l, m, n, q, s, t, u, v, w, x, y, z, range over N. Empty set, member of, subset, superset, proper subset, and proper superset are respectively denoted by $\emptyset, \in, \subseteq, \supseteq, \subset, \supset$. Maximum, minimum, and cardinality of a set are respectively denoted by $\max(), \min(), \mathrm{card}()$, where by convention $\max(\emptyset) = 0$ and $\min(\emptyset) = \infty$. We use $\mathrm{card}(S) \leq *$ to mean that cardinality of set S is finite. We let a, b range over $N \cup \{*\}$. We let $\langle \cdot, \cdot \rangle$ stand for an arbitrary but fixed, one to one, computable encoding of all pairs of natural numbers onto N. Similarly, $\langle \cdot, \cdot, \cdot \rangle$ denotes a computable, 1–1 encoding of all triples of natural numbers onto N. The complement of a set L is denoted by $\overline{L}$. Characteristic function of a set L is denoted by χ_L. $L_1 \boldsymbol{\Delta} L_2$ denotes the symmetric difference of L_1 and L_2, i.e., $L_1 \boldsymbol{\Delta} L_2 = (L_1 - L_2) \cup (L_2 - L_1)$. $L_1 =^a L_2$ means that $\mathrm{card}(L_1 \boldsymbol{\Delta} L_2) \leq a$. Quantifiers $\forall^\infty, \exists^\infty$, and $\exists!$ denote for all but finitely many, there exist infinitely many, and there exists a unique respectively.

The set of total computable functions from N to N is denoted by $\mathcal{R}$. We let f, g, range over total computable functions. The set of all recursively enumerable sets is denoted by $\mathcal{E}$. We let L range over $\mathcal{E}$. We let $\mathcal{L}$ range over subsets of $\mathcal{E}$. REC denotes the set of all recursive languages. The power set of REC is denoted by 2^{REC}. We fix a standard acceptable programming system (acceptable numbering) φ. The function computed by the i-th program in the programming system φ is denoted by φ_i. We

[10] Incidentally, all the results in this paper about the *non*-existence of computable synthesizers are can also be shown to be non-existence results for *limit recursive* synthesizers.

[11] Decorations are subscripts, superscripts, primes and the like.

also call i a program or index for φ_i. For a (partial) function η, domain(η) and range(η) respectively denote the domain and range of partial function η. We often write $\eta(x)\downarrow$ ($\eta(x)\uparrow$) to denote that $\eta(x)$ is defined (undefined). W_i denotes the domain of φ_i. Thus, W_i is considered as the language accepted (or enumerated) by the i-th program in φ system, and we say that i is a grammar or index for W_i. We let Φ denote a standard Blum complexity measure [Blu67] for the programming system φ. $W_{i,s} = \{x < s \mid \Phi_i(x) < s\}$. A program j such that $\varphi_j = \chi_L$, is called a decision procedure for L.

A *text* is a mapping from N to $N \cup \{\#\}$. We let T range over texts. content(T) is defined to be the set of natural numbers in the range of T (i.e. content(T) = range$(T) - \{\#\}$). T is a *text for* L iff content$(T) = L$. That means a text for L is an infinite sequence whose range, except for a possible #, is just L.

An *information sequence or informant* is a mapping from N to $(N \times \{0,1\}) \cup \{\#\}$. We let I range over informants. content(I) is defined to be the set of pairs in the range of I (i.e. content(I) = range$(I) - \{\#\}$). An *informant for* L is an informant I such that content$(I) = \{(x,b) \mid \chi_L(x) = b\}$. It is useful to consider the canonical information sequence for L. I is a canonical information sequence for L iff $I(x) = (x, \chi_L(x))$. We sometimes abuse notation and refer to the canonical information sequence for L by χ_L.

We let σ and τ range over finite initial segments of texts or information sequences, where the context determines which is meant. We denote the set of finite initial segments of texts by SEG and set of finite initial segments of information sequences by SEQ. We use $\sigma \preceq T$ (respectively, $\sigma \preceq I$, $\sigma \preceq \tau$) to denote that σ is an initial segment of T (respectively, I, τ). Length of σ is denoted by $|\sigma|$. We use $T[n]$ to denote the initial segment of T of length n. Similarly, $I[n]$ denotes the initial segment of I of length n. Let $T[m:n]$ denote the segment $T(m), T(m+1), \ldots, T(n-1)$ (i.e. $T[n]$ with the first m elements, $T[m]$, removed). $I[m:n]$ is defined similarly. We use $\sigma \diamond \tau$ (respectively, $\sigma \diamond T$, $\sigma \diamond I$) to denote the concatenation of σ and τ (respectively, concatenation of σ and T, concatenation of σ and I). We sometimes abuse notation and say $\sigma \diamond w$ to denote the concatenation of σ with the sequence of one element w.

A *learning machine* $\mathbf{M}$ is a mapping from initial segments of texts (information sequences) to N. We say that $\mathbf{M}$ converges on T to i, (written: $\mathbf{M}(T)\downarrow = i$) iff, for all but finitely many n, $\mathbf{M}(T[n]) = i$. If there is no i such that $\mathbf{M}(T)\downarrow = i$, then we say that M diverges on T (written: $\mathbf{M}(T)\uparrow$). Convergence on information sequences is defined similarly.

Let ProgSet$(\mathbf{M}, \sigma) = \{\mathbf{M}(\tau) \mid \tau \subseteq \sigma\}$.

Definition 1 Suppose $a, b \in N \cup \{*\}$.
(a) Below, for each of several learning criteria $\mathbf{J}$, we define what it means for a machine $\mathbf{M}$ to $\mathbf{J}$-*identify* a language L from a text T or informant I.

- [Gol67, CL82] $\mathbf{M}$ $\mathbf{TxtEx}^a$-*identifies* L *from text* T iff $(\exists i \mid W_i =^a L)[\mathbf{M}(T)\downarrow = i]$.
- [Gol67, CL82] $\mathbf{M}$ $\mathbf{InfEx}^a$-*identifies* L *from informant* I iff $(\exists i \mid W_i =^a L)[\mathbf{M}(I)\downarrow = i]$.
- [B$\bar{7}$4, CL82]. $\mathbf{M}$ $\mathbf{TxtBc}^a$-*identifies* L *from text* T iff $(\forall^\infty n)[W_{\mathbf{M}(T[n])} =^a L]$.
- [B$\bar{7}$4, CL82]. $\mathbf{M}$ $\mathbf{InfBc}^a$-*identifies* L *from informant* I iff $(\forall^\infty n)[W_{\mathbf{M}(I[n])} =^a L]$.

(b) Suppose $\mathbf{J} \in \{\mathbf{TxtEx}^a, \mathbf{TxtBc}^a\}$. $\mathbf{M}$ $\mathbf{J}$-*identifies* L iff, for all texts T for L, $\mathbf{M}$ $\mathbf{J}$-identifies L from T. In this case we also write $L \in \mathbf{J}(\mathbf{M})$.

We say that $\mathbf{M}$ $\mathbf{J}$-identifies $\mathcal{L}$ iff $\mathbf{M}$ $\mathbf{J}$-identifies each $L \in \mathcal{L}$.

$\mathbf{J} = \{\mathcal{L} \mid (\exists \mathbf{M})[\mathcal{L} \subseteq \mathbf{J}(\mathbf{M})]\}$.

(c) Suppose $\mathbf{J} \in \{\mathbf{InfEx}^a, \mathbf{InfBc}^a\}$. $\mathbf{M}$ $\mathbf{J}$-*identifies* L iff, for all information sequences I for L, $\mathbf{M}$ $\mathbf{J}$-identifies L from I. In this case we also write $L \in \mathbf{J}(\mathbf{M})$.

We say that $\mathbf{M}$ $\mathbf{J}$-identifies $\mathcal{L}$ iff $\mathbf{M}$ $\mathbf{J}$-identifies each $L \in \mathcal{L}$.

$\mathbf{J} = \{\mathcal{L} \mid (\exists \mathbf{M})[\mathcal{L} \subseteq \mathbf{J}(\mathbf{M})]\}$.

We often write $\mathbf{TxtEx}^0$ as $\mathbf{TxtEx}$. A similar convention applies to the other learning criteria of this paper.

Next we prepare to introduce our noisy inference criteria, and, in that interest, we define some ways to calculate the number of occurrences of words in (initial segments of) a text or informant. For $\sigma \in \mathrm{SEG}$, and text T, let

$$\mathrm{occur}(\sigma, w) \stackrel{\mathrm{def}}{=} \mathrm{card}(\{j \mid j < |\sigma| \ \wedge \ \sigma(j) = w\}) \text{ and}$$
$$\mathrm{occur}(T, w) \stackrel{\mathrm{def}}{=} \mathrm{card}(\{j \mid j \in N \ \wedge \ T(j) = w\}).$$

For $\sigma \in \mathrm{SEQ}$ and information sequence I, $\mathrm{occur}(\cdot,\cdot)$ is defined similarly except that w is replaced by (v, b).

For any language L, $\mathrm{occur}(T, L) \stackrel{\mathrm{def}}{=} \Sigma_{x \in L}\, \mathrm{occur}(T, x)$.

Definition 2 [Ste95] An information sequence I is a *noisy information sequence* (or noisy informant) for L iff $(\forall x)\,[\mathrm{occur}(I, (x, \chi_L(x))) = \infty \ \wedge \ \mathrm{occur}(I, (x, \chi_{\overline{L}}(x))) < \infty]$. A text T is a *noisy text* for L iff $(\forall x \in L)[\mathrm{occur}(T, x) = \infty]$ and $\mathrm{occur}(T, \overline{L}) < \infty$.

On the one hand, both concepts are similar since $L = \{x \mid \mathrm{occur}(I, (x, 1)) = \infty\} = \{x \mid \mathrm{occur}(T, x) = \infty\}$. On the other hand, the concepts differ in the way they treat errors. In the case of informant every false item $(x, \chi_{\overline{L}}(x))$ may occur a finite number of times. In the case of text, it is mathematically more interesting to require, as we do, that the total amount of false information has to be finite.[12]

Definition 3 [Ste95, CJS00] *Suppose* $a \in N \cup \{*\}$. *Suppose* $\mathbf{J} \in \{\mathbf{TxtEx}^a, \mathbf{TxtBc}^a\}$. *Then* $\mathbf{M}$ $\mathbf{NoisyJ}$-identifies L *iff, for all noisy texts* T *for* L, $\mathbf{M}$ $\mathbf{J}$*-identifies* L *from* T. *In this case we write* $L \in \mathbf{NoisyJ}(\mathbf{M})$.

$\mathbf{M}$ $\mathbf{NoisyJ}$-identifies *a class* $\mathcal{L}$ *iff* $\mathbf{M}$ $\mathbf{NoisyJ}$*-identifies each* $L \in \mathcal{L}$.

$\mathbf{NoisyJ} = \{\mathcal{L} \mid (\exists \mathbf{M})[\mathcal{L} \subseteq \mathbf{NoisyJ}(\mathbf{M})]\}$.

Inference criteria for learning from noisy informants are defined similarly.

It is useful to introduce the set of positive and negative occurrences in (initial segment of) an informant. Suppose $\sigma \in \mathrm{SEQ}$

$$\begin{aligned}\mathrm{PosInfo}(\sigma) &\stackrel{\mathrm{def}}{=} \{v \mid \mathrm{occur}(\sigma, (v, 1)) \geq \mathrm{occur}(\sigma, (v, 0)) \wedge \mathrm{occur}(\sigma, (v, 1)) \geq 1\} \\ \mathrm{NegInfo}(\sigma) &\stackrel{\mathrm{def}}{=} \{v \mid \mathrm{occur}(\sigma, (v, 1)) < \mathrm{occur}(\sigma, (v, 0)) \wedge \mathrm{occur}(\sigma, (v, 0)) \geq 1\}\end{aligned}$$

That means, that $\mathrm{PosInfo}(\sigma) \cup \mathrm{NegInfo}(\sigma)$ is just the set of all v such that either $(v, 0)$ or $(v, 1)$ occurs in σ. Then $v \in \mathrm{PosInfo}(\sigma)$ if $(v, 1)$ occurs at least as often as $(v, 0)$ and $v \in \mathrm{NegInfo}(\sigma)$ otherwise. Similarly,

$$\begin{aligned}\mathrm{PosInfo}(I) &= \{v \mid \mathrm{occur}(I, (v, 1)) \geq \mathrm{occur}(I, (v, 0)) \wedge \mathrm{occur}(I, (v, 1)) \geq 1\} \\ \mathrm{NegInfo}(I) &= \{v \mid \mathrm{occur}(I, (v, 1)) < \mathrm{occur}(I, (v, 0)) \wedge \mathrm{occur}(I, (v, 0)) \geq 1\}\end{aligned}$$

where, if $\mathrm{occur}(I, (v, 0)) = \mathrm{occur}(I, (v, 1)) = \infty$, then we place v in $\mathrm{PosInfo}(I)$ (this is just to make the definition precise; we will not need this for criteria of inference discussed in this paper).

Several proofs in this paper depend on the concept of locking sequence.

[12] As we noted in Section 1 above, the alternative of allowing each false item in a text to occur finitely often is too restrictive; it would, then, be impossible to learn even the class of all singleton sets [Ste95].

Definition 4 (Based on [BB75]) Suppose $a \in N \cup \{*\}$.

(a) σ is said to be a $\mathbf{TxtEx}^a$-*locking sequence for* $\mathbf{M}$ *on* L iff, $\mathrm{content}(\sigma) \subseteq L$, $W_{\mathbf{M}(\sigma)} =^a L$, and $(\forall \tau \mid \mathrm{content}(\tau) \subseteq L)[\mathbf{M}(\sigma \diamond \tau) = \mathbf{M}(\sigma)]$.

(b) σ is said to be a $\mathbf{TxtBc}^a$-*locking sequence for* $\mathbf{M}$ *on* L iff, $\mathrm{content}(\sigma) \subseteq L$, and $(\forall \tau \mid \mathrm{content}(\tau) \subseteq L)[W_{\mathbf{M}(\sigma \diamond \tau)} =^a L]$.

Lemma 1 (Based on [BB75]) *Suppose* $a, b \in N \cup \{*\}$. *Suppose* $\mathbf{J} \in \{\mathbf{TxtEx}^a, \mathbf{TxtBc}^a\}$. *If* $\mathbf{M}$ $\mathbf{J}$*-identifies* L *then there exists a* $\mathbf{J}$*-locking sequence for* $\mathbf{M}$ *on* L.

Definition of locking sequences for learning from noisy texts is similar to that of learning from noise free texts (we just drop the requirement that $\mathrm{content}(\sigma) \subseteq L$). However, the definition of locking sequence for learning from a noisy informant is more involved.

Definition 5 [CJS00] Suppose $a, b \in N \cup \{*\}$.

(a) σ is said to be a $\mathbf{NoisyTxtEx}^a$-*locking sequence for* $\mathbf{M}$ *on* L iff, $W_{\mathbf{M}(\sigma)} =^a L$, and $(\forall \tau \mid \mathrm{content}(\tau) \subseteq L)[\mathbf{M}(\sigma \diamond \tau) = \mathbf{M}(\sigma)]$.

(b) σ is said to be a $\mathbf{NoisyTxtBc}^a$-*locking sequence for* $\mathbf{M}$ *on* L iff $(\forall \tau \mid \mathrm{content}(\tau) \subseteq L)[W_{\mathbf{M}(\sigma \diamond \tau)} =^a L]$.

For defining locking sequences for learning from noisy informant, we need the following.

Definition 6 Let $S \subseteq N$ and $L \subseteq N$. $\mathrm{Inf}[S, L] \stackrel{\mathrm{def}}{=} \{\tau \mid (\forall x \in S)\,[\mathrm{occur}(\tau, (x, \chi_{\overline{L}}(x))) = 0]\}$.

Definition 7 Suppose $a, b \in N \cup \{*\}$.

(a) σ is said to be a $\mathbf{NoisyInfEx}^a$-*locking sequence for* $\mathbf{M}$ *on* L iff, $\mathrm{PosInfo}(\sigma) \subseteq L$, $\mathrm{NegInfo}(\sigma) \subseteq \overline{L}$, $W_{\mathbf{M}(\sigma)} =^a L$, and $(\forall \tau \in \mathrm{Inf}[\mathrm{PosInfo}(\sigma) \cup \mathrm{NegInfo}(\sigma), L])[\mathbf{M}(\sigma \diamond \tau) = \mathbf{M}(\sigma)]$.

(b) σ is said to be a $\mathbf{NoisyInfBc}^a$-*locking sequence for* $\mathbf{M}$ *on* L iff, $\mathrm{PosInfo}(\sigma) \subseteq L$, $\mathrm{NegInfo}(\sigma) \subseteq \overline{L}$, and $(\forall \tau \in \mathrm{Inf}[\mathrm{PosInfo}(\sigma) \cup \mathrm{NegInfo}(\sigma), L])[W_{\mathbf{M}(\sigma \diamond \tau)} =^a L]$.

For the criteria of noisy inference discussed in this paper, as mentioned in [CJS00], one can prove the existence of a locking sequence using the technique of [Ste95, Theorem 2, proof for $\mathbf{NoisyEx} \subseteq \mathbf{Ex}_0[K]$].

Proposition 1 *Suppose* $a, b \in N \cup \{*\}$. *If* $\mathbf{M}$ *learns* L *from noisy text or informant according to one of the criteria* $\mathbf{NoisyTxtEx}^a$, $\mathbf{NoisyTxtBc}^a$, $\mathbf{NoisyInfEx}^a$, *or* $\mathbf{NoisyInfBc}^a$, *then there exists a corresponding locking sequence for* $\mathbf{M}$ *on* L.

Note that in all the learning criteria formally defined thus far in this section, the (possibly noisy) texts or informants may be of arbitrary complexity. In a computable universe all texts and informants (even noisy ones) must be recursive (synonym: computable). As noted in Section 1 above, this motivates our concentrating in this paper on recursive texts and informants.

When a learning criterion is restricted to requiring learning from *recursive texts/informants only*, then we name the resultant criteria by adding, in an appropriate spot, '**Rec**' to the name of the unrestricted criterion. For example, **RecTxtEx**-identification is this restricted variant of **TxtEx**-identification. Formally, **RecTxtEx**-identification may be defined as follows.

Definition 8 $\mathbf{M}$ $\mathbf{RecTxtEx}^a$-*identifies* L iff, for all *recursive* texts T for L, $\mathbf{M}$ $\mathbf{TxtEx}^a$-identifies L from T.

One can similarly define $\mathbf{RecInfEx}^a$, $\mathbf{RecTxtBc}^a$, $\mathbf{RecInfBc}^a$, $\mathbf{NoisyRecTxtEx}^a$, $\mathbf{NoisyRecTxtBc}^a$, $\mathbf{NoisyRecInfEx}^a$, $\mathbf{NoisyRecInfBc}^a$.

$\mathbf{RecTxtBc}^a \neq \mathbf{TxtBc}^a$ [CL82, Fre85]; however, $\mathbf{TxtEx}^a = \mathbf{RecTxtEx}^a$ [BB75, Wie77, Cas99]. In Section 3 below, we indicate the remaining comparisons.

2.2 Recursively Enumerable Classes and Indexed Families

This paper is about the synthesis of algorithmic learners for r.e. classes of r.e. languages and of indexed families of recursive languages. To this end we define, for all i, $\mathcal{C}_i \stackrel{\text{def}}{=} \{W_j \mid j \in W_i\}$. Hence, $\mathcal{C}_i$ is *the r.e. class with index* i. For a decision procedure j, we let $U_j \stackrel{\text{def}}{=} \{x \mid \varphi_j(x) = 1\}$. For a decision procedure j, we let $U_j[n]$ denote $\{x \in U_j \mid x < n\}$. For all i,

$$\mathcal{U}_i \stackrel{\text{def}}{=} \begin{cases} \{U_j \mid j \in W_i\}, & \text{if } (\forall j \in W_i)[j \text{ is a decision procedure}]; \\ \emptyset, & \text{otherwise.} \end{cases}$$

Hence, $\mathcal{U}_i$ is *the indexed family with index* i.

2.3 Some Previous Results on Noise Tolerant Learning

In this section, we state some results from [CJS00] and some consequences of these results (or related results) which we will apply later in the present paper. We let $2* \stackrel{\text{def}}{=} *$.

Using Proposition 1 we have the following two theorems,

Theorem 1 *Suppose* $a \in N \cup \{*\}$. *Suppose* $\mathcal{L} \in \mathbf{NoisyInfBc}^a$. *Then for all* $L \in \mathcal{L}$, *there exists an* n *such that,* $(\forall L' \in \mathcal{L} \mid \{x \in L \mid x \leq n\} = \{x \in L' \mid x \leq n\})[L =^{2a} L']$.

Theorem 2 *Suppose* $a \in N \cup \{*\}$. *Suppose* $\mathcal{L} \in \mathbf{NoisyInfEx}^a$. *Then, for all* $L \in \mathcal{L}$, *there exist* n, S *such that,* $(\forall L' \in \mathcal{L} \mid \{x \in L \mid x \leq n\} = \{x \in L' \mid x \leq n\})[(L \mathbf{\Delta} S) =^a L']$.

As a corollary to Theorem 2 we have

Theorem 3 *Suppose* $a \in N \cup \{*\}$. *Suppose* $\mathcal{L} \in \mathbf{NoisyInfEx}^a$. *Then, for all* $L \in \mathcal{L}$, *there exists an* n *such that,* $(\forall L' \in \mathcal{L} \mid \{x \in L \mid x \leq n\} = \{x \in L' \mid x \leq n\})[L =^a L']$.

The following two theorems were proved in [CJS00].

Theorem 4 [CJS00] *Suppose* $a \in N \cup \{*\}$. $\mathcal{L} \in \mathbf{NoisyTxtBc}^a \Rightarrow [(\forall L \in \mathcal{L})(\forall L' \in \mathcal{L} \mid L' \subseteq L)[L =^{2a} L']]$.

Theorem 5 [CJS00] *Suppose* $a \in N \cup \{*\}$. *Then* $\mathbf{NoisyInfBc}^a \cup \mathbf{NoisyTxtBc}^a \subseteq \mathbf{TxtBc}^a$ *and* $\mathbf{NoisyInfEx}^a \cup \mathbf{NoisyTxtEx}^a \subseteq \mathbf{TxtEx}^a$.

The proof of Theorem 5 also shows:

Theorem 6 *Suppose* $a \in N \cup \{*\}$. *Then* $\mathbf{NoisyRecInfBc}^a \cup \mathbf{NoisyRecTxtBc}^a \subseteq \mathbf{RecTxtBc}^a$ *and* $\mathbf{NoisyRecInfEx}^a \cup \mathbf{NoisyRecTxtEx}^a \subseteq \mathbf{RecTxtEx}^a$.

The following proposition is easy to prove:

Proposition 2 *Suppose* $\mathcal{L} \subseteq \mathcal{E}$ *is a finite class of languages such that for all* $L, L' \in \mathcal{L}$, $L \subseteq L' \Rightarrow L = L'$. *Then,* $\mathcal{L} \in \mathbf{NoisyTxtEx} \cap \mathbf{NoisyInfEx}$.

Suppose $\mathcal{L} \subseteq \mathcal{E}$ *is a finite class of languages. Then,* $\mathcal{L} \in \mathbf{NoisyInfEx}$.

3 Comparisons

In this section we consider the comparisons between the inference criteria introduced in this paper among themselves and with the related inference criteria from the literature.

The next theorem says that for $\mathbf{Bc}^*$-learning, with computable noise, from either texts or informants, some machine learns grammars for all the r.e. languages. It improves a similar result from [CL82] for the noise-free case.

Theorem 7 *(a)* $\mathcal{E} \in \mathbf{NoisyRecTxtBc}^*$.
(b) $\mathcal{E} \in \mathbf{NoisyRecInfBc}^*$.

PROOF. (a) Define $\mathbf{M}$ as follows: $\mathbf{M}(T[n]) = prog(T[n])$, where $W_{prog(T[n])}$ is defined by the following enumeration.

$W_{prog(T[n])}$
Go to stage 0
Stage s
Let $m = \min(\{n\} \cup \{i \mid i \leq n \ \wedge\ (\forall x < n)[\Phi_i(x) \leq s \ \wedge\ \varphi_i(x) = T(x)]\})$.
Enumerate $\{\varphi_m(x) \mid x \leq s \ \wedge\ \Phi_m(x) \leq s\}$.
Go to stage $s+1$.
End Stage s
End

Now suppose T is a noisy recursive text for $L \in \mathcal{E}$. Let m' be the minimum program such that $\varphi_{m'} = T$. Let $n_0 > m'$ be large enough so that, for all $i < m'$, there exists an $x < n_0$ such that $\varphi_i(x) \neq T(x)$. Now, for all $n > n_0$, for all but finitely many s, m as computed in the procedure for $W_{prog(T[n])}$ in stage s is m'. It follows that $W_{prog(T[n])}$ is a finite variant of content(T), and thus a finite variant of L. Thus $\mathbf{M}$ $\mathbf{NoisyRecTxtBc}^*$-identifies $\mathcal{E}$.

(b) Define $\mathbf{M}$ as follows: $\mathbf{M}(I[n]) = prog(I[n])$, where $W_{prog(I[n])}$ is defined by the following enumeration.

$W_{prog(I[n])}$
Go to stage 0
Stage s
Let $m = \min(\{n\} \cup \{i \mid i \leq n \ \wedge\ (\forall x < n)[\Phi_i(x) \leq s \ \wedge\ \varphi_i(x) = I(x)]\})$.
Let $p = \min(\{n\} \cup \{i \mid i \leq n \ \wedge\ (\forall x < n)[x \in W_{i,s} \Leftrightarrow \mathrm{card}(\{w \mid w < s \ \wedge\ \Phi_m(w) < s \ \wedge\ \varphi_m(w) = (x,1)\}) \geq \mathrm{card}(\{w \mid w < s \ \wedge\ \Phi_m(w) < s \ \wedge\ \varphi_m(w) = (x,0)\})]\})$.
Enumerate $W_{p,s}$.
Go to stage $s+1$.
End Stage s
End

Now suppose I is a noisy informant for L. Let m' be the minimum program such that $\varphi_{m'} = I$. Let p' be the minimum grammar for L. Let $n_0 > \max(\{m', p'\})$ be large enough so that, for all $i < m'$, there exists an $x < n_0$ such that $\varphi_i(x) \neq I(x)$ and, for all $j < p'$, there exists an $x < n_0$, such that $x \in L\boldsymbol{\Delta}W_j$. Thus, for all $n > n_0$, for all but finitely many s, in stage s of the procedure for

$W_{prog(T[n])}$, we have $m = m'$ and $p = p'$. It follows that $W_{prog(I[n])}$ is a finite variant of $W_{p'} = L$. Thus $\mathbf{M}$ $\mathbf{NoisyRecInfBc}^*$-identifies $\mathcal{E}$. ∎

The next result says that for **Ex**-style learning with noisy texts or informants, restricting the data sequences to be computable does not help us.[13]

Theorem 8 *Suppose $a \in N \cup \{*\}$.*
(a) $\mathbf{NoisyTxtEx}^a = \mathbf{NoisyRecTxtEx}^a$.
(b) $\mathbf{NoisyInfEx}^a = \mathbf{NoisyRecInfEx}^a$.

PROOF. Clearly, $\mathbf{NoisyTxtEx}^a \subseteq \mathbf{NoisyRecTxtEx}^a$, and $\mathbf{NoisyInfEx}^a \subseteq \mathbf{NoisyRecInfEx}^a$. We show below that $\mathbf{NoisyTxtEx}^a \supseteq \mathbf{NoisyRecTxtEx}^a$, and $\mathbf{NoisyInfEx}^a \supseteq \mathbf{NoisyRecInfEx}^a$. Essentially the proof idea is to generalize the locking sequence arguments used to show $\mathbf{TxtEx}^a = \mathbf{RecTxtEx}^a$ to the noise setting.

(a) Suppose $\mathbf{M}$ $\mathbf{NoisyRecTxtEx}^a$-identifies $\mathcal{L}$.

Claim 1 *For each $L \in \mathcal{L}$, there exists a σ such that, for all τ satisfying $\mathrm{content}(\tau) \subseteq L$, $\mathbf{M}(\sigma) = \mathbf{M}(\sigma \diamond \tau)$.*

PROOF. Suppose by way of contradiction otherwise. Let $L \in \mathcal{L}$ be such that, for all σ there exists a τ, such that $\mathrm{content}(\tau) \subseteq L$, but $\mathbf{M}(\sigma) \neq \mathbf{M}(\sigma \diamond \tau)$. Suppose $W_i = L$. Define σ_0 to be empty sequence. For, $s \geq 0$, search (in some algorithmic way) for τ_s, τ'_s such that, $\mathrm{content}(\tau_s) \subseteq L$, $\mathrm{content}(\tau'_s) \subseteq L$, $\mathbf{M}(\sigma_s \diamond \tau_s) \neq \mathbf{M}(\sigma_s)$, and $\mathrm{content}(\tau'_s) \supseteq W_{i,s}$. Then, let $\sigma_{s+1} = \sigma_s \diamond \tau_s \diamond \tau'_s$. Note that, for all s such τ_s, τ'_s exist, and each σ_{s+1} is well defined. Let $T = \bigcup_{s \in N} \sigma_s$. Now T is a recursive noisy text for L, but $\mathbf{M}(T)\uparrow$. This, contradicts the hypothesis that $\mathbf{M}$ $\mathbf{NoisyRecTxtEx}^a$-identifies L. □

Claim 2 *Suppose T is a noisy text for $L \in \mathcal{L}$. Then,*
(i) there exists a σ and n such that $(\forall \tau \mid \mathrm{content}(\tau) \subseteq \mathrm{content}(T[n:\infty]))[\mathbf{M}(\sigma) = \mathbf{M}(\sigma \diamond \tau)]$, *and*
(ii) For all σ and n: If $(\forall \tau \mid \mathrm{content}(\tau) \subseteq \mathrm{content}(T[n:\infty]))[\mathbf{M}(\sigma) = \mathbf{M}(\sigma \diamond \tau)]$, *then* $W_{\mathbf{M}(\sigma)} =^a L$.

PROOF. Part (i) follows from Claim 1, and the fact that, for some n, $T[n:\infty]$ is a text for L. For part (ii) suppose $(\forall \tau \mid \mathrm{content}(\tau) \subseteq \mathrm{content}(T[n:\infty]))[\mathbf{M}(\sigma) = \mathbf{M}(\sigma \diamond \tau)]$. Note that $L \subseteq \mathrm{content}(T[n:\infty])$. Now, consider any recursive text T' for L such that each $x \in L$ appears infinitely often in T'. Then, $\mathbf{M}(\sigma \diamond T')\downarrow = \mathbf{M}(\sigma)$. Since $\mathbf{M}$ $\mathbf{NoisyRecTxtEx}^a$-identifies L, it follows that $W_{\mathbf{M}(\sigma)} =^a L$. □

Now construct $\mathbf{M}'$ as follows: $\mathbf{M}'$ on text T searches for a σ and n such that: $(\forall \tau \mid \mathrm{content}(\tau) \subseteq \mathrm{content}(T[n:\infty]))[\mathbf{M}(\sigma) = \mathbf{M}(\sigma \diamond \tau)]$. $\mathbf{M}'$ then outputs, in the limit on T, $\mathbf{M}(\sigma)$. It follows from Claim 2 that $\mathbf{M}'$ $\mathbf{NoisyTxtEx}^a$-identifies L. This proves part (a) of the theorem.

(b) Suppose $\mathbf{M}$ $\mathbf{NoisyRecInfEx}^a$-identifies $\mathcal{L}$.

---

[13] Suppose $a, b \in N \cup \{*\}$.

From [Cas99, BP73] we also have the following criteria intermediate between **Ex** style and **Bc** style.

$\mathbf{M}$ $\mathbf{TxtFex}^a_b$*-identifies L from text T* iff $(\exists S \mid \mathrm{card}(S) \leq b \ \wedge\ (\forall i \in S)[W_i =^a L])(\forall^\infty n)[\mathbf{M}(T[n]) \in S]$.

$\mathbf{M}$ $\mathbf{TxtFex}^a_b$*-identifies* L iff, for all texts T for L, $\mathbf{M}$ $\mathbf{TxtFex}^a_b$-identifies L from T. In this case we also write $L \in \mathbf{TxtFex}^a_b(\mathbf{M})$.

We say that $\mathbf{M}$ $\mathbf{TxtFex}^a_b$-identifies $\mathcal{L}$ iff $\mathbf{M}$ $\mathbf{TxtFex}^a_b$-identifies each $L \in \mathcal{L}$.

$\mathbf{TxtFex}^a_b = \{\mathcal{L} \mid (\exists \mathbf{M})[\mathcal{L} \subseteq \mathbf{TxtFex}^a_b(\mathbf{M})]\}$.

$\mathbf{InfFex}^a_b$ is defined similarly.

The definitions of the variants of these learning criteria involving noisy data or computable noisy data are handled similarly to such variants above.

By generalizing locking sequence arguments from [Cas99] and the present paper, Theorem 8 can be improved to say: $\mathbf{NoisyTxtFex}^a_b = \mathbf{NoisyRecTxtFex}^a_b$ and $\mathbf{NoisyInfFex}^a_b = \mathbf{NoisyRecInfFex}^a_b$.

Claim 3 *For each $L \in \mathcal{L}$, there exist σ and n such that, for all $\tau \in \mathrm{Inf}[\{x \mid x \leq n\}, L]$, $\mathbf{M}(\sigma) = \mathbf{M}(\sigma \diamond \tau)$.*

Proof. Suppose by way of contradiction, for some $L \in \mathcal{L}$, for all σ, n, there exists a $\tau \in \mathrm{Inf}[\{x \mid x \leq n\}, L]$, such that $\mathbf{M}(\sigma) \neq \mathbf{M}(\sigma \diamond \tau)$. Then, we construct a recursive noisy information sequence I for L such that $\mathbf{M}(I)\uparrow$. Suppose i is a grammar for L. We will define $\sigma_0, \sigma_1, \ldots$, such that $I = \bigcup_{i \in N} \sigma_i$. Let $\sigma_0 = \Lambda$. Suppose σ_s, has been defined. Then σ_{s+1} is defined as follows.

Definition of σ_{s+1}

1. Search for a $t > s$, and τ_s such that (a) and (b) are satisfied:
   (a) $\mathbf{M}(\sigma_s) \neq \mathbf{M}(\sigma_s \diamond \tau_s)$.
   (b) For all $x \leq \min(\{s\} \cup (W_{i,t} - W_{i,s}))$
      If $(x, 1) \in \mathrm{content}(\tau_s)$, then $x \in W_{i,t}$, and
      If $(x, 0) \in \mathrm{content}(\tau_s)$, then $x \notin W_{i,t}$].
2. If and when such τ_s and t are found, let $\sigma_{s+1} = \sigma \diamond \tau_s \diamond \tau'_s$, where $\mathrm{content}(\tau'_s) = \{(x, 1) \mid x \leq s \wedge x \in W_{i,s}\} \cup \{(x, 0) \mid x \leq s \ \wedge \ x \notin W_{i,s}\}$.

End

We claim that, (i) for all s, search in step 1 of the definition of σ_{s+1} succeeds, and (ii) $I = \bigcup_{s \in N} \sigma_s$ is a noisy informant for L. This (using $\mathbf{M}(I)\uparrow$) would prove the claim. To see (i), let $t' = \min(\{t'' \mid W_i \cap \{x \mid x \leq s\} \subseteq W_{i,t''}\})$. Let τ' be such that $\tau' \in \mathrm{Inf}[\{x \mid x \leq s\}, L]$ and $\mathbf{M}(\sigma_s) \neq \mathbf{M}(\sigma_s \diamond \tau')$. Then, $t = t'$ and $\tau_s = \tau'$ witness that search in step 1 succeeds. To see (ii), for any x, let $s = \max(\{x\} \cup \min(\{t \mid W_i \cap \{y \mid y \leq x\} \subseteq W_{i,t}\}))$. Then, for all $s' > s$, $\tau_{s'} \diamond \tau'_{s'}$ as in the definition of $\sigma_{s'+1}$, will satisfy: $(x, 1) \in \mathrm{content}(\tau_{s'} \diamond \tau'_{s'})$ iff $x \in L$ and $(x, 0) \in \mathrm{content}(\tau_{s'} \diamond \tau'_{s'})$ iff $x \notin L$. Thus, I is a noisy informant for L. $\square$

Claim 4 *Suppose I is a noisy informant for $L \in \mathcal{L}$. Then,*

(i) there exist σ, n and m such that, $(\forall \tau \in \mathrm{Inf}[\{x \mid x \leq n\}, \mathit{PosInfo}(I[m])])[\mathbf{M}(\sigma) = \mathbf{M}(\sigma \diamond \tau)]$, and $(\forall x \leq n)(\forall m' \geq m)[I(m') \neq (x, 1 - \chi_{\mathit{PosInfo}(I[m])})]$.

(ii) for all σ, n and m: If $(\forall \tau \in \mathrm{Inf}[\{x \mid x \leq n\}, \mathit{PosInfo}(I[m])])[\mathbf{M}(\sigma) = \mathbf{M}(\sigma \diamond \tau)]$, and $(\forall x \leq n)(\forall m' \geq m)[I(m') \neq (x, 1 - \chi_{\mathit{PosInfo}(I[m])})]$, then $W_{\mathbf{M}(\sigma)} =^a L$.

Proof. Part (i) follows from Claim 3 by picking m such that, for $x \leq n$, $x \in \mathrm{PosInfo}(I[m])$ iff $x \in L$ and $(\forall x \leq n)(\forall m' \geq m)[I(m') \neq (x, 1 - \chi_{\mathrm{PosInfo}(I[m])})]$.

For part (ii), suppose σ, n and m are given satisfying the hypothesis. Let I' be a recursive noisy informant for L, such that, for all $x \leq n$, $(x, 1) \in \mathrm{content}(I')$ iff $x \in L$, and $(x, 0) \in \mathrm{content}(I')$ iff $x \notin L$. Note that there exists such a recursive noisy informant. Now we have by hypothesis that $\mathbf{M}(\sigma \diamond I') = \mathbf{M}(\sigma)$. Since $\mathbf{M}$ $\mathbf{NoisyRecInfEx}^a$-identifies L, it follows that $W_{\mathbf{M}(\sigma)} =^a L$. $\square$

Now construct $\mathbf{M}'$ as follows: $\mathbf{M}'$ on noisy informant I, searches for a σ, n and m such that: $(\forall \tau \in \mathrm{Inf}[\{x \mid x \leq n\}, \mathrm{PosInfo}(I[m])])[\mathbf{M}(\sigma) = \mathbf{M}(\sigma \diamond \tau)]$, and $(\forall x \leq n)(\forall m' \geq m)[I(m') \neq (x, 1 - \chi_{\mathrm{PosInfo}(I[m])})]$, Note that such σ, n and m can be found in the limit, if they exist. $\mathbf{M}'$ then outputs, in the limit on I, $\mathbf{M}(\sigma)$. It follows from Claim 4 that $\mathbf{M}'$ $\mathbf{NoisyInfEx}^a$-identifies every $L \in \mathcal{L}$. This proves part (b) of the theorem. ∎

Theorem 9 *Suppose $n \in N$.*

(a) $\mathbf{NoisyTxtEx} - \mathbf{NoisyRecInfBc}^n \neq \emptyset$.

(b) $\mathbf{NoisyInfEx} - \mathbf{NoisyRecTxtBc}^n \neq \emptyset$.

PROOF. (a) Let $L_0 = \{\langle x, 0\rangle \mid x \in N\}$. For $i > 0$, let $L_i = \{\langle x, 0\rangle \mid x \leq i\} \cup \{\langle x, i\rangle \mid x > i\}$. Let $\mathcal{L} = \{L_i \mid i \in N\}$. It is easy to verify that $\mathcal{L} \in \mathbf{NoisyTxtEx}$. Suppose by way of contradiction that $\mathbf{M}$ $\mathbf{NoisyRecInfBc}^n$-identifies $\mathcal{L}$. Let σ_0 be empty sequence. Go to stage 0.

Stage s

1. Search for a $\tau_s \in \mathrm{Inf}[\{y \mid y \leq s\}, L_0]$ such that, $W_{\mathbf{M}(\sigma_s \diamond \tau_s)}$ enumerates at least $n+1$ elements not in L_0.
2. If and when such a τ_s is found, let $\sigma_{s+1} = \sigma_s \diamond \tau_s \diamond \tau'_s$, where $\mathrm{content}(\tau'_s) = \{(y, \chi_{L_0}(y)) \mid y \leq s\}$.
   Go to stage $s+1$.

End stage s

We now consider the following cases:

Case 1: There exist infinitely many stages.

In this case $I = \bigcup_{s \in N} \sigma_s$ is a recursive noisy informant for L_0. However, $\mathbf{M}$ on I infinitely often (at each $\sigma_s \diamond \tau_s$) outputs a grammar, which enumerates at least $n+1$ elements not in L_0.

Case 2: Stage s starts but does not finish.

Let $i > 0$ be such that $L_i \cap \{y \mid y \leq s\} = L_0 \cap \{y \mid y \leq s\}$. Let I be a recursive informant for L_i, in which each $(x, \chi_{L_i}(x))$ appears infinitely often. $\mathbf{M}$ does not $\mathbf{InfBc}^n$-identify L_i from $\sigma_s \diamond I$ (since, for each $\tau \subseteq I$, $\mathbf{M}(\sigma \diamond \tau)$ does not enumerate more than n elements of $L_i - L_0$). It follows that $\mathbf{M}$ does not $\mathbf{NoisyRecInfBc}^n$-identify L_i.

From the above cases it follows that $\mathcal{L} \notin \mathbf{NoisyRecInfBc}^n$.

(b) Let $\mathcal{L} = \{L \mid W_{\min(L)} = L\}$. Clearly, $\mathcal{L} \in \mathbf{NoisyInfEx}$. We show that $\mathcal{L} \notin \mathbf{NoisyRecTxtBc}^n$. Suppose by way of contradiction, $\mathbf{M}$ $\mathbf{NoisyRecTxtBc}^n$-identifies $\mathcal{L}$. Then, by operator recursion theorem [Cas74], there exists a recursive, 1–1, increasing function p, such that $W_{p(\cdot)}$ may be defined as follows. For all $i > 0$, $W_{p(i)} = \{p(j) \mid j \geq i\}$. Note that $W_{p(i)} \in \mathcal{L}$, for all $i \geq 1$. We will define $W_{p(0)}$ below. It will be the case that $W_{p(0)} \in \mathcal{L}$. Let σ_0 be such that $\mathrm{content}(\sigma_0) = \{p(0)\}$. Enumerate $p(0)$ in $W_{p(0)}$. Let $q_0 = 1$. Go to stage 0.

Stage s

1. Search for a τ_s and set S such that $\mathrm{content}(\tau_s) \subseteq W_{p(q_s)}$, $\mathrm{card}(S) \geq n+1$, $S \cap \mathrm{content}(\sigma_s \diamond \tau_s) = \emptyset$, and $S \subseteq W_{\mathbf{M}(\sigma_s \diamond \tau_s)}$.
2. If and when such a τ_s is found, let
   $X = \mathrm{content}(\sigma_s \diamond \tau_s)$.
   Enumerate X in $W_{p(0)}$.
   Let τ'_s be such that $\mathrm{content}(\tau'_s) = X$.
   Let $\sigma_{s+1} = \sigma_s \diamond \tau_s \diamond \tau'_s$.
   Let $q_{s+1} = 1 + \max(\{w \mid p(w) \in X \cup S\})$.
   Go to stage $s+1$.

End stage s

We now consider two cases.

Case 1: All stages halt.

In this case, let $T = \bigcup_{s \in N} \sigma_s$. Clearly, T is a noisy recursive text for $W_{p(0)} \in \mathcal{L}$, and $\mathbf{M}$ does not $\mathbf{TxtBc}^n$-identifies $W_{p(0)}$ from T, since, for each s, $\mathbf{M}(\sigma_s \diamond \tau_s)$ enumerates at least $n+1$ elements not in $W_{p(0)}$.

Case 2: Stage s starts but does not halt.

In this case let $L = W_{p(q_s)}$. Clearly, $L \in \mathcal{L}$. Let T be a recursive text for L such that every element for L appears infinitely often in T. Now, $\mathbf{M}$ does not $\mathbf{TxtBc}^n$-identify L from $\sigma_s \diamond T$, since $\mathbf{M}(\sigma_s \diamond \tau)$ is finite for all $\tau \subseteq T$ (otherwise step 1 in stage s would succeed).

It follows from the above cases that $\mathbf{M}$ does not $\mathbf{NoisyRecTxtBc}^n$-identify $\mathcal{L}$. ∎

Theorem 10 *Suppose $n \in N$.*

(a) $\mathbf{NoisyTxtBc}^{n+1} - \mathbf{RecInfBc}^n \neq \emptyset$.

(b) $\mathbf{NoisyInfBc}^{n+1} - \mathbf{RecInfBc}^n \neq \emptyset$.

Proof. The main idea is to modify the construction of $\mathbf{Bc}^{n+1} - \mathbf{Bc}^n$ in [CS83].

(a) Let $\mathcal{L} = \{L \in \mathrm{REC} \mid \mathrm{card}(L) = \infty \wedge (\forall^\infty x \in L)[W_x =^{n+1} L]\}$. Clearly, $\mathcal{L} \in \mathbf{NoisyTxtBc}^{n+1}$. An easy modification of the proof of $\mathbf{Bc}^{n+1} - \mathbf{Bc}^n \neq \emptyset$ in [CS83] shows that $\mathcal{L} \notin \mathbf{RecInfBc}^n$. We omit the details.

(b) Let $\mathcal{L} = \{L \in \mathrm{REC} \mid (\forall x \in W_{\min(L)})[W_x =^{n+1} L] \vee [\mathrm{card}(W_{\min(L)}) < \infty \wedge W_{\max(W_{\min(L)})} =^{n+1} L]\}$. It is easy to verify that $\mathcal{L} \in \mathbf{NoisyInfBc}^{n+1}$. An easy modification of the proof of $\mathbf{Bc}^{n+1} - \mathbf{Bc}^n \neq \emptyset$ in [CS83] shows that $\mathcal{L} \notin \mathbf{RecInfBc}^n$. We omit the details. ∎

Theorem 11 *(a)* $\mathbf{NoisyRecTxtBc} - \mathbf{TxtBc}^* \neq \emptyset$.

(b) $\mathbf{NoisyRecInfBc} - \mathbf{TxtBc}^* \neq \emptyset$.

Proof. (a) Corollary 1 below shows that all indexed families are in $\mathbf{NoisyRecTxtBc}$. However, $\mathcal{L} = \{L \mid \mathrm{card}(L) < \infty\} \cup \{N\}$ is an indexed family which is not in $\mathbf{TxtBc}^*$.

(b) Let $L_0 = N$, and for $i > 0$, $L_i = \{x \mid x \leq i\}$. Let $\mathcal{L} = \{L_i \mid i \in N\}$. Note that $\mathcal{L} \notin \mathbf{TxtBc}^*$ (essentially due to [Gol67]). Now let $z_i = i + 1$. It is easy to verify that, for all i, for all $L' \in N$, if $L' \cap \{x \mid x \leq z_i\} = L \cap \{x \mid x \leq z_i\}$, then $L' \subseteq L$. It follows from Corollary 3 below that $\mathcal{L} \in \mathbf{NoisyRecInfBc}$. ∎

It is open at present whether, for $m \leq n$, (i) $\mathbf{NoisyRecTxtBc}^m - \mathbf{InfBc}^n \neq \emptyset$? and whether (ii) $\mathbf{NoisyRecInfBc}^m - \mathbf{InfBc}^n \neq \emptyset$? In this context note that

Theorem 12 $\mathbf{RecTxtBc}^a \cap 2^{\mathrm{REC}} \subseteq \mathbf{InfBc}^a$.

Proof. For a recursive language L, consider the text:

$$T_L(x) = \begin{cases} \#, & \text{if } x \notin L; \\ x, & \text{otherwise.} \end{cases}$$

Note that T_L is a recursive text for L. Moreover, T_L can be obtained algorithmically from an informant for L. Thus, one can convert an informant for a recursive language algorithmically into a recursive text for L. It follows that $\mathbf{RecTxtBc}^a \cap 2^{\mathrm{REC}} \subseteq \mathbf{InfBc}^a$. ∎

4 Principal Results on Synthesizers

Since $\mathcal{E} \in \mathbf{NoisyRecTxtBc}^*$ and $\mathcal{E} \in \mathbf{NoisyRecInfBc}^*$, the only cases of interest are regarding when $\mathbf{NoisyRecTxtBc}^n$ and $\mathbf{NoisyRecInfBc}^n$ synthesizers can be obtained algorithmically.

4.1 Principal Results on Synthesizing From Uniform Decision Indices

The next result is the main theorem of the present paper.

Theorem 13 $(\exists f \in \mathcal{R})(\forall i)[\mathcal{U}_i \subseteq \mathbf{NoisyRecTxtBc}(\mathbf{M}_{f(i)})]$.

PROOF. Let $\mathbf{M}_{f(i)}$ be such that, $\mathbf{M}_{f(i)}(T[n]) = \mathit{prog}(T[n])$, where, $W_{\mathit{prog}(T[n])}$ is defined as follows. Construction of prog will easily be seen to be algorithmic in i.

If $\mathcal{U}_i$ is empty, then trivially $\mathbf{M}_{f(i)}$ **NoisyRecTxtBc**-identifies $\mathcal{U}_i$. So suppose $\mathcal{U}_i$ is nonempty (in particular, for all $j \in W_i$, j is a decision procedure). In the construction below, we will thus assume without loss of generality that, for each $j \in W_i$, j is a decision procedure.

Let g be a computable function such that, range$(g) = \{\langle j, k\rangle \mid j \in W_i \ \wedge \ k \in N\}$. Intuitively, for an input noisy recursive text T for a language L, think of m such that $g(m) = \langle j, k\rangle$ as representing the hypothesis: (i) $L = U_j$, (ii) $\varphi_k = T$, and (iii) $T[m:\infty]$ does not contain any element from $\overline{L}$. In the procedure below, we just try to collect "non-harmful" and "good" hypothesis in P_n and Q_n^s (more details on this in the analysis of $\mathit{prog}(T[n])$ below). Let P1 and P2 be recursive functions such that $g(m) = \langle \mathrm{P1}(m), \mathrm{P2}(m)\rangle$.

$W_{\mathit{prog}(T[n])}$

1. Let $P_n = \{m \mid m \leq n\} - [\{m \mid \mathrm{content}(T[m:n]) \not\subseteq U_{\mathrm{P1}(m)}\} \cup \{m \mid (\exists k < n)[\Phi_{\mathrm{P2}(m)}(k) \leq n \ \wedge \ \varphi_{\mathrm{P2}(m)}(k) \neq T(k)]\}]$.

   (* Intuitively, P_n is obtained by deleting $m \leq n$ which represent a clearly wrong hypothesis. *)

   (* Q_n^s below is obtained by refining P_n so that some further properties are satisfied. *)

2 Let $Q_n^0 = P_n$.

   Go to stage 0.

3. Stage s

   3.1 Enumerate $\bigcap_{m \in Q_n^s} U_{\mathrm{P1}(m)}$.

   3.2 Let $Q_n^{s+1} = Q_n^s - \{m' \mid (\exists m'' \in Q_n^s)(\exists k \leq s)[m'' < m' \leq k \ \wedge \ [\Phi_{\mathrm{P2}(m'')}(k) \leq s \ \wedge \ \varphi_{\mathrm{P2}(m'')}(k) \notin U_{\mathrm{P1}(m')}]]\}$.

   3.3 Go to stage $s+1$.

   End stage s.

End

Let T be a noisy recursive text for $L \in \mathcal{U}_i$. Let m be such that $U_{\mathrm{P1}(m)} = L$, $T[m:\infty]$ is a text for L, and $\varphi_{\mathrm{P2}(m)} = T$. Note that there exists such an m (since φ is acceptable numbering, and T is a noisy recursive text for L). Consider the definition of $W_{\mathit{prog}(T[n])}$ for $n \in N$ as above.

Claim 5 *For all $m' \leq m$, for all but finitely many n, if $m' \in P_n$ then*

(a) $L \subseteq U_{\mathrm{P1}(m')}$, and

(b) $(\forall k)[\varphi_{\mathrm{P2}(m')}(k)\uparrow \ \vee \ \varphi_{\mathrm{P2}(m')}(k) = T(k)]$.

PROOF. Suppose $m' \leq m$.

(a) If $U_{\mathrm{P1}(m')} \not\supseteq L$, then there exists a $k > m'$ such that $T(k) \notin U_{\mathrm{P1}(m')}$. Thus, for $n > k$, $m' \notin P_n$.

(b) If there exists a k such that $[\varphi_{\mathrm{P2}(m')}(k)\downarrow \neq T(k)]$, then for all $n > \max(\{k, \Phi_{\mathrm{P2}(m')}(k)\})$, $m' \notin P_n$.

The claim follows. □

Claim 6 *For all but finitely many n: $m \in P_n$.*

PROOF. For $n \geq m$, clearly $m \in P_n$. □

Let n_0 be such that, for all $n \geq n_0$, (a) $m \in P_n$, and (b) for all $m' \leq m$, if $m' \in P_n$, then $L \subseteq U_{\mathrm{P1}(m')}$ and $(\forall k)[\varphi_{\mathrm{P2}(m')}(k)\uparrow \vee \varphi_{\mathrm{P2}(m')}(k) = T(k)]$. (There exists such a n_0 by Claims 5 and 6.)

Claim 7 *Consider any $n \geq n_0$. Then, for all s, we have $m \in Q_n^s$. It follows that $W_{prog(T[n])} \subseteq L$.*

PROOF. Fix $n \geq n_0$. The only way m can be missing from Q_n^s is the existence of $m'' < m$, and $t > m$ such that $m'' \in P_n$, and $\varphi_{\mathrm{P2}(m'')}(t)\downarrow \notin L$. But then $m'' \notin P_n$ by the condition on n_0. Thus $m \in Q_n^s$, for all s. □

Claim 8 *Consider any $n \geq n_0$. Suppose $m \leq m' \leq n$. If $(\exists^\infty s)[m' \in Q_n^s]$, then $L \subseteq U_{\mathrm{P1}(m')}$. Note that, using the condition on n_0, this claim implies $L \subseteq W_{prog(T[n])}$.*

PROOF. Fix any $n \geq n_0$. Suppose $(\exists^\infty s)[m' \in Q_n^s]$. Thus, $(\forall s)[m' \in Q_n^s]$. Suppose $L \not\subseteq U_{\mathrm{P1}(m')}$. Let $y \in L - U_{\mathrm{P1}(m')}$. Let $k \geq m'$ be such that $T(k) = y$. Note that there exists such a k, since y appears infinitely often in T. But then $\varphi_{\mathrm{P2}(m)}(k)\downarrow \notin U_{\mathrm{P1}(m')}$. This would imply that $m' \notin Q_n^s$, for some s, by step 3.2 in the construction. Thus, $L \subseteq U_{\mathrm{P1}(m')}$, and claim follows. □

From Claims 7 and 8 it follows that, for $n \geq n_0$, $W_{prog(T[n])} = L$. Thus, $\mathbf{M}_{f(i)}$ **NoisyRecTxtBc**-identifies $\mathcal{U}_i$. ∎

As a corollary we get the following result.

Corollary 1 *Every indexed family belongs to* **NoisyRecTxtBc**.

As noted in Section 1 above, then, the class of finite unions of pattern languages *is* **NoisyRecTxtBc**-learnable.

Remark 1 *In the above theorem, learnability is* not *obtained by learning the rule for generating the noise. In fact, in general, it is* impossible *to learn (in the* **Bc**-*sense) the rule for noisy text generation (even though the noisy text is computable).*

While the **NoisyRecTxtBc**^{a}-hierarchy collapses *for indexed families*, we see below that the **NoisyRecInfBc**^{a}-hierarchy does not so collapse.

Lemma 2 *Let $n \in N$.*

(a) Suppose L is a recursive language, and **M** **NoisyRecInfBc**^{n}*-identifies L. Then there exists a σ and z such that* $(\forall \tau \in \mathrm{Inf}[\{x \mid x \leq z\}, L])[\mathrm{card}(W_{\mathbf{M}(\sigma \diamond \tau)} - L) \leq n]$.

(b) Suppose $\mathcal{L}$ is an indexed family in **NoisyRecInfBc**^{n}*. Then, for all $L \in \mathcal{L}$, there exists a z such that, for all $L' \in \mathcal{L}$,* $[(\{x \leq z \mid x \in L\} = \{x \leq z \mid x \in L'\}) \Rightarrow (\mathrm{card}(L' - L) \leq 2n)]$.

PROOF. (a) Suppose by way of contradiction otherwise. Thus

$$(\forall \sigma)(\forall z)(\exists \tau \in \mathrm{Inf}[\{x \mid x \leq z\}, L])[\mathrm{card}(W_{M(\sigma \diamond \tau)} - L) > n]$$

We will construct a recursive noisy informant I for L such that, for infinitely many m, $W_{\mathbf{M}(I[m])} \neq^n L$. This would contradict the hypothesis that **M** **NoisyRecInfBc**^{n}-identifies L. Note that L is recursive. So one can algorithmically determine whether $\tau \in \mathrm{Inf}[\{x \mid x \leq z\}, L]$. Initially let σ_0 be empty sequence. Go to stage 0.

Stage s

1. Search for a $\tau_s \in \text{Inf}[\{x \mid x \leq s\}, L]$ such that $\text{card}(W_{\mathbf{M}(\sigma_s \diamond \tau_s)} - L) > n$.
2. If and when such τ_s is found, let τ_s' be such that $\text{content}(\tau_s') = \{(x, \chi_L(x)) \mid x \leq s\}$.
   (That is, τ_s' is the first $s+1$ elements of the canonical information sequence for L)
   Let $\sigma_{s+1} = \sigma_s \diamond \tau_s \diamond \tau_s'$.
   Go to stage $s+1$.

End

First note that the search for τ_s succeeds in every stage (otherwise σ_s and s witness part (a) of the lemma). Let $I = \bigcup_{s \in N} \sigma_s$. Now I is recursive and is a noisy informant for L (since $(x, 1-\chi_L(x))$ does not appear in I beyond σ_x, and $(x, \chi_L(x))$ appears in τ_s', for every $s \geq x$). However, $W_{\mathbf{M}(\sigma_s \diamond \tau_s)} \neq^n L$, for every s. Thus, $\mathbf{M}$ does not $\mathbf{NoisyRecTxtBc}^n$-identify L. This proves part (a) of the lemma.

(b) Suppose $\mathbf{M}$ $\mathbf{NoisyRecInfBc}^n$-identifies $\mathcal{L}$. Let σ and z be such that $(\forall \tau \in \text{Inf}[\{x \mid x \leq z\}, L])[\text{card}(W_{\mathbf{M}(\sigma \diamond \tau)} - L) \leq n]$ (by part (a) there exist such σ and z). Consider, any $L' \in \mathcal{L}$ such that $\{x \leq z \mid x \in L\} = \{x \leq z \mid x \in L'\}$. Consider any recursive informant I for L' such that, for all x, $(x, \chi_{L'}(x))$ appears infinitely often in I. Now, for all $\tau \subseteq I$, $\text{card}(W_{\mathbf{M}(\sigma \diamond \tau)} - L) \leq n$. Since, for all but finitely many $\tau \subseteq I$, $W_{\mathbf{M}(\sigma \diamond \tau)} =^n L'$, it follows that $\text{card}(L' - L) \leq 2n$. ∎

Theorem 14 *Suppose* $n \in N$. $\{L \mid \text{card}(L) \leq 2(n+1)\} \in \mathbf{NoisyInfBc}^{n+1} - \mathbf{NoisyRecInfBc}^n$.

Proof. For a finite set S, let $prog(S)$ denote a grammar for S, algorithmically obtained from S. Let $\mathbf{M}(I[m]) = prog(S)$, where S is the least $n+1$ elements in $\text{PosInfo}(I[m])$ (if $\text{PosInfo}(I[m])$ contains less than $n+1$ elements, then $S = \text{PosInfo}(I[m])$). Now consider any $L \in \mathcal{L}$. Let I be a noisy informant for L.

We consider two cases:

Case 1: $\text{card}(L) \geq n+1$.

Let S' denote the least $n+1$ elements of L. Now, for all but finitely many m, S' is the set of least $n+1$ elements in $\text{PosInfo}(I[m])$. It follows that, for all but finitely many m, $\mathbf{M}(I[m]) = prog(S')$. Thus $\mathbf{M}$ $\mathbf{NoisyInfBc}^{n+1}$-identifies L.

Case 2: $\text{card}(L) \leq n+1$.

In this case, for all but finitely many m, L is the set of least $\text{card}(L)$ elements in $\text{PosInfo}(I[m])$. It follows that, for all but finitely many m, $L \subseteq W_{\mathbf{M}(I[m])}$. Since $W_{\mathbf{M}(I[m])}$ contains at most $n+1$ elements, it follows that $\mathbf{M}$ $\mathbf{NoisyInfBc}^{n+1}$-identifies L.

Thus $\mathcal{L} \in \mathbf{NoisyInfBc}^{n+1}$.

We now show that $\mathcal{L} \notin \mathbf{NoisyRecInfBc}^n$. Suppose by way of contradiction that $\mathcal{L} \in \mathbf{NoisyRecInfBc}^n$. Note that $\emptyset \in \mathcal{L}$. Thus, by Lemma 2(b), there exists a z such that, for all $L' \in \mathcal{L}$, $[\{x \mid x \leq z\} \cap L' = \emptyset] \Rightarrow [\text{card}(L') \leq 2n]$. Now clearly there are languages $L' \in \mathcal{L}$ of cardinality $2n+1$ such that $\{x \mid x \leq z\} \cap L' = \emptyset$. It follows that $\mathcal{L} \notin \mathbf{NoisyRecInfBc}^n$. ∎

We will see in Corollary 2 below that it is possible to algorithmically synthesize learners for $\mathbf{NoisyRecInfBc}$-learnable indexed families.

Theorem 15 *There exists* $f \in \mathcal{R}$ *such that the following is satisfied. Suppose* $(\forall L \in \mathcal{U}_i)(\exists z)(\forall L' \in \mathcal{U}_i)[(\{x \leq z \mid x \in L\} = \{x \leq z \mid x \in L'\}) \Rightarrow L' \subseteq L]$. *Then,* $[\mathcal{U}_i \in \mathbf{NoisyRecInfBc}(\mathbf{M}_{f(i)})]$.

PROOF. Let $\mathbf{M}_{f(i)}$ be such that, $\mathbf{M}_{f(i)}(I[n]) = prog(I[n])$, where, $W_{prog(I[n])}$ is defined as follows. Construction of $prog$ will easily be seen to be algorithmic in i.

If $\mathcal{U}_i$ is empty, then trivially $\mathbf{M}_{f(i)}$ **NoisyRecInfBc**-identifies $\mathcal{U}_i$. So suppose $\mathcal{U}_i$ is nonempty (in particular, for all $j \in W_i$, j is a decision procedure). In the construction below, we will thus assume without loss of generality that, for each $j \in W_i$, j is a decision procedure.

Let g be a computable function such that, range$(g) = \{\langle j, k, \ell\rangle \mid j \in W_i \ \wedge\ k, \ell \in N\}$. Intuitively, for an input noisy recursive informant I for a language L, think of m such that $g(m) = \langle j, k, l\rangle$ as representing the hypothesis: (i) $L = U_j$, (ii) $\varphi_k = I$, (iii) $\ell = z$ as in Lemma 2(b) for $L = U_j$ and $\mathcal{L} = \mathcal{U}_i$, and (iv) $(\forall x \leq \ell)(\forall t \geq m)[I(t) \neq (x, 1 - \chi_L(x))]$ (see more details on this in the analysis of $prog(I[n])$ below).

Let P1, P2 and P3 be recursive functions such that $g(m) = \langle \mathrm{P1}(m), \mathrm{P2}(m), \mathrm{P3}(m)\rangle$.

$W_{prog(I[n])}$

1. Let $P_n = \{m \mid m \leq n\} - [\{m \mid (\exists x \leq \mathrm{P3}(m))(\exists t \mid m \leq t < n)[I(t) = (x, 1 - \chi_{U_{\mathrm{P1}(m)}}(x))]\} \cup \{m \mid (\exists k < n)[\Phi_{\mathrm{P2}(m)}(k) \leq n \ \wedge\ \varphi_{\mathrm{P2}(m)}(k) \neq I(k)]\}]$.

   (* Intuitively, P_n is obtained by deleting $m \leq n$ which represent a clearly wrong hypothesis. *)

   (* Q_n^s below is obtained by refining P_n so that some further properties are satisfied. *)

2 Let $Q_n^0 = P_n$.

   Go to stage 0.

3. Stage s

   3.1 Enumerate $\bigcap_{m \in Q_n^s} U_{\mathrm{P1}(m)}$.

   3.2 Let

   $A_s = \{m' \in Q_n^s \mid (\exists m'' \leq s)[(\forall x \leq \mathrm{P3}(m'))[x \in U_{\mathrm{P1}(m')} \Leftrightarrow x \in U_{\mathrm{P1}(m'')}] \ \wedge\ (\exists y \leq s)[y \in U_{\mathrm{P1}(m'')} - U_{P1(m')}]]\}$.

   $B_s = \{m' \in Q_n^s \mid (\exists m'' \in Q_n^s)[m'' < m' \ \wedge\ (\exists k, x \mid m' \leq k \leq s \ \wedge\ x \leq \mathrm{P3}(m'))[\Phi_{\mathrm{P2}(m'')}(k) \leq s \ \wedge\ \varphi_{\mathrm{P2}(m'')}(k) = (x, 1 - \chi_{U_{P1(m')}}(x))]]\})$.

   $Q_n^{s+1} = Q_n^s - (A_s \cup B_s)$.

   Go to stage $s+1$.

   End stage s.

End

Let I be a noisy recursive informant for $L \in \mathcal{U}_i$. Let m be such that (a) $U_{\mathrm{P1}(m)} = L$, (b) $I = \varphi_{\mathrm{P2}(m)}$, (c) for all $L' \in \mathcal{U}_i$, $[\{x \leq \mathrm{P3}(m) \mid x \in L\} = \{x \leq \mathrm{P3}(x) \mid x \in L'\} \Rightarrow L' \subseteq L]$, and (d) $(\forall t \geq m)(\forall x \leq \mathrm{P3}(m))[I(t) \neq (x, 1 - \chi_L(x))]$. Note that there exists such an m (since φ is acceptable numbering, I is a noisy recursive informant for L, and using Lemma 2(b)). Consider the definition of $W_{prog(I[n])}$ for $n \in N$ as above.

Claim 9 *For all $m' \leq m$, for all but finitely many n, if $m' \in P_n$ then*

(a) for all $x \leq \mathrm{P3}(m')$, $x \in U_{\mathrm{P1}(m')} \Leftrightarrow x \in L$.

(b) $(\forall k, x \mid m \leq k \ \wedge\ x \leq \mathrm{P3}(m))[\varphi_{\mathrm{P2}(m')}(k)\uparrow \ \vee\ \varphi_{\mathrm{P2}(m')}(k) \neq (x, 1 - \chi_L(x))]$.

PROOF. Consider any $m' \leq m$. (a) If there exists a $x \leq \mathrm{P3}(m')$ such that $x \in L \boldsymbol{\Delta} U_{\mathrm{P1}(m')}$, then there exists a $t > m'$ such that $I(t) = (x, \chi_L(x)) = (x, 1 - \chi_{U_{\mathrm{P1}(m')}}(x))$. Thus, for large enough n, $m' \notin P_n$.

(b) Suppose k, x are such that $m \leq k$, $x \leq \mathrm{P3}(m)$ and $\varphi_{\mathrm{P2}(m')}(k)\downarrow = (x, 1 - \chi_L(x))$. Then due to the definition of m, $I(k) = (x, \chi_L(x)) \neq \varphi_{\mathrm{P2}(m')}(k)$. Thus, for large enough n, $m' \notin P_n$. $\Box$

Claim 10 *For all but finitely many n: $m \in P_n$.*

PROOF. For $n \geq m$, clearly $m \in P_n$. $\square$

Let n_0 be such that for all $n \geq n_0$, $m \in P_n$, and for all $m' \leq m$, if $m' \in P_n$ then (a) for all $x \leq \mathrm{P3}(m')$, $x \in U_{\mathrm{P1}(m')} \Leftrightarrow x \in L$, and (b) $(\forall k, x \mid m \leq k \ \wedge \ x \leq \mathrm{P3}(m))[\varphi_{\mathrm{P2}(m')}(k)\uparrow \ \vee \ \varphi_{\mathrm{P2}(m')}(k) \neq (x, 1 - \chi_L(x))]$.

Claim 11 *Consider any $n \geq n_0$. Then, for all s, we have $m \in Q_n^s$. It follows that $W_{prog(T[n])} \subseteq L$.*

PROOF. Fix $n \geq n_0$, and consider the computation of $W_{prog(I[n])}$. Note that, by the definition of m and hypothesis of the theorem, m cannot belong to A_s in stage s in the computation of $W_{prog(I[n])}$. We now show that m cannot belong to B_s either. Suppose $m'' < m$, $m'' \in P_n$, a $t > m$, and a $x \leq \mathrm{P3}(m)$, are such that $\varphi_{\mathrm{P2}(m'')}(t)\downarrow = (x, 1 - \chi_L(x))$. But then $m'' \notin P_n$ by the requirements on n_0. Thus $m \in Q_n^s$, for all s. $\square$

Claim 12 *Consider any $n \geq n_0$. Suppose $m' \leq n$. If $(\exists^\infty s)[m' \in Q_n^s]$, then $L \subseteq U_{\mathrm{P1}(m')}$. Note that this implies $L \subseteq W_{prog(T[n])}$.*

PROOF. Fix any $n \geq n_0$, and consider the computation of $W_{prog(I[n])}$. Suppose $(\exists^\infty s)[m' \in Q_n^s]$. Thus, $(\forall s)[m' \in Q_n^s]$. Suppose $L \not\subseteq U_{\mathrm{P1}(m')}$. Let $w \in L - U_{\mathrm{P1}(m')}$. We consider two cases:

Case 1: $m' < m$.

In this case, by the condition on n_0, we have that, $(\forall x \leq \mathrm{P3}(m'))[x \in U_{\mathrm{P1}(m')} \Leftrightarrow x \in L = U_{\mathrm{P1}(m)}]$. Thus, for large enough s, $m' \notin Q_n^s$ (since for $m'' = m$ and $y = w$, $(\forall x \leq \mathrm{P3}(m'))[x \in U_{\mathrm{P1}(m')} \Leftrightarrow x \in U_{\mathrm{P1}(m'')}] \ \wedge \ [y \in U_{\mathrm{P1}(m'')} - U_{P1(m')}]$, and thus $m' \in A_s$ for large enough s).

Case 2: $m' > m$.

Case 2.1: For all $x \leq \mathrm{P3}(m')$, $[x \in U_{\mathrm{P1}(m')} \Leftrightarrow x \in U_{\mathrm{P1}(m)}]$.

In this case, as in Case 1, for large enough s, $m' \notin \mathrm{P3}(m')$.

Case 2.2: For some $x \leq \mathrm{P3}(m')$, $[x \in U_{\mathrm{P1}(m')} \mathbf{\Delta} U_{\mathrm{P1}(m)}]$.

In this case, there exists a $k > m'$, such that $\varphi_{\mathrm{P2}(m)}(k)\downarrow = (x, \chi_L(x)) = (x, 1 - \chi_{U_{\mathrm{P1}(m')}})$. Thus, for large enough s, $m' \in B_s$ and thus $m' \notin Q_n^s$.

Claim follows from the above cases. $\square$

From Claims 11 and 12 it follows that, for $n \geq n_0$, $W_{prog(T[n])} = L$. Thus, $\mathbf{M}_{f(i)}$ **NoisyRecInfBc**-identifies $\mathcal{U}_i$. ∎

As a corollary to Lemma 2(b) and Theorem 15 we have the second main, positive result of the present paper:

Corollary 2 $(\exists f \in \mathcal{R})(\forall i \mid \mathcal{U}_i \in \mathbf{NoisyRecInfBc})[\mathcal{U}_i \subseteq \mathbf{NoisyRecInfBc}(\mathbf{M}_{f(i)})]$.

The following corollary to Lemma 2(b) and Theorem 15 provides the very nice characterization of indexed families in **NoisyRecInfBc**.[14]

Corollary 3 $\mathcal{U}_i \in \mathbf{NoisyRecInfBc} \Leftrightarrow$ *for all $L \in \mathcal{U}_i$, there exists a z such that, for all $L' \in \mathcal{U}_i$, $[(\{x \leq z \mid x \in L\} = \{x \leq z \mid x \in L'\}) \Rightarrow L' \subseteq L]$.*

For $n > 0$, we do not know about synthesizing learners for $\mathcal{U}_i \in \mathbf{NoisyRecInfBc}^n$.

[14]Hence, as was noted in Section 1 above, we have: $\{L \mid \mathrm{card}(N - L)$ is finite $\} \in (\mathbf{NoisyRecInfBc} - \mathbf{NoisyInfBc})$.

4.2 Principal Results on Synthesizing From R.E. Indices

Theorem 16 $\neg(\exists f \in \mathcal{R})(\forall i \mid \mathcal{C}_i \in \mathbf{NoisyTxtEx} \cap \mathbf{NoisyInfEx})[\mathcal{C}_i \subseteq \mathbf{RecTxtBc}^n(\mathbf{M}_{f(x)})]$.

PROOF. Theorem 17 in [CJS99] showed $\neg(\exists f \in \mathcal{R})(\forall i \mid \mathcal{C}_i \in \mathbf{NoisyTxtEx} \cap \mathbf{NoisyInfEx})[\mathcal{C}_i \subseteq \mathbf{TxtBc}^n(\mathbf{M}_{f(x)})]$. The proof of this given in [CJS99] also shows that $\neg(\exists f \in \mathcal{R})(\forall i \mid \mathcal{C}_i \in \mathbf{NoisyTxtEx} \cap \mathbf{NoisyInfEx})[\mathcal{C}_i \subseteq \mathbf{RecTxtBc}^n(\mathbf{M}_{f(x)})]$. ∎

Corollary 4 $\neg(\exists f \in \mathcal{R})(\forall i \mid \mathcal{C}_i \in \mathbf{NoisyTxtEx} \cap \mathbf{NoisyInfEx})[\mathcal{C}_i \subseteq \mathbf{NoisyRecTxtBc}^n(\mathbf{M}_{f(x)})]$.

Corollary 5 $\neg(\exists f \in \mathcal{R})(\forall i \mid \mathcal{C}_i \in \mathbf{NoisyTxtEx} \cap \mathbf{NoisyInfEx})[\mathcal{C}_i \subseteq \mathbf{NoisyRecInfBc}^n(\mathbf{M}_{f(x)})]$.

4.3 Synthesizing Learners Employing Noise-Free Data

We first note that for recursive languages, identification from recursive informant is same as identification from general informants (since the canonical informants are recursive). For non-recursive languages, there are no recursive informants. Thus, we only consider $\mathbf{RecTxtBc}^a$ below. In this context, for learning from uniform decisions procedures, we have the following corollary to Theorem 13 above in Section 4.1.

Corollary 6 $(\exists f \in \mathcal{R})(\forall x)[\mathcal{U}_x \subseteq \mathbf{RecTxtBc}(\mathbf{M}_{f(x)})]$.

For learning from indices for r.e. classes, Theorem 16 shows $\neg(\exists f \in \mathcal{R})(\forall i \mid \mathcal{C}_i \in \mathbf{NoisyTxtEx} \cap \mathbf{NoisyInfEx})[\mathcal{C}_i \subseteq \mathbf{RecTxtBc}^n(\mathbf{M}_{f(x)})]$. Also, as a corollary to Theorem 7 we have $\mathcal{E} \in \mathbf{RecTxtBc}^*$.

5 A World of Limiting-Recursive Texts

As indicated in Section 1 above, in this section, we consider briefly what would happen if the world provided *limit recursive* data sequences (instead of computable or unrestricted ones).

One can extend the definition of learning from texts to limit recursive texts too. Since, for every r.e. language, the canonical informant is limit recursive, the notion of learning from limit recursive *informant* collapses to the notion of learning from arbitrary informants (for $\mathbf{Bc}^a$ and $\mathbf{Ex}^a$ style learning criteria).

Definition 9 [KR88] A text T is *normalized*, iff for all n, $T(n) = \#$, or $T(n) < n$. A sequence σ is *normalized* iff for all $n < |\sigma|$, $\sigma(n) = \#$, or $\sigma(n) < n$.

Note that one can algorithmically convert any (noisy) text for a language L to a normalized (noisy) text for L. Thus, any class of languages, which can be (**Noisy**)$\mathbf{TxtBc}^a$-identified from normalized texts, can also be (**Noisy**)$\mathbf{TxtBc}^a$-identified from arbitrary texts. We say that σ is a *normalized*-$\mathbf{TxtBc}^a$-*locking sequence for* $\mathbf{M}$ *on* L, iff σ is normalized, content$(\sigma) \subseteq L$, and for all normalized extensions τ of σ such that content$(\tau) \subseteq L$, $W_{\mathbf{M}(\tau)} =^a L$. Similarly, we say that σ is a *normalized*-$\mathbf{NoisyTxtBc}^a$-*locking sequence for* $\mathbf{M}$ *on* L, iff σ is normalized, and for all τ such that content$(\tau) \subseteq L$, and $\sigma \diamond \tau$ is normalized, $W_{\mathbf{M}(\sigma\diamond\tau)} =^a L$.

Proposition 3 *Suppose* $\mathbf{M}$ *and a r.e. language* L *are given. Suppose normalized sequence* σ *is such that, there exists a* τ, content$(\tau) \subseteq L$, *and* $\sigma \diamond \tau$ *is normalized, and* $W_{\mathbf{M}(\sigma\diamond\tau)} \neq L$. *Then one can find one such* τ, *limit effectively in* σ, $\mathbf{M}$, *and a grammar for* L.

Note that one can canonically index all the finite sequences. Below we identify finite sequences with their canonical indices. Thus, comparisons such as $\langle\sigma, m\rangle < \langle\tau, k\rangle$, mean the comparisons so formed by replacing the sequences with their canonical indices.

Theorem 17 *(a) Suppose* $\mathbf{M}$ $\mathbf{LimRecTxtBc}$*-identifies* L*. Then, for all normalized* σ *such that* $\mathrm{content}(\sigma) \subseteq L$*, there exists a* τ*, such that* $\mathrm{content}(\tau) \subseteq L$*,* $\sigma \diamond \tau$ *is normalized, and* $\sigma \diamond \tau$ *is a normalized-*$\mathbf{TxtBc}$*-locking sequence for* $\mathbf{M}$ *on* L*.*

(b) Suppose $\mathbf{M}$ $\mathbf{LimRecNoisyTxtBc}$*-identifies* L*. Then, for all normalized* σ*, there exists a* τ*, such that* $\mathrm{content}(\tau) \subseteq L$*,* $\sigma \diamond \tau$ *is normalized, and* $\sigma \diamond \tau$ *is a normalized-*$\mathbf{NoisyTxtBc}$*-locking sequence for* $\mathbf{M}$ *on* L*.*

Proof. We show part (a). Part (b) is similar. Suppose $\mathbf{M}$, L and σ are given as in hypothesis. Suppose by way of contradiction that there is no τ such that $\sigma \diamond \tau$ is normalized and $\sigma \diamond \tau$ is a normalized-$\mathbf{TxtBc}$-locking sequence for $\mathbf{M}$ on L. Then for all τ such that $\mathrm{content}(\tau) \subseteq L$ and $\sigma \diamond \tau$ is normalized, there exists a τ' such that $\mathrm{content}(\tau') \subseteq L$, $\sigma \diamond \tau \diamond \tau'$ is normalized and $W_{M(\sigma\diamond\tau\diamond\tau')} \neq L$.

Now suppose T is a recursive text for L. Now define σ_i as follows: $\sigma_0 = \sigma$. $\sigma_{2i+1} = \sigma_{2i} \diamond (T(i))$. $\sigma_{2i+2} = \sigma_{2i+1} \diamond \tau_{2i+1}$, where τ_{2i+1} is such that $\mathrm{content}(\tau_{2i+1}) \subseteq L$, σ_{2i+2} is normalized, $W_{\mathbf{M}(\sigma_{2i+2})} \neq L$, and τ_{2i+1} can be obtained in the limit from σ_{2i+1} (see Proposition 3). It follows that $T' = \bigcup_{i\in N} \sigma_i$ is a limit recursive text which is not $\mathbf{TxtBc}$-identified by $\mathbf{M}$. ∎

The proof of Theorem 4.9 in [KR88] also showed that

Theorem 18 *From a given* $\mathbf{M}$ *one can algorithmically generate an* $\mathbf{M}'$ *such that, for all* L*:*

If every normalized σ*, such that* $\mathrm{content}(\sigma) \subseteq L$*, has a normalized extension* σ'*, which is a normalized* $\mathbf{TxtBc}$*-locking sequence for* $\mathbf{M}$ *on* L*, then* $\mathbf{M}'$ $\mathbf{TxtBc}$*-identifies* L *from normalized texts.*

As a corollary we have

Corollary 7 $\mathbf{TxtBc} = \mathbf{LimRecTxtBc}$*. Moreover, for any* $\mathbf{M}$*, one can algorithmically find* $\mathbf{M}'$ *such that* $\mathbf{LimRecTxtBc}(\mathbf{M}) \subseteq \mathbf{TxtBc}(\mathbf{M}')$*.*

The proof of Theorem 18 can be generalized to show

Theorem 19 *From a given* $\mathbf{M}$ *one can algorithmically generate an* $\mathbf{M}'$ *such that, for all* L*:*

If for every normalized σ*, there exists a* τ*, such that* $\mathrm{content}(\tau) \subseteq L$*,* $\sigma \diamond \tau$ *is normalized and* $\sigma \diamond \tau$ *is a* $\mathbf{NoisyTxtBc}$*-locking sequence for* L*, then* $\mathbf{M}'$ $\mathbf{NoisyTxtBc}$*-identifies* L *from normalized texts.*

Proof. Define $W_{\mathbf{M}'(T[n])}$ as follows:

$W_{\mathbf{M}'(T[n])} = \{x \mid (\exists \text{ normalized } \sigma)(\exists m)[(\forall \tau \mid \mathrm{content}(\tau) \subseteq T[m:n] \text{ and } \sigma \diamond \tau \text{ is normalized}) [x \in W_{\mathbf{M}(\sigma\diamond\tau)}] \wedge (\forall \text{ normalized } \sigma')(\forall m' \mid \langle\sigma', m'\rangle \leq \langle\sigma, m\rangle)(\exists \tau \mid \mathrm{content}(\tau) \subseteq T[m':n] \text{ and } \sigma' \diamond \tau \text{ is normalized})[x \in W_{\mathbf{M}(\sigma'\diamond\tau)}]]\}$.

Now suppose T is a noisy text for $L \in \mathbf{LimRecNoisyTxtBc}(\mathbf{M})$. Let α, k be such that α is normalized $\mathbf{NoisyTxtBc}$-locking sequence for $\mathbf{M}$ on L, and $T[k:\infty]$ is a text for L. Note that there exist such α and k. For all $\langle\alpha', k'\rangle \leq \langle\alpha, k\rangle$, let $S(\alpha', k')$ be sequence such that $\mathrm{content}(S(\alpha', k')) \subseteq L$, and $\alpha' \diamond S(\alpha', k')$ is a $\mathbf{NoisyTxtBc}$ locking sequence for $\mathbf{M}$ on L. Let n_0 be so large that, for all $\langle\alpha', k'\rangle \leq \langle\alpha, k\rangle$, $\mathrm{content}(S(\alpha', k')) \subseteq \mathrm{content}(T[k':n_0])$.

We then claim that, for $n \geq n_0$, $W_{\mathbf{M}'(T[n])} = L$.

Claim 13 *For* $n \geq n_0$*,* $L \subseteq W_{\mathbf{M}'(T[n])}$*.*

PROOF. For $n \geq n_0$, by choosing $\sigma = \alpha$ and $m = k$, and for $\langle \sigma', m' \rangle \leq \langle \alpha, m \rangle$, choosing $\tau = S(\sigma', m')$, in the definition of $W_{\mathbf{M}'(T[n])}$, it is easy to verify that $L \subseteq W_{\mathbf{M}'(T[n])}$. $\square$

Claim 14 *For $n \geq n_0$, $W_{\mathbf{M}'(T[n])} \subseteq L$.*

PROOF. Suppose $x \in W_{\mathbf{M}'(T[n])}$. Let σ, m be as chosen in the definition of $W_{\mathbf{M}'(T[n])}$ due to which x is included in $W_{\mathbf{M}'(T[n])}$. We consider two cases:

Case 1: $\langle \sigma, m \rangle \geq \langle \alpha, k \rangle$

In this case, due to existance of τ such that $\alpha \diamond \tau$ is normalized and content$(\tau) \subseteq T[k:\infty]$, and $x \in W_{\mathbf{M}(\alpha \diamond \tau)}$, we immediately have that $x \in L$ (since α is normalized-**NoisyTxtBc**-locking sequence for $\mathbf{M}$ on L, and $T[k:\infty]$ is a text for L).

Case 2: $\langle \sigma, m \rangle < \langle \alpha, k \rangle$

In this case, since for $\tau = S(\sigma, m)$, $x \in W_{\mathbf{M}(\sigma \diamond S(\sigma,m))}$, we have that $x \in L$ (since $\sigma \diamond S(\sigma, m)$ is normalized-**NoisyTxtBc**-locking sequence for $\mathbf{M}$ on L).

From the above cases, claim follows. $\square$

The theorem follows from the above claims. ∎

Corollary 8 **NoisyTxtBc** = **LimRecNoisyTxtBc**. *Moreover, for any* $\mathbf{M}$, *one can algorithmically find* $\mathbf{M}'$ *such that* $\mathbf{LimRecNoisyTxtBc}(\mathbf{M}) \subseteq \mathbf{NoisyTxtBc}(\mathbf{M}')$.

It is presently open whether, for $n > 0$, $\mathbf{NoisyTxtBc}^n = \mathbf{LimRecNoisyTxtBc}^n$. It is also open whether $\mathbf{NoisyInfBc}^n = \mathbf{LimRecNoisyInfBc}^n$, for $n \geq 0$.

6 Conclusions and Future Directions

In a computable universe, all data sequences, even noisy ones, are computable. Based on this, we studied in this paper the effects of having computable noisy data as input. In addition to comparing the criteria so formed within themselves and with related criteria from the literature, we studied the problem of synthesizing learners for r.e. classes and indexed families of recursive languages. The main result of the paper (Theorem 13) showed that *all* indexed families of recursive languages can be learned (in **Bc**-sense) from computable noisy texts. Moreover, one can algorithmically find a learner doing so, from an index for any indexed family. Another main positive result of the paper, Corollary 3, gives a characterization of indexed families which can be learned (in **Bc**-sense) from computable noisy informant.

It is interesting to extend the study to the case where the texts have some other restriction than the computability restriction we considered in this paper. In this regard we briefly considered limiting recursive texts. One of the surprising results we have here is that **TxtBc** = **LimRecTxtBc** and **NoisyTxtBc** = **LimRecNoisyTxtBc**. One can also similarly consider texts from natural *subrecursive* classes [RC94], linear-time computable and above. From [Gol67, Cas86], in that setting, some machine learns $\mathcal{E}$. *However*, it remains to determine the possible tradeoffs between the complexity of the texts and useful complexity features of the resultant learners. [Cas86] mentions that, in some cases, subrecursiveness of texts forces infinite repetition of data. Can this be connected to complexity tradeoffs? [Cas86] further notes that, if the texts we present to children, contain many repetitions, that would be consistent with a restriction in the world to subrecursive texts.

References

[Ang80] D. Angluin. Inductive inference of formal languages from positive data. *Information and Control*, 45:117–135, 1980.

[AS83] D. Angluin and C. Smith. Inductive inference: Theory and methods. *Computing Surveys*, 15:237–289, 1983.

[AS94] H. Arimura and T. Shinohara. Inductive inference of Prolog programs with linear data dependency from positive data. In H. Jaakkola, H. Kangassalo, T. Kitahashi, and A. Markus, editors, *Proc. Information Modelling and Knowledge Bases V*, pages 365–375. IOS Press, 1994.

[ASY92] S. Arikawa, T. Shinohara, and A. Yamamoto. Learning elementary formal systems. *Theoretical Computer Science*, 95:97–113, 1992.

[B$\bar{7}$4] J. Bārzdiņš. Two theorems on the limiting synthesis of functions. In *Theory of Algorithms and Programs, vol. 1*, pages 82–88. Latvian State University, 1974. In Russian.

[BB75] L. Blum and M. Blum. Toward a mathematical theory of inductive inference. *Information and Control*, 28:125–155, 1975.

[BCJ99] G. Baliga, J. Case, and S. Jain. The synthesis of language learners. *Information and Computation*, 152(1):16–43, July 1999.

[Ber85] R. Berwick. *The Acquisition of Syntactic Knowledge*. MIT Press, 1985.

[Blu67] M. Blum. A machine-independent theory of the complexity of recursive functions. *Journal of the ACM*, 14:322–336, 1967.

[BP73] J. Bārzdiņš and K. Podnieks. The theory of inductive inference. In *Second Symposium on Mathematical Foundations of Computer Science*, pages 9–15. Math. Inst. of the Slovak Academy of Sciences, 1973.

[Cas74] J. Case. Periodicity in generations of automata. *Mathematical Systems Theory*, 8:15–32, 1974.

[Cas86] J. Case. Learning machines. In W. Demopoulos and A. Marras, editors, *Language Learning and Concept Acquisition*. Ablex Publishing Company, 1986.

[Cas92] J. Case. Turing machine. In Stuart Shapiro, editor, *Encyclopedia of Artificial Intelligence*. John Wiley and Sons, New York, NY, second edition, 1992.

[Cas99] J. Case. The power of vacillation in language learning. *SIAM Journal on Computing*, 28(6):1941–1969, 1999.

[CJLZ99] J. Case, S. Jain, S. Lange, and T. Zeugmann. Incremental concept learning for bounded data mining. *Information and Computation*, 152(1):74–110, July 1999.

[CJS99] J. Case, S. Jain, and A. Sharma. Synthesizing noise-tolerant language learners. *Theoretical Computer Science A*, 1999. Special Issue for ALT'97, to appear.

[CJS00] J. Case, S. Jain, and F. Stephan. Vacillatory and BC learning on noisy data. *Theoretical Computer Science A*, 241:115–141, 2000. Special Issue for ALT'96.

[CL82] J. Case and C. Lynes. Machine inductive inference and language identification. In M. Nielsen and E. M. Schmidt, editors, *Proceedings of the 9th International Colloquium on Automata, Languages and Programming*, volume 140 of *Lecture Notes in Computer Science*, pages 107–115. Springer-Verlag, 1982.

[CRS94] J. Case, D. Rajan, and A. Shende. Representing the spatial/kinematic domain and lattice computers. *Journal of Experimental and Theoretical Artificial Intelligence*, 6:17–40, 1994.

[CS83] J. Case and C. Smith. Comparison of identification criteria for machine inductive inference. *Theoretical Computer Science*, 25:193–220, 1983.

[dJK96] D. de Jongh and M. Kanazawa. Angluin's thoerem for indexed families of r.e. sets and applications. In *Proceedings of the Ninth Annual Conference on Computational Learning Theory*, pages 193–204. ACM Press, July 1996.

[dMSS56] K. deLeeuw, E. Moore, C. Shannon, and N. Shapiro. Computability by probabilistic machines. *Automata Studies, Annals of Math. Studies*, 34:183–212, 1956.

[Fey82] R. Feynman. Simulating physics with computers. *International Journal of Theoretical Physics*, 21(6/7), 1982.

[FJ96] M. Fulk and S. Jain. Learning in the presence of inaccurate information. *Theoretical Computer Science A*, 161:235–261, 1996.

[Fre85] R. Freivalds. Recursiveness of the enumerating functions increases the inferrability of recursively enumerable sets. *Bulletin of the European Association for Theoretical Computer Science*, 27:35–40, 1985.

[Ful90] M. Fulk. Robust separations in inductive inference. In *31st Annual IEEE Symposium on Foundations of Computer Science*, pages 405–410. IEEE Computer Society Press, 1990.

[Gil72] J. Gill. *Probabilistic Turing Machines and Complexity of Computation*. PhD thesis, University of California, Berkeley, 1972.

[Gil77] J. Gill. Computational complexity of probabilistic Turing machines. *SIAM Journal of Computing*, 6:675–695, 1977.

[Gol67] E. M. Gold. Language identification in the limit. *Information and Control*, 10:447–474, 1967.

[Jai96] S. Jain. Program synthesis in the presence of infinite number of inaccuracies. *Journal of Computer and System Sciences*, 53(1):583–591, 1996.

[Jan79a] K. Jantke. Automatic synthesis of programs and inductive inference of functions. In *Int. Conf. Fundamentals of Computations Theory*, pages 219–225. Akademie-Verlag, Berlin, 1979.

[Jan79b] K. Jantke. Natural properties of strategies identifying recursive functions. *Journal of Information Processing and Cybernetics (EIK)*, 15:487–496, 1979.

[JORS99] S. Jain, D. Osherson, J. Royer, and A. Sharma. *Systems that Learn: An Introduction to Learning Theory.* MIT Press, Cambridge, Mass., second edition, 1999.

[Kap91] S. Kapur. *Computational Learning of Languages.* PhD thesis, Cornell University, 1991.

[KB92] S. Kapur and G. Bilardi. Language learning without overgeneralization. In A. Finkel and M. Jantzen, editors, *Proceedings of the Ninth Annual Symposium on Theoretical Aspects of Computer Science*, volume 577 of *Lecture Notes in Computer Science*, pages 245–256. Springer-Verlag, 1992.

[KMU95] P. Kilpeläinen, H. Mannila, and E. Ukkonen. MDL learning of unions of simple pattern languages from positive examples. In Paul Vitányi, editor, *Second European Conference on Computational Learning Theory*, volume 904 of *Lecture Notes in Artificial Intelligence*, pages 252–260. Springer-Verlag, 1995.

[KR88] S. Kurtz and J. Royer. Prudence in language learning. In D. Haussler and L. Pitt, editors, *Proceedings of the Workshop on Computational Learning Theory*, pages 143–156. Morgan Kaufmann, 1988.

[KW80] R. Klette and R. Wiehagen. Research in the theory of inductive inference by GDR mathematicians – A survey. *Information Sciences*, 22:149–169, 1980.

[LD94] N. Lavarač and S. Džeroski. *Inductive Logic Programming.* Ellis Horwood, New York, 1994.

[LZK96] S. Lange, T. Zeugmann, and S. Kapur. Monotonic and dual monotonic language learning. *Theoretical Computer Science A*, 155:365–410, 1996.

[MR94] S. Muggleton and L. De Raedt. Inductive logic programming: Theory and methods. *Journal of Logic Programming*, 19/20:669–679, 1994.

[Muk92] Y. Mukouchi. Characterization of finite identification. In K. Jantke, editor, *Analogical and Inductive Inference, Proceedings of the Third International Workshop*, pages 260–267, 1992.

[Nix83] R. Nix. Editing by examples. Technical Report 280, Department of Computer Science, Yale University, New Haven, CT, USA, 1983.

[Odi89] P. Odifreddi. *Classical Recursion Theory.* North-Holland, Amsterdam, 1989.

[OSW86] D. Osherson, M. Stob, and S. Weinstein. *Systems that Learn: An Introduction to Learning Theory for Cognitive and Computer Scientists.* MIT Press, 1986.

[OSW88] D. Osherson, M. Stob, and S. Weinstein. Synthesising inductive expertise. *Information and Computation*, 77:138–161, 1988.

[RC94] J. Royer and J. Case. *Subrecursive programming systems: Complexity & succinctness.* Birkhäuser, 1994.

[SA95] T. Shinohara and A. Arikawa. Pattern inference. In Klaus P. Jantke and Steffen Lange, editors, *Algorithmic Learning for Knowledge-Based Systems*, volume 961 of *Lecture Notes in Artificial Intelligence*, pages 259–291. Springer-Verlag, 1995.

[Sha71] N. Shapiro. Review of "Limiting recursion" by E.M. Gold and "Trial and error predicates and the solution to a problem of Mostowski" by H. Putnam. *Journal of Symbolic Logic*, 36:342, 1971.

[Shi83] T. Shinohara. Inferring unions of two pattern languages. *Bulletin of Informatics and Cybernetics*, 20:83–88., 1983.

[Shi91] T. Shinohara. Inductive inference of monotonic formal systems from positive data. *New Generation Computing*, 8:371–384, 1991.

[Smu61] R. Smullyan. *Theory of Formal Systems, Annals of Mathematical Studies, No. 47.* Princeton, NJ, 1961.

[Soa87] R. Soare. *Recursively Enumerable Sets and Degrees.* Springer-Verlag, 1987.

[SR85] G. Schäfer-Richter. Some results in the theory of effective program synthesis - learning by defective information. In W. Bibel and K. Jantke, editors, *Mathematical Methods of Specification and Synthesis of Software Systems, Wendisch-Rietz, GDR*, volume 215 of *Lecture Notes in Computer Science*, pages 219–225. Springer-Verlag, 1985.

[SSS+94] S. Shimozono, A. Shinohara, T. Shinohara, S. Miyano, S. Kuhara, and S. Arikawa. Knowledge acquisition from amino acid sequences by machine learning system BONSAI. *Trans. Information Processing Society of Japan*, 35:2009–2018, 1994.

[Ste95] F. Stephan. Noisy inference and oracles. In K. Jantke, T. Shinohara, and T. Zeugmann, editors, *Algorithmic Learning Theory: Sixth International Workshop (ALT '95)*, volume 997 of *Lecture Notes in Artificial Intelligence*, pages 185–200. Springer-Verlag, 1995.

[TM87] T. Toffoli and N. Margolus. *Cellular Automata Machines.* MIT Press, 1987.

[Tof77] T. Toffoli. Cellular automata machines. Technical Report 208, Comp. Comm. Sci. Dept., University of Michigan, 1977.

[Wie77] R. Wiehagen. Identification of formal languages. In *Mathematical Foundations of Computer Science*, volume 53 of *Lecture Notes in Computer Science*, pages 571–579. Springer-Verlag, 1977.

[Wri89] K. Wright. Identification of unions of languages drawn from an identifiabl e class. In R. Rivest, D. Haussler, and M.K. Warmuth, editors, *Proceedings of the Second Annual Workshop on Computational Learning Theory*, pages 328–333. Morgan Kaufmann Publishers, Inc., 1989.

[ZL95] T. Zeugmann and S. Lange. A guided tour across the boundaries of learning recursive languages. In K. Jantke and S. Lange, editors, *Algorithmic Learning for Knowledge-Based Systems*, volume 961 of *Lecture Notes in Artificial Intelligence*, pages 190–258. Springer-Verlag, 1995.

[ZLK95] T. Zeugmann, S. Lange, and S. Kapur. Characterizations of monotonic and dual monotonic language learning. *Information and Computation*, 120:155–173, 1995.

[Zus69] K. Zuse. *Rechnender Raum.* Vieweg, Braunshweig, 1969. Translated as *Calculating Space*, Tech. Transl. AZT–70–164–GEMIT, MIT Project MAC, 1970.